drones

MDPI

Article

Combining Camera–LiDAR Fusion and Motion Planning Using Bird's-Eye View Representation for End-to-End Autonomous Driving

Ze Yu [1], Jun Li [1], Yuzhen Wei [1], Yuandong Lyu [2] and Xiaojun Tan [1,*]

[1] School of Intelligent Systems Engineering, Shenzhen Campus of Sun Yat-sen University, Shenzhen 518107, China; yuze5@mail2.sysu.edu.cn (Z.Y.); stslijun@mail.sysu.edu.cn (J.L.); weiyzh25@mail2.sysu.edu.cn (Y.W.)
[2] Li Auto Inc., Shanghai 201800, China; lvyuandong@lixiang.com
* Correspondence: tanxj@mail.sysu.edu.cn

Abstract: End-to-end autonomous driving has become a key research focus in autonomous vehicles. However, existing methods struggle with effectively fusing heterogeneous sensor inputs and converting dense perceptual features into sparse motion representations. To address these challenges, we propose BevDrive, a novel end-to-end autonomous driving framework that unifies camera–LiDAR fusion and motion planning through a bird's-eye view (BEV) representation. BevDrive consists of three core modules: the bidirectionally guided BEV feature construction module, the dual-attention BEV feature fusion module, and the BEV-based motion planning module. The bidirectionally guided BEV feature construction module comprises two branches: depth-guided image BEV feature construction and image-guided LiDAR BEV feature construction. Depth-guided image BEV feature construction employs a lifting and projection approach guided by depth information from LiDAR, transforming image features into a BEV representation. Meanwhile, image-guided LiDAR BEV feature construction enriches sparse LiDAR BEV features by integrating complementary information from the images. Then, the dual-attention BEV feature fusion module combines multi-modal BEV features at both local and global levels using a hybrid approach of window self-attention and global self-attention mechanisms. Finally, the BEV-based motion planning module integrates perception and planning by refining control and trajectory queries through interactions with the scene context in the fused BEV features, generating precise trajectory points and control commands. Extensive experiments on the CARLA Town05 Long benchmark demonstrate that BevDrive achieves state-of-the-art performance. Furthermore, we validate the feasibility of the proposed algorithm on a real-world vehicle platform, confirming its practical applicability and robustness.

Keywords: deep learning; autonomous driving; end-to-end autonomous driving; sensor fusion; motion planning

Academic Editor: Pablo Rodríguez-Gonzálvez

Received: 2 March 2025
Revised: 27 March 2025
Accepted: 4 April 2025
Published: 8 April 2025

Citation: Yu, Z.; Li, J.; Wei, Y.; Lyu, Y.; Tan, X. Combining Camera–LiDAR Fusion and Motion Planning Using Bird's-Eye View Representation for End-to-End Autonomous Driving. *Drones* **2025**, *9*, 281. https://doi.org/10.3390/drones9040281

1. Introduction

Autonomous driving technology is revolutionizing the way people travel, offering the potential for safer, more efficient, and more convenient transportation. Traditional autonomous driving systems are typically designed using a modular pipeline, where the core functions of perception [1–3], prediction [4–6], planning [7–9], and control [10–12] are implemented as separate components. While this modular approach has been effective in many scenarios, it often suffers from error propagation across modules, limiting

overall system performance. With the rapid advancements in artificial intelligence and computational power, a new paradigm known as end-to-end autonomous driving has emerged [13–17]. This paradigm leverages deep learning to directly map raw sensor data to future trajectory points or control commands, enabling a globally differentiable model for autonomous driving. Unlike traditional modular pipelines, end-to-end methods optimize system performance holistically, mitigating potential bottlenecks caused by intermediate errors.

End-to-end autonomous driving requires models to directly infer a comprehensive understanding of the environment from raw sensor data and generate safe, accurate motion plans. Early approaches [18,19] primarily relied on visual inputs, using RGB images to produce control commands via deep neural networks, showcasing the feasibility of end-to-end driving. However, vision-based methods face significant limitations in complex scenarios, such as dynamic interactions, dense traffic, and adverse weather conditions (e.g., rain or fog), due to their inherent weaknesses in depth perception and susceptibility to lighting variations and occlusions. To overcome these challenges, recent approaches [20,21] leverage multi-modal data fusion, combining inputs from cameras and LiDAR to complement the strengths and weaknesses of each modality. Current multi-modal end-to-end autonomous driving systems are typically divided into two key stages: sensor encoding and motion decoding. In the sensor encoding stage, the model must effectively fuse heterogeneous data from different modalities, such as the rich 2D color and texture information from RGB images and the precise 3D geometric data from LiDAR point clouds. Bridging the gap between these vastly different data modalities and integrating them seamlessly remains a significant technical challenge. Early end-to-end autonomous driving approaches [21,22] improved performance by simply concatenating LiDAR and image features as inputs to the sensor encoding network. Subsequent studies [23–26] explored Transformer-based methods, utilizing attention mechanisms to facilitate information interaction and fusion between image and LiDAR point cloud features. However, these methods have not fully addressed the intrinsic gaps between sensor data in both the data space and feature space, which constrain the effectiveness of the fusion process. In the motion decoding stage, the model must translate dense perceptual features into sparse motion representations, such as trajectory points or control commands. This transformation is particularly challenging due to the significant differences in the spatial distributions of perceptual and motion features. However, the aforementioned methods often rely on pooled or hybrid features as inputs to the motion decoder, failing to tightly integrate motion planning and scene encoding within a unified feature space. This shortcoming results in a loss of critical spatial information. Addressing these challenges is crucial for developing robust and reliable end-to-end autonomous driving systems.

To address the challenges outlined above, this paper introduces a camera–LiDAR fusion and motion planning framework using bird's-eye view (BEV) representation for end-to-end autonomous driving, abbreviated as BevDrive. BevDrive utilizes a BEV representation to effectively unify multi-modal scene perception and motion planning within a shared feature space, enabling seamless integration and collaboration between the two tasks. The framework consists of three core modules: the bidirectionally guided BEV feature construction module, the dual-attention BEV feature fusion module, and the BEV-based motion planning module. During the scene encoding stage, the bidirectionally guided BEV feature construction module employs a dual-stream network to independently extract features from image and LiDAR data, mapping both modalities into a unified BEV feature plane. This effectively addresses the spatial gap between the two data types. In the image branch, the depth-guided image BEV feature construction leverages a lifting and projection approach, utilizing depth information from LiDAR point clouds to transform

image features into a BEV representation. Meanwhile, in the LiDAR branch, image-guided LiDAR BEV feature construction integrates complementary information from the image branch to refine the sparse LiDAR BEV features, enhancing the completeness and accuracy of the feature representation. Building on this, the dual-attention BEV feature fusion module adopts a hybrid approach that combines window self-attention and global self-attention mechanisms to deeply fuse multi-modal BEV features at both local and global levels. After fusing the multi-modal BEV features, the BEV-based motion planning module utilizes learnable trajectory queries and control queries to extract scene context information from the fused BEV features and generate motion plans for the ego vehicle. Specifically, self-attention mechanisms are employed to capture the relationships between trajectory and control queries, while cross-attention mechanisms facilitate interactions between these queries and the fused BEV features, dynamically refining the results. Finally, an output fusion strategy integrates the trajectory and control commands, delivering robust and reliable outputs for vehicle control.

Our main contributions are summarized as follows:

- We propose BevDrive, a novel end-to-end framework that unifies sensor fusion and motion planning within a unified BEV representation, bridging the gap between perception and planning process.
- We propose a bidirectionally guided BEV feature construction module, which aligns 2D and 3D data by projecting image and LiDAR features into a unified BEV space, eliminating spatial discrepancies.
- We design a dual-attention mechanism that integrates multi-modal BEV features at both local and global levels, enhancing contextual reasoning.
- We develop a BEV-based motion planning module that dynamically refines trajectory and control queries through interaction with fused BEV features to generate robust trajectory points and control commands.

2. Related Work

2.1. End-to-End Autonomous Driving

End-to-end autonomous driving has been widely explored through two major paradigms: imitation learning (IL) [27] and reinforcement learning (RL) [28]. IL-based methods aim to mimic expert demonstrations to optimize perception-to-control pipelines. Early works like CIL [14] introduced conditional branching networks guided by navigation commands to predict control commands. LBC [29] adopted privileged expert supervision to enhance feature learning, while LAV [30] incorporated surrounding vehicle trajectories to supervise ego-vehicle predictions. In contrast, RL-based approaches optimize policies by interacting with the environment and learning through feedback signals. For example, Maximilian et al. [31] utilized RL with novel reward designs to achieve robust end-to-end driving from RGB images. Roach [32] proposed an RL-based method that leverages privileged simulation states as an expert to supervise imitation learning. Although these RL-based methods demonstrate the feasibility of learning driving policies through exploration, they often require extensive simulation training with dense rewards to achieve stability and performance. However, RL-based approaches face challenges such as the simulation-to-reality gap and the complexities of reward design. To address these challenges and enable real-world deployment, we adopt an IL-based framework to build our BevDrive.

2.2. Multi-Modal Fusion for End-to-End Autonomous Driving

Multi-modal fusion has become a vital approach to enhancing the robustness of end-to-end autonomous driving. Early methods [33–35] primarily relied on camera-based

inputs, directly predicting driving actions from visual features. While these methods perform well in simple scenarios, they struggle to generalize in complex and dynamic environments due to the lack of depth perception and their susceptibility to variations in weather and lighting conditions. To address these challenges, recent research [15,25,26,29] has focused on combining data from cameras and LiDAR to improve perception and planning capabilities. TransFuser [15] integrated image and LiDAR features into a hybrid representation through self-attention, enabling better fusion of complementary data modalities. LAV [29] adopted a point-painting [36] strategy to annotate LiDAR points with semantic features from images, generating BEV features for downstream tasks. InterFuser [25] further leveraged Transformer-based architectures to fuse multi-view images and LiDAR inputs, significantly enhancing scene understanding and motion planning by capturing global contextual information. Despite these advancements, multi-modal fusion methods still face challenges such as sensor misalignment, where discrepancies between camera and LiDAR data compromise the quality of fused representations. To overcome these limitations, we propose a bidirectionally guided BEV feature construction module that aligns 2D and 3D data by projecting image and LiDAR features into a unified BEV space, effectively eliminating spatial discrepancies between modalities. Furthermore, we design a dual-attention mechanism to integrate multi-modal BEV features at both local and global levels.

2.3. Motion Planning for End-to-End Autonomous Driving

Early end-to-end autonomous driving motion planning methods [37] employed multi-layer perceptrons (MLPs) to directly map perceptual features to control commands, such as throttle, brake, and steering. Building on this foundation, CIL [14] introduced navigation commands as conditional inputs to guide the selection of specialized submodules, enabling the generation of task-specific control commands to resolve decision-making ambiguities at intersections. TransFuser [23] proposed an autoregressive GRU-based framework that generates trajectory points from latent scene features, significantly improving motion planning for point-to-point navigation. TCP [38] bridged the gap between trajectory planning and direct control through a dual-branch architecture, leveraging the complementary strengths of both approaches. InterFuser [25] addressed the shortcut learning issue observed in TransFuser by introducing a Transformer-GRU architecture that combines global attention with temporal modeling, further enhancing the robustness and accuracy of trajectory planning. However, these methods used pooled or hybrid features as inputs to the motion decoder, failing to integrate motion planning and scene encoding into a unified feature space, which leads to a loss of spatial information. Our BevDrive addresses this limitation by unifying sensor fusion and motion planning within a BEV feature space, utilizing trajectory and control queries to extract scene context for more reliable motion planning.

3. Methodology

The overall architecture of the proposed BevDrive is depicted in Figure 1, comprising three key components: (1) a bidirectionally guided BEV feature construction module, where image and LiDAR features are extracted via a dual-branch network and transformed into a unified BEV space; (2) a dual-attention BEV feature fusion module, which fuses multi-modal BEV features from local to global levels; and (3) a BEV-based motion planning decoder, which predicts future trajectory points and control commands using scene context in the fused BEV features. In the following sections, we first formulate the inputs and outputs in Section 3.1, followed by a detailed discussion of the various components of BevDrive in Sections 3.2–3.5.

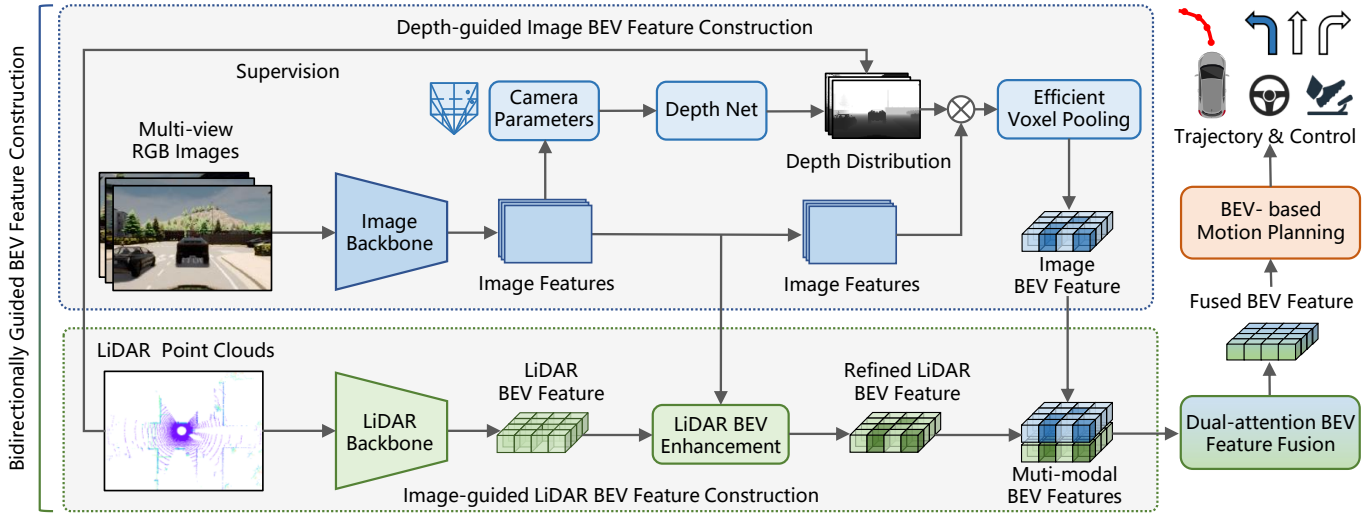


Figure 1. Overview of BevDrive. In the scene encoding stage, a bidirectional BEV feature construction module extracts image and LiDAR features separately and transforms them into a unified BEV space using depth-guided image BEV feature construction and image-guided LiDAR BEV feature construction. A dual-attention mechanism integrates multi-modal BEV features at both local and global levels. In the motion decoding stage, a BEV-based decoder predicts future trajectory points and control commands by leveraging scene context in the fused BEV feature.

3.1. Input and Output Representation

3.1.1. Input Representation

For RGB image inputs, we utilize three cameras oriented forward, 60° to the left, and 60° to the right. Each camera captures images at a resolution of 300 × 400 pixels with a horizontal field of view (FOV) of 60°. Together, these cameras provide a combined horizontal FOV of 180° for the ego vehicle. The raw images are cropped to 288 × 400 pixels before being processed by the image feature extraction backbone. For LiDAR inputs, the raw point cloud is voxelized with a resolution of 0.2 m.

3.1.2. Output Representation

BevDrive generates two types of outputs: trajectory points and control commands. For trajectory point prediction, the model predicts a future trajectory to guide the ego vehicle, represented as $\{wp_t\}_{t=1}^{T}$, where $wp_t \in \mathbb{R}^{2\times 1}$ denotes the 2D coordinates of the ego vehicle at time step t. For control command prediction, the model outputs the current steering angle, throttle, and brake values, denoted as $\{steer_{\text{ctr}}, throttle_{\text{ctr}}, brake_{\text{ctr}}\}$.

3.2. Bidirectionally Guided BEV Feature Construction Module

To eliminate the spatial discrepancies between images and LiDAR point clouds and fully exploit the complementarity of different modalities, we propose a bidirectionally guided BEV feature construction module. This module consists of two key modules: depth-guided image BEV feature construction and image-guided LiDAR BEV feature construction. The depth-guided image BEV feature construction module transforms 2D image features into the BEV space using depth estimation, while the image-guided LiDAR BEV feature construction module enriches LiDAR BEV features with semantic information extracted from image features. Collectively, these modules unify image and LiDAR data into a shared BEV feature space, facilitating the effective fusion of multi-modal data.

3.2.1. Depth-Guided Image BEV Feature Construction

As illustrated in Figure 1, for RGB image inputs, we first utilize an image backbone to extract basic image features, formulated as follows:

$$F_{\text{img}} = \text{ConvBlock}(I), \tag{1}$$

where $\text{ConvBlock}(\cdot)$ represents the convolution operation performed by the image backbone, I denotes the input images, and $F_{\text{img}} \in \mathbb{R}^{N \times H_i \times W_i \times C_i}$ is the resulting extracted image feature. Here, N is the number of cameras, H_i and W_i are the spatial dimensions of the feature maps, and C_i is the number of feature channels.

After extracting image features, as shown in Figure 2, the 2D image features are lifted into 3D space through depth estimation and then projected onto the BEV plane using efficient voxel pooling. Specifically, a depth estimation network predicts the discrete depth distribution $D_{\text{img}} \in \mathbb{R}^{N \times H_i \times W_i \times D_i}$, which is aligned with the image features F_{img}, where D_i represents the number of discrete depth levels. Finally, the frustum feature $\bar{F}_{\text{img}}$ is computed by performing an outer product between the image features F_{img} and their corresponding depth distribution D_{img}.

$$\bar{F}_{\text{img}} = D_{\text{img}} \otimes F_{\text{img}}, \tag{2}$$

where $\otimes$ denotes outer product operation, and the resulting frustum feature $\bar{F}_{\text{img}} \in \mathbb{R}^{N \times H_i \times W_i \times D \times C_i}$. With the intrinsic and extrinsic parameters of the camera, each voxel in the frustum can be mapped to the corresponding grid in the BEV. The final image BEV feature, $B_{\text{img}} \in \mathbb{R}^{H \times W \times C}$, is generated through sum pooling, where H and W represent the spatial dimensions of the image BEV feature, and C denotes the number of channels.

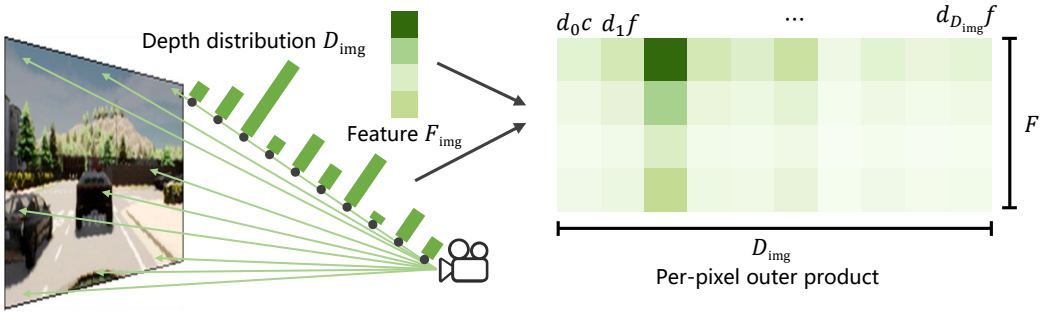
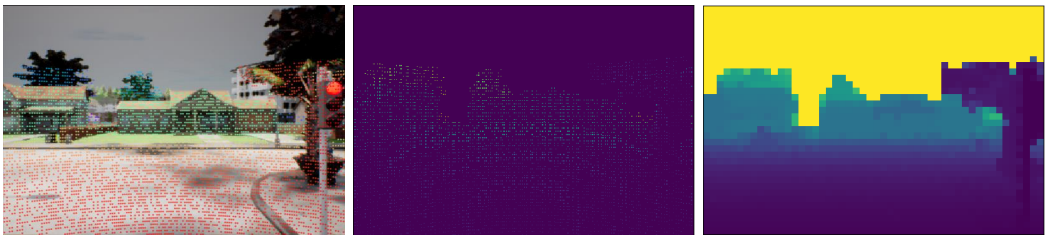


Figure 2. The lifting and projection approach in depth-guided image BEV feature construction. The 2D image features are lifted into 3D space using depth estimation, where the discrete depth distribution is aligned with the image features.

In the depth distribution estimation process, the sparse LiDAR point cloud is projected onto the image coordinate system using intrinsic and extrinsic transformation matrices to generate a sparse depth map, which is then refined through spatial interpolation to produce a dense depth map, as shown in Figure 3. Concretely, the sparse depth map $D_{\text{img}}^{\text{sparse}}$ is obtained by projecting the LiDAR point cloud onto the image plane, as described by the following equation:

$$D_{\text{img}}^{\text{sparse}} = K \cdot \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \tag{3}$$

where $K \in \mathbb{R}^{3\times3}$ is the intrinsic matrix of the camera, $R \in \mathbb{R}^{3\times3}$ is the rotation matrix, $t \in \mathbb{R}^{3\times1}$ is the translation vector, and $[x, y, z]^T \in \mathbb{R}^{3\times1}$ corresponds to the 3D coordinates of a LiDAR point in the LiDAR coordinate system.



Projected point clouds Sparse depth map Dense depth map

Figure 3. The process of generating a dense depth map. The sparse LiDAR point cloud is projected onto the image plane using intrinsic and extrinsic matrices to create a sparse depth map, which is then refined through spatial interpolation to produce the final dense depth map.

Finally, a classical depth completion method [39] is applied to densify the projected sparse depth map $D_{\text{img}}^{\text{sparse}}$. The resulting dense depth map $D_{\text{img}}^{\text{gt}}$ is subsequently utilized to supervise the depth estimation process.

3.2.2. Image-Guided LiDAR BEV Feature Construction

For LiDAR point cloud inputs, we employ a LiDAR backbone to generate the initial LiDAR BEV feature $B_{\text{lid}} \in \mathbb{R}^{H\times W\times C}$, which corresponds to the image BEV features B_{img} in the same BEV space. However, relying solely on a simple LiDAR backbone as the encoder poses challenges due to the sparsity of LiDAR point clouds, making it difficult for the model to effectively learn features in sparse regions. To address this limitation, we propose an image-guided LiDAR BEV feature enhancement module for LiDAR feature learning. As shown in Figure 4, this module compresses LiDAR BEV features, interacts with image features via cross-attention, and decodes them into enhanced LiDAR BEV features. Specifically, we first use a BEV encoder comprising several convolutional layers to compress the initial LiDAR BEV feature B_{lid} into a smaller bottleneck feature $B'_{\text{lid}} \in \mathbb{R}^{\frac{H}{2}\times\frac{W}{2}\times C}$:

$$B'_{\text{lid}} = \mathcal{F}_{\text{conv}}(\mathcal{F}_{\text{down}}(\mathcal{F}_{\text{conv}}(B_{\text{lid}}))), \tag{4}$$

where $\mathcal{F}_{\text{conv}}$ denotes a 3×3 convolutional layer function, and $\mathcal{F}_{\text{down}}$ represents a down-sampling layer function. The compressed LiDAR feature B'_{lid} is used as the query Q_{lid}, while the image feature $\mathcal{F}_{\text{img}}$, generated by concatenating the multi-view image features F_{img} along the width dimension, serves as the key K_{img} and value V_{img}. By applying a cross-attention mechanism followed by a convolutional layer, we obtain the aggregated feature A_{lid} as follows:

$$Q = W_{\text{Q}} B'_{\text{lid}}, K = W_{\text{K}} F_{\text{img}}, V = W_{\text{V}} F_{\text{img}}, \tag{5}$$

$$\text{Cross-Atten}(B'_{\text{lid}}, F_{\text{img}}) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_{\text{k}}}}\right)V, \tag{6}$$

$$A_{\text{lid}} = \mathcal{F}_{\text{conv}}(\text{Cross-Atten}(B'_{\text{lid}}, F_{\text{img}})), \tag{7}$$

where W_{Q}, W_{K}, and W_{V} are the weight matrices used for linear transformations to generate the query matrix Q, the key matrix K, and the value matrix V, respectively, and d_k is the dimension of the key vector, used to scale the dot product for stabilizing gradient computation.

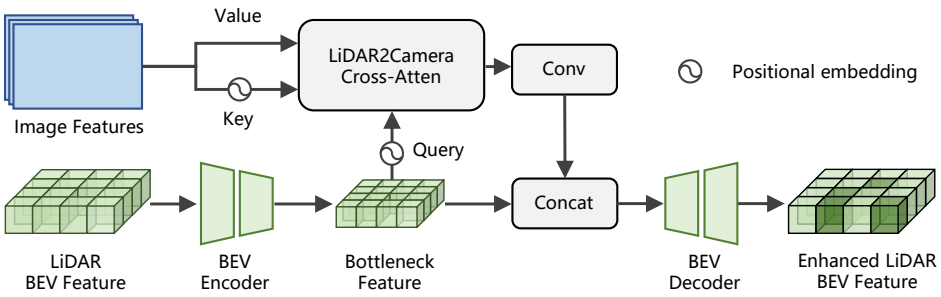


Figure 4. LiDAR BEV enhancement. This module first compresses the initial LiDAR BEV features into a bottleneck representation. It then performs cross-attention between LiDAR BEV queries and image feature key-value pairs. After feature aggregation through attention mechanisms, the module decodes the enhanced representation to generate refined LiDAR BEV features. This process significantly improves feature quality and semantic richness in the BEV space.

Subsequently, we concatenate the aggregated feature A_{lid} with the bottleneck feature B'_{lid} and apply a BEV decoder consisting of several convolutional layers to generate the final LiDAR BEV feature $\bar{B}_{\text{lid}} \in \mathbb{R}^{H \times W \times C}$:

$$\bar{B}_{\text{lid}} = \mathcal{F}_{\text{conv}}(\mathcal{F}_{\text{up}}(\mathcal{F}_{\text{conv}}([A_{\text{lid}}, B'_{\text{lid}}]))), \tag{8}$$

where $[.,.]$ denotes the concatenation operation along the channel dimension, and $\mathcal{F}_{\text{up}}$ represents an upsampling layer function.

3.3. Dual-Attention BEV Feature Fusion

After generating the image and LiDAR BEV features, we directly concatenate them into a single multi-modal BEV representation, as they are generally aligned within the BEV space. However, spatial misalignments may occur due to inaccuracies in depth estimation. To address this issue, we propose a cross-modal BEV fusion module based on dual-attention, as shown in Figure 5a. This module utilizes both the window self-attention block and global self-attention block to compensate for local misalignments and capture global feature dependencies, thereby enhancing the robustness and expressiveness of the fused BEV feature. The process begins with a residual network block (RNB), which fuses the concatenated BEV features and reduces their resolution to $\mathcal{Z} \in \mathbb{R}^{\frac{H}{2} \times \frac{W}{2} \times C}$:

$$\mathcal{Z} = \text{RNB}([B_{\text{img}}, \bar{B}_{\text{lid}}]), \tag{9}$$

where RNB denotes a residual network block. Next, we apply several window self-attention blocks [40], as depicted in Figure 5b. These blocks divide the feature map $\mathcal{Z}$ into four windows $\{z_i \in \mathbb{R}^{\frac{H}{8} \times \frac{W}{8} \times C} \mid i = 1,2,3,4\}$. Self-attention is performed independently within each window, effectively capturing local correlations while significantly reducing computational complexity. After computing the attention for each window, the resulting features are concatenated and reshaped to restore the original feature map dimensions. The computation within the window self-attention block is defined as

$$q = w_{\text{q}} z_i^{n-1}, k = w_{\text{k}} z_i^{n-1}, v = w_{\text{v}} z_i^{n-1}, \tag{10}$$

$$\text{Self-Attent}(z_i^{n-1}) = \text{Softmax}\left(\frac{qk^{\top}}{\sqrt{d_{\text{k}}}}\right) v, \tag{11}$$

$$z_i^n = \text{Self-Atten}(\mathcal{F}_{\text{LN}}(z_i^{n-1})) + z_i^{n-1}, \tag{12}$$

$$\bar{z}_i^n = \mathcal{F}_{\text{MLP}}(\mathcal{F}_{\text{LN}}(z_i^n)) + z_i^n, \tag{13}$$

$$\bar{\mathcal{Z}}^n = \text{Reshape}([\bar{z}_1^n, \bar{z}_2^n, \ldots, \bar{z}_i^n]). \tag{14}$$

where w_q, w_k, and w_v are the weight matrices used for linear transformations to generate the query matrix q, the key matrix k, and the value matrix v, respectively; Self-Attent denotes the self-attention mechanisms; z_i^n and z_i^{n-1} correspond to the output and input features of the i-th window in the n-th window self-attention block, respectively; d_k is the dimension of the key vector k, used to scale the dot product for stabilizing gradient computation; $\mathcal{F}_{\text{LN}}$ represents the layer normalization function, and $\mathcal{F}_{\text{MLP}}$ refers to an MLP layer; and finally, $\bar{\mathcal{Z}}^n$ represents the reconstructed feature map after concatenating and reshaping the outputs of all windows. Following this, we apply another residual network block to further reduce the resolution of the BEV feature to $\frac{H}{4} \times \frac{W}{4}$:

$$\bar{\bar{\mathcal{Z}}}^n = \text{RNB}(\bar{\mathcal{Z}}^n). \tag{15}$$

Subsequently, a global self-attention block, as shown in Figure 5c, is employed to model global feature dependencies. When the BEV features after local fusion are fed into the n-th global self-attention layer, the processing is described by the following formulas:

$$\bar{\bar{\bar{\mathcal{Z}}}}^n = \text{Self-Attent}(\mathcal{F}_{\text{LN}}(\bar{\bar{\mathcal{Z}}}^n)) + \bar{\bar{\mathcal{Z}}}^n, \tag{16}$$

$$\hat{\mathcal{Z}}^n = \mathcal{F}_{\text{MLP}}(\mathcal{F}_{\text{LN}}(\bar{\bar{\bar{\mathcal{Z}}}}^n)) + \bar{\bar{\bar{\mathcal{Z}}}}^n, \tag{17}$$

Finally, the image BEV features and the LiDAR BEV feature are fused to form a unified BEV feature $\hat{\mathcal{Z}}$, which serves as the input to the subsequent BEV-based motion planning decoder.

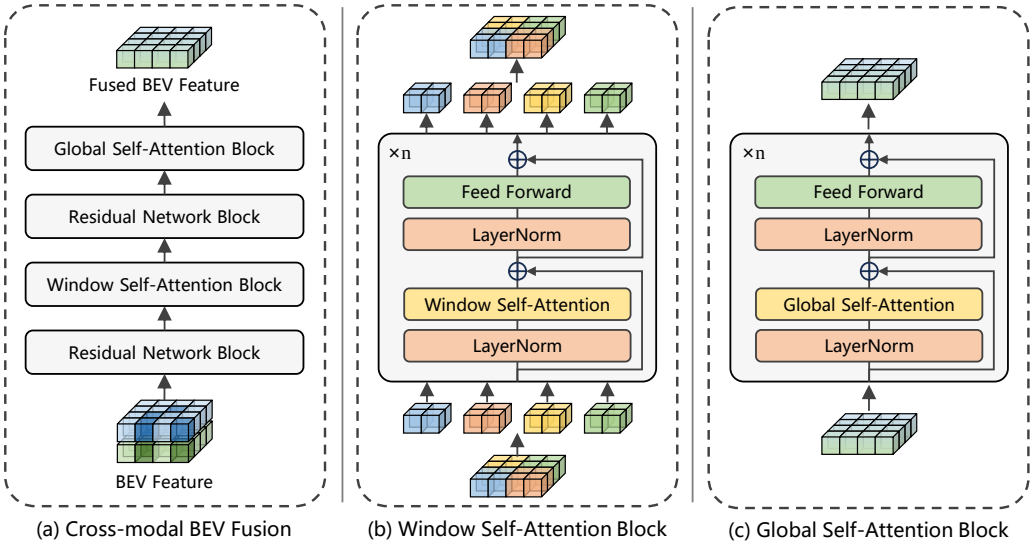


Figure 5. Dual-attention BEV feature fusion. This module integrates image and LiDAR BEV features through dual attention mechanisms: window self-attention captures local spatial correlations, while global self-attention models long-range dependencies.

3.4. Bev-Based Motion Planning

The fused BEV feature contains spatial information that is crucial for motion planning. Previous works [23,26] apply global average pooling (GAP) to convert the feature into a single vector, which is then processed by an MLP + GRU decoder to predict future trajectory points. However, GAP averages all features, leading to the loss of spatial information and the relative positional relationships between features. To address this issue, we propose a BEV-based motion planning decoder. As illustrated in Figure 6, our BEV-based motion planning decoder consists of three key components: joint motion queries, attention-based

motion planning, and an output fusion strategy. The joint motion queries (including trajectory point queries and control command queries) are designed to extract and aggregate information about predicted trajectory points and control commands from the implicit BEV features. The attention-based motion planning network utilizes a self-attention mechanism to capture the intrinsic relationships between trajectory point queries and control command queries. It then interacts with the fused BEV features through a cross-attention mechanism, dynamically updating the joint queries based on the scene context contained within the implicit BEV features. Finally, the network adopts an output fusion strategy to jointly optimize both types of outputs, producing more robust control commands to precisely guide the motion of the autonomous vehicle.

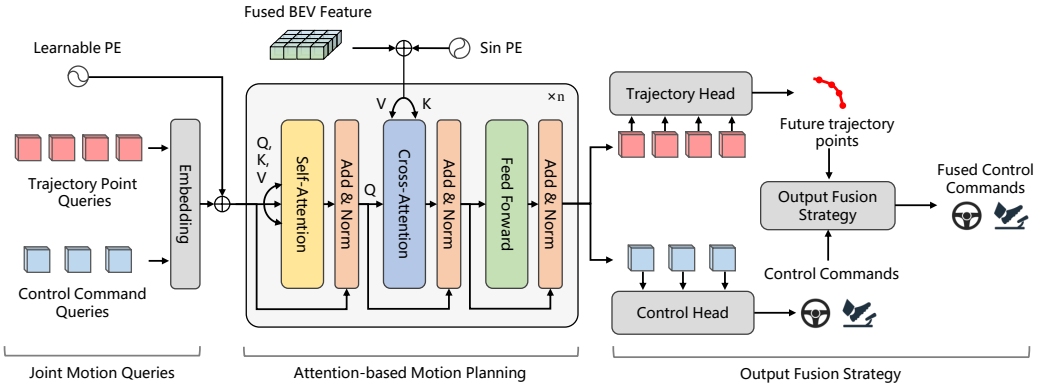


Figure 6. BEV-based motion planning decoder. In the BEV-based motion planning decoder, self-attention connects trajectory point queries with control command queries, while cross-attention enables interactions with fused BEV features to extract contextual information and iteratively refine the queries, enhancing their understanding and adaptability to complex scenes.

Specifically, we first initialize feature vectors using local target points to represent multiple trajectory points, which are then employed as query embeddings for the decoder, as follows:

$$Q_{\text{motion}}^{(0)} = [Q_{\text{traj}}^{(0)}, Q_{\text{ctr}}^{(0)}] + PE_{\text{motion}}, \tag{18}$$

where $Q_{\text{traj}}^{(0)}$ denotes the initial trajectory points queries, $Q_{\text{ctr}}^{(0)}$ denotes the initial control commands queries, and PE_{motion} represents the learnable positional encodings. By incorporating learnable positional encodings, the decoder can effectively distinguish and capture the relationships between different query embeddings.

Then, in the n-th motion planning decoder layer, we first propagate information between trajectory point queries and control command queries through a self-attention mechanism, thereby establishing their intrinsic relationships.

$$\bar{Q}_{\text{motion}}^{n} = \mathcal{F}_{\text{MLP}}(\mathcal{F}_{\text{LN}}(\text{Self-Atten}(Q_{\text{motion}}^{n}))) + Q_{\text{motion}}^{n}, \tag{19}$$

The updated query embeddings $\bar{Q}_{\text{motion}}^{n}$ are utilized to deeply interact with the fused BEV features, leveraging scene contextual information to iteratively refine trajectory point queries and control command queries.

$$Q_{\text{motion}}^{(n+1)} = \mathcal{F}_{\text{MLP}}(\mathcal{F}_{\text{LN}}(\text{Cross-Atten}(\bar{Q}_{\text{motion}}^{(n)}, \hat{\mathcal{Z}}))) + \bar{Q}_{\text{motion}}^{(n)}, \tag{20}$$

After the queries have been processed by multiple layers of the motion planning decoder, we obtain the final trajectory point queries and control command queries. The trajectory point queries are used to generate trajectory points through a GRU-based trajectory head, while the control command queries produce the corresponding control commands

via an MLP-based control head. To convert trajectory points into vehicle control commands (steering, throttle, brake), methods like Transfuser [23] rely on PID controllers. While effective for smooth driving, PID controllers require extensive tuning and may struggle with sharp turns or rapid maneuvers due to model inertia. To overcome this, we propose an output fusion strategy (Algorithm 1) that combines steering commands with trajectory predictions during turns using a weighted coefficient while relying solely on trajectory predictions for straight driving. This strategy reduces PID dependence and enhances control accuracy in dynamic scenarios.

Algorithm 1 : Generating control commands from the output fusion strategy.

Input: sensor perception information i, vehicle status information s, navigation information g, and weight coefficient α;
Output: $steer \in [-1, 1]$, throttle $\in [0, 1]$, brake $\in [0, 1]$;

$\{wp_t\}_{t=1}^{T}, steer_{\text{ctr}}, throttle_{\text{ctr}}, brake_{\text{ctr}} = model(i, s, g)$ // Predicted trajectory points and control command.
$steer_{\text{traj}}, throttle_{\text{traj}}, brake_{\text{traj}} = PIDcontroller(\{wp_t\}_{t=1}^{T})$ // The controller generates control commands based on the predicted trajectory points.
$throttle = throttle_{\text{traj}}, brake = brake_{\text{traj}}$
if $n_{t-1} > 10$ **then**
    $steer = (1 - \alpha) \times steer_{\text{traj}} + \alpha \times steer_{\text{ctr}}$ // If the number of times the historical steering angle exceeds 0.1 is greater than 10, the vehicle is considered to be in a turning state.
else
    $steer = steer_{\text{traj}}$
end if
if $|steer| > 0.1$ **then**
    $n_t = n_{t-1} + 1$
else
    $n_t = 0$
end if

3.5. Loss Function

The overall loss function consists of three components: trajectory point prediction loss $\mathcal{L}_{\text{traj}}$, control command prediction loss $\mathcal{L}_{\text{dep}}$, and depth estimation loss $\mathcal{L}_{\text{ctr}}$. These components are combined as follows:

$$\mathcal{L} = \lambda_{\text{traj}}\mathcal{L}_{\text{traj}} + \lambda_{\text{dep}}\mathcal{L}_{\text{dep}} + \lambda_{\text{ctr}}\mathcal{L}_{\text{ctr}}, \tag{21}$$

where λ_{traj}, λ_{dep}, and λ_{ctr} are hyperparameters used to balance the contributions of the different loss terms. These weights are tuned based on validation errors observed across multiple experiments.

For depth distribution estimation, we employ Binary Cross-Entropy loss [41], defined as

$$\mathcal{L}_{\text{dep}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{D}\left(d_{ij}^{\text{gt}}\log(d_{ij}) + \left(1 - d_{ij}^{\text{gt}}\right)\log(1 - d_{ij})\right), \tag{22}$$

where N is the number of pixels in the depth map, D is the number of discrete depth bins, and $d_{ij}^{\text{gt}} \in D_{\text{img}}^{\text{gt}}$ represents the dense depth values projected by LiDAR.

For trajectory point prediction, we use the ground truth trajectory points w^{gt} provided by the expert to supervise the predicted trajectory points using an L1 loss:

$$\mathcal{L}_{\text{traj}} = \sum_{t=1}^{T}\|w_t - w_t^{\text{gt}}\|_1, \tag{23}$$

where T denotes the number of trajectory points.

For direct control command prediction, an L1 loss similar to the trajectory point prediction is applied. The control loss is defined as

$$\mathcal{L}_{\text{ctr}} = \lambda_{\text{steer}}\mathcal{L}_{\text{steer}} + \lambda_{\text{throttle}}\mathcal{L}_{\text{throttle}} + \lambda_{\text{brake}}\mathcal{L}_{\text{brake}}, \tag{24}$$

where λ_{steer}, $\lambda_{\text{throttle}}$, and λ_{brake} are weighting factors that balance the steering, throttle, and brake losses, respectively.

4. Experiments in CARLA Simulator

This section presents a comprehensive evaluation of BevDrive's closed-loop driving performance in the CARLA simulator, including (1) comparative analysis with state-of-the-art methods, (2) systematic module ablation studies, and (3) qualitative and interpretability analyses.

4.1. Experimental Setup

4.1.1. Implementation Details

We employed ResNet-34 [42] as the backbone for the camera branch and PointPillars [43] as the backbone for the LiDAR branch. Both the image BEV Grid and LiDAR BEV Grid were configured with a range of [0, 32] m × [−16, 16] m (front = 32 m; back = 0 m; left = −16 m; right = 16 m) and a resolution of 0.4 m, resulting in a grid size of 80 × 80. The discrete depth distribution bins were set to a range of [2, 40] m and a spacing of 0.4 m. The model was trained using the AdamW optimizer [44] on four NVIDIA Tesla T4 GPUs for 40 epochs. The learning rate was set to 1×10^{-4} for the first 30 epochs and reduced to 1×10^{-5} for the final 10 epochs. For the loss weights, we used the following settings: $\lambda_{\text{traj}} = 10.0$; $\lambda_{\text{dep}} = 1.0$; $\lambda_{\text{ctr}} = 1.0$; $\lambda_{\text{steer}} = 2.0$; $\lambda_{\text{throttle}} = 1.0$; and $\lambda_{\text{brake}} = 1.0$.

4.1.2. Dataset

We utilized a rule-based expert agent to collect expert driving data across all eight towns and weather conditions in Carla (version 0.9.10). The expert agent operated using privileged information provided by the simulator. Following standard protocols [23,26,45], we collected a total of 253K frames of data at 2 FPS. This dataset included both vehicle state information and sensor data. For sensors, we employed three RGB cameras (facing forward, 60° left, and 60° right), a LiDAR, an IMU, a GPS, and a speedometer. Each camera had a resolution of 300 × 400 pixels and a horizontal field of view (FOV) of 60°. Together, the three cameras provided a combined horizontal FOV of 180° for the ego vehicle.

4.1.3. Benchmark

We evaluated our method through closed-loop autonomous driving experiments in the CARLA simulator. First, we trained our model using expert driving data collected across seven towns (excluding Town05). For evaluation, we employed the widely recognized Town05 Long benchmark, which comprises 10 long routes spanning 1000 to 2000 m, each featuring 10 intersections. These routes were populated with high-density dynamic agents and adversarial scenarios, strategically placed at predefined locations to rigorously test driving performance. This benchmark served as an effective measure of our approach's generalization capabilities in complex and challenging environments.

4.1.4. Metrics

To ensure a fair and rigorous comparison with existing approaches, we adopted a set of standardized metrics that enabled a detailed assessment of diverse driving behaviors. These metrics, calculated per route and averaged across all routes, included route completion

(RC), infraction score (IS), and driving score (DS), with DS as the key metric. Infractions per kilometer provided insight into violations, and infraction counts were converted into scoring rates for intuitive comparisons in adversarial scenarios.

Route Completion (RC): RC measures the percentage of a predefined route completed by an agent. It is calculated as

$$\mathrm{RC} = \frac{1}{K}\sum_{i}^{K} R^i, \tag{25}$$

where R^i denotes the completion ratio for route i, and $K = 10$ is the total number of routes in the Town 05 Long benchmark.

Infraction Score (IS): IS penalizes the agent for committing infractions during a route, such as collisions (with pedestrians, vehicles, or static objects) and traffic violations (e.g., running red lights). It is defined as

$$\mathrm{IS} = \prod_{j}^{\mathrm{Ped},\cdots,\mathrm{Red}} (p^j)^{\mathrm{infractions}^j}, \tag{26}$$

where p^j is the penalty coefficient for infraction type j, and $\mathrm{infractions}^j$ denotes the number of occurrences of infraction j.

Driving Score (DS): DS evaluates overall driving performance by combining route completion and infraction penalties. It is defined as

$$\mathrm{DS} = \frac{1}{K}\sum_{i}^{K} R^i P^i, \tag{27}$$

where P^i represents the infraction multiplier for route i.

4.2. Comparison with State-of-the-Art Methods

We evaluated our BevDrive method against several state-of-the-art approaches on the Town05 Long benchmark in a closed-loop driving scenario, designed to test robustness and generalization due to its complex environments and long routes. Following the standard setup of the Town05 Long benchmark, we performed three evaluations using different random seeds and reported the mean and standard deviation to account for experimental uncertainties. As summarized in Table 1, BevDrive achieves the highest DS and RC among all compared methods, demonstrating its superior performance. Notably, BevDrive achieves these results while relying solely on depth supervision, outperforming methods that utilize additional supervision signals such as semantic segmentation, BEV map segmentation, and object detection. Specifically, BevDrive achieves a DS of 58.1, representing a significant improvement over strong baselines such as Interfuser [25] (51.6) and LAV [30] (46.5). In addition, BevDrive strikes a better balance between route completion and compliance with traffic rules. For methods with an RC exceeding 85%, BevDrive achieves competitive Infraction Score (IS) values, highlighting its ability to handle complex driving scenarios without compromising safety. For instance, while Roach [32] achieves a slightly higher RC of 96.4%, its DS and IS are significantly lower, reflecting inferior driving behavior and weaker adherence to traffic rules. In contrast, BevDrive not only ensures high route completion but also maintains consistent and safe driving behavior, making it a robust and reliable solution for challenging driving tasks.

Table 1. Performance on the Town05 Long benchmark. ↑ indicates that higher values are better, while ⋆ highlights the primary metric. For modalities, C represents camera images, and L denotes LiDAR point clouds. Expert refers to imitation learning based on the driving behavior of a privileged expert. Seg. stands for semantic segmentation, Map. corresponds to BEV map segmentation, Dep. represents depth estimation, and Det. refers to object detection. Optimal values are shown in bold.

Method	Modality	Supervision	DS ⋆ ↑	RC ↑	IS ↑
CILRS [19]	C	Expert	7.8 ± 0.3	10.3 ± 0.0	0.75 ± 0.05
LBC [29]	C	Expert	12.3 ± 2.0	31.9 ± 2.2	0.66 ± 0.02
Roach [32]	C	Expert	41.6 ± 1.8	96.4 ± 2.1	0.43 ± 0.03
TCP [38]	C	Expert	57.2 ± 1.5	80.4 ± 1.5	0.73 ± 0.02
CrossFuser [26]	C + L	Expert	25.8 ± 1.7	71.8 ± 4.3	-
Transfuser [23]	C + L	Expert, Dep., Seg., Map., Det.	31.0 ± 3.6	47.5 ± 5.3	**0.77 ± 0.04**
LAV [30]	C + L	Expert, Seg., Map., Det.	46.5 ± 2.3	69.8 ± 2.3	0.73 ± 0.02
Interfuser [25]	C + L	Expert, Map., Det.	51.6 ± 3.4	88.9 ± 2.5	0.59 ± 0.05
BevDrive (ours)	C + L	Expert, Dep.	**58.1 ± 1.3**	**89.3 ± 2.9**	0.66 ± 0.03

4.3. Ablation Study

4.3.1. Ablation Study of Feature Encoder Modules

The feature encoder consists of two key components: the bidirectionally guided BEV feature construction module and the dual-attention BEV feature fusion module. Specifically, the bidirectionally guided BEV feature construction module is further divided into two sub-modules: depth-guided image BEV feature construction and image-guided LiDAR BEV feature construction. To assess the impact of these components, we performed ablation studies on the DIFC, ILFC, and DF modules within the feature encoder. Here, DIFC refers to the depth-guided image BEV feature construction module, ILFC denotes the image-guided LiDAR BEV feature construction module, and DF represents the dual-attention BEV feature fusion module. As shown in Table 2, each module contributes significantly to performance improvements. Starting with Model 0, which excludes all components, adding DIFC (Model 1) significantly improves performance (DS: 28.5 → 45.2, RC: 56.7 → 75.3), showing the effectiveness of projecting image features into BEV space. Model 2 adds ILFC to DIFC, aligning image and LiDAR features in BEV space, resulting in further gains (DS: 46.5, IS: 0.63). Model 3 evaluates the DF module with DIFC but without ILFC, demonstrating that DF independently enhances fusion (DS: 49.6, RC: 77.3, IS: 0.66) by integrating global and local features. Finally, Model 4 combines DIFC, ILFC, and DF, achieving the best results (DS: 51.8, RC: 78.2, IS: 0.66). These results highlight the importance of DIFC and ILFC for feature representation and the complementary improvements provided by DF for feature fusion and alignment.

We also conducted ablation experiments on the lifting and projection module and depth supervision within the depth-guided image BEV feature construction module. As shown in Table 3, Model 5 uses the same experimental setup as Model 0, serving as the baseline for these ablation studies. Integrating the lifting and projection module into Model 5 results in Model 6, leading to an improvement of 7.8 in DS, 6.1 in RC, and 0.08 in IS. Furthermore, adding depth supervision to Model 6 produces Model 7, which achieves an additional improvement of 8.9 in DS and 12.5 in RC compared to Model 6. These findings demonstrate that both the lifting and projection module and depth supervision play a significant role in enhancing the model's performance.

In addition, we conducted ablation experiments on the dual-attention BEV feature fusion module. As shown in Table 4, Model 8, which does not use any attention mechanisms, serves as the baseline. By incorporating the window self-attention mechanism into Model 8, we obtain Model 9, which achieves an improvement of 2.7 in DS, 1.2 in RC, and 0.04 in IS. Furthermore, adding the global self-attention mechanism to Model 9 results in Model

10, which further improves DS by 2.6 and RC by 2.5. These results demonstrate that the dual-attention mechanism, combining both WSA and GSA, effectively enhances the model's performance.

Table 2. Performance comparison of different modules in feature encoder. DIFC represents the depth-guided image BEV feature construction module, ILFC represents image-guided LiDAR BEV feature construction, and DF represents the dual-attention BEV feature fusion module.

ID	DIFC	ILFC	DF	DS * ↑	RC ↑	IS ↑
0				28.5	56.7	0.56
1	✓			45.2	75.3	0.56
2	✓	✓		46.5	74.5	0.63
3	✓		✓	49.6	77.3	0.66
4	✓	✓	✓	**51.8**	**78.2**	**0.66**

Table 3. Ablation study of the depth-guided image BEV feature construction module. LP represents the lifting and projection module, and Dep represents the depth supervision.

ID	Module	DS * ↑	RC ↑	IS ↑
5	None	28.5	56.7	0.56
6	+ LP	36.3	62.8	**0.64**
7	+ LP + Dep	**45.2**	**75.3**	0.56

Table 4. Ablation study of the dual-attention BEV feature fusion module. WSA represents the window self-attention mechanism and GSA represents the global self-attention mechanism.

ID	Module	DS * ↑	RC ↑	IS ↑
8	None	46.5	74.5	0.63
9	+ WSA	49.2	75.7	**0.67**
10	+ WSA + GSA	**51.8**	**78.2**	0.66

4.3.2. Ablation Study on the Motion Decoder Modules

We evaluated the contributions of the attention-based motion planning network (AMP), joint motion queries (HMQ), and output fusion strategy (OFS) in the motion decoder. As shown in Table 5, the baseline model (Model 11) without these modules performs the worst (DS = 51.8, RC = 78.2, IS = 0.66). Adding AMP (Model 12) improves performance (DS = 55.7, RC = 83.9) by enhancing spatial feature interaction through attention, enabling better handling of complex roads. Including HMQ (Model 13) further boosts performance (DS = 56.2, RC = 85.6, IS = 0.67) by combining global trajectory and local control, improving decision-making in complex environments. Finally, integrating OFS (Model 14) achieves the best results (DS = 58.1, RC = 89.3, IS = 0.66), demonstrating that OFS effectively balances trajectory prediction and control outputs, enhancing robustness and stability. These results validate the synergy of AMP, HMQ, and OFS in improving motion planning.

Table 5. Ablation study of the motion decoder modules. HMQ denotes joint motion queries, AMP denotes the attention-based motion planning network, and OFS represents the output fusion strategy.

ID	AMP	HMQ	OFS	DS * ↑	RC ↑	IS ↑
11				51.8	78.2	0.66
12	✓			55.7	83.9	0.65
13	✓	✓		56.2	85.6	**0.67**
14	✓	✓	✓	**58.1**	**89.3**	0.66

4.3.3. Ablation Study on the Parameters of the Output Fusion Strategy

We analyzed the effect of fusion weights in the output fusion strategy. As shown in Table 6, a weight of 0.2 achieves the best performance (DS = 58.1, RC = 89.3, IS = 0.66), balancing control and trajectory predictions for optimal performance during turns. Higher weights (e.g., 0.5 or 1.0) degrade performance (DS = 55.6, RC = 86.1 for 0.5; DS = 48.5, RC = 81.4 for 1.0), indicating that over-reliance on control commands reduces robustness. Conversely, a weight of 0.0 also underperforms, showing that control commands are necessary for turning scenarios. A small, non-zero weight effectively enhances performance in turns, while trajectory prediction and PID control remain reliable for general driving.

Table 6. Ablation study of the parameters of the output fusion strategy. This table presents the impact of varying fusion weights on driving performance metrics.

ID	Weight	DS * ↑	RC ↑	IS ↑
15	0.0	56.2	85.6	0.**67**
16	0.2	**58.1**	**89.3**	0.66
17	0.5	55.6	86.1	0.66
18	0.8	52.1	84.3	0.65
19	1.0	48.5	81.4	0.64

4.4. Qualitative Results

We conducted a qualitative analysis to demonstrate the effectiveness of BevDrive in typical driving scenarios. In Figure 7a, the ego vehicle approaches an intersection with a red traffic light. BevDrive brings the vehicle to a complete stop and, upon detecting the green light, initiates movement smoothly. Figure 7b shows the collision avoidance capability of BevDrive. When an obstacle (another vehicle) is detected, BevDrive reduces the throttle from 0.30 to 0.00, decreases the vehicle speed from 3.93 m/s to 0.00 m/s, and activates braking to avoid a collision. In Figure 7c, BevDrive performs a left turn at an intersection by reducing the throttle from 0.75 to 0.06, adjusting the steering angle to -0.08, and decelerating to execute the turn safely and smoothly.

4.5. Interpretability Analysis

We visualize BEV feature maps to analyze the interpretability of the decision-making process. In Figure 8a, the ego vehicle encounters a traffic light, stopping for red and resuming movement when the light turns green. The BEV feature maps show clear shifts in focus on the traffic light state, with noticeable changes corresponding to the transition of the light. This highlights the ability of the model to detect traffic light changes and incorporate them into motion planning. Figure 8b illustrates a scenario where a vehicle ahead and to the right makes a left turn. The model brakes to yield and resumes motion once the vehicle clears the path. The BEV feature maps consistently highlight the turning vehicle, demonstrating the ability of the model to perceive dynamic obstacles and integrate this information into its decisions. This effective planning strategy ensures smooth and cautious navigation in complex scenarios.

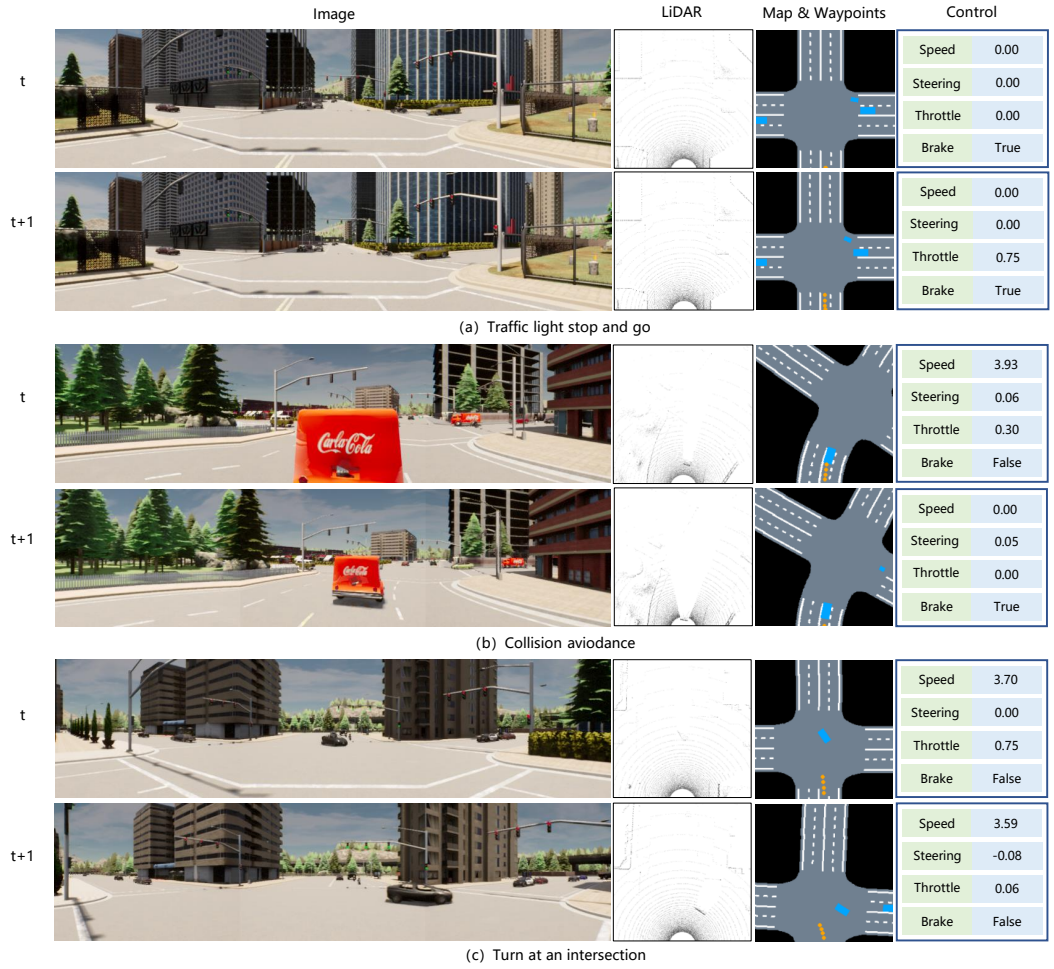


Figure 7. Qualitative results in typical driving scenarios. BevDrive showcases its ability to stop and go at traffic lights, avoid collisions by detecting obstacles and activating brakes, and navigate intersections with precise speed, throttle, and steering adjustments for smooth turns.

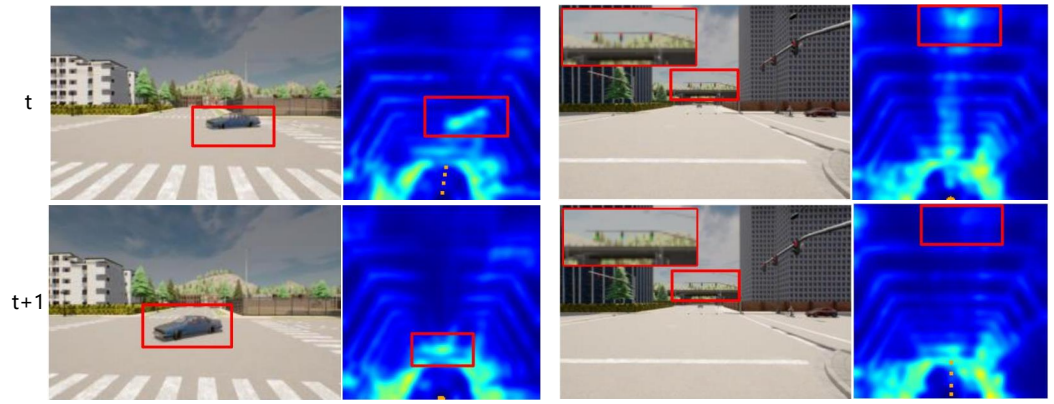


Figure 8. Interpretability analysis of our BevDrive. (**a**) visualizes the BEV feature map focusing on traffic light changes, while (**b**) visualizes the BEV feature map highlighting the model yielding to a turning vehicle and resuming safely.

To improve the interpretability of the BEV-based motion planning decoder, we visualize the attention maps corresponding to the BEV feature regions for each trajectory point query. As illustrated in Figure 9, in a left-turn scenario, the attention for the first trajectory point query is concentrated in the left-front area of the vehicle. This suggests that the model is focusing on detecting potential obstacles in this region that could affect the vehicle's turning behavior. As the trajectory point queries advance to later moments, the model's attention shifts further into the left-front area. This observation highlights that the BEV feature regions associated with the trajectory point queries are closely aligned with areas critical for motion planning, offering valuable insights into how the model's focus influences specific driving decisions.

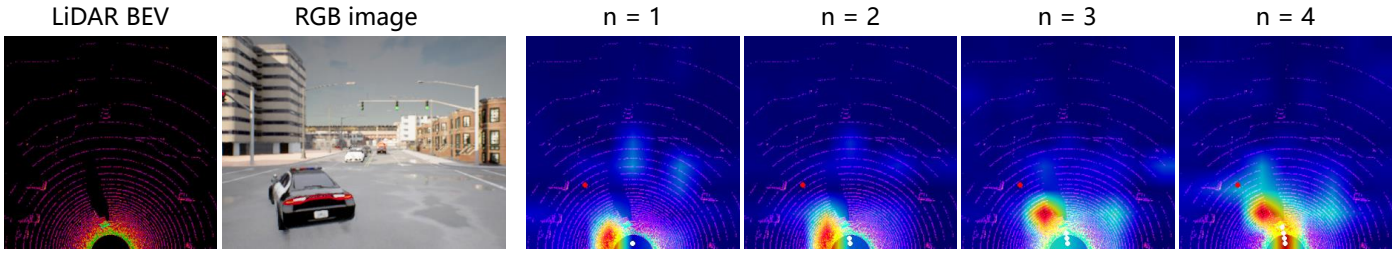


Figure 9. Attention analysis of the BEV-based motion planning decoder: (**a**) sensor inputs of BevDrive; (**b**) attention heatmaps corresponding to different trajectory point queries. In (**b**), the red dot represents the target point, while the white dots represent the predicted points. "*n*" indicates the trajectory point query corresponding to the attention heatmap. In the heatmaps, red indicates higher attention, and blue indicates lower attention.

5. Experiments Using Real Navsim Benchmark

5.1. Dataset and Metrics

We conducted a systematic evaluation of end-to-end autonomous driving algorithms on real-world data using the Navsim benchmark. Navsim utilizes the Openscene [46] dataset for training and testing, which is one of the largest real-world autonomous driving datasets to date. The Navsim end-to-end environment includes 1192 training and validation scenarios, as well as 136 test scenarios, comprising over 100,000 samples collected at a 2 Hz sampling rate.

In the Navsim environment, the 2 Hz, 4 s trajectories were interpolated into 10 Hz, 4 s trajectories using an LQR controller. These trajectories were evaluated based on closed-loop metrics, including No-fault Collisions ($S_{\mathbf{NC}}$), Drivable Area Compliance ($S_{\mathbf{DAC}}$), Time to Collision ($S_{\mathbf{TTC}}$) with bounds, Ego Progress ($S_{\mathbf{SEP}}$), Driving Comfort ($S_{\mathbf{CF}}$), and Driving Direction Compliance ($S_{\mathbf{DDC}}$). The final score was calculated by aggregating these metrics.

$$\text{PDMS} = S_{\text{NC}} \times S_{\text{DAC}} \times S_{\text{TTC}} \times \frac{5 \times S_{\mathbf{EP}} + 5 \times S_{\mathbf{CF}} + 2 \times S_{\mathbf{DDC}}}{12}. \tag{28}$$

5.2. Quantitative Analysis

Table 7 presents a performance comparison of various methods on the NAVSIM benchmark, emphasizing the effectiveness of the proposed BevDrive framework. BevDrive delivers outstanding overall performance, achieving a PDMS score of 83.8, the highest among all methods in the comprehensive evaluation. Notably, it excels in the DAC metric with a score of 92.5, demonstrating its exceptional ability to adhere to drivable area constraints. These results highlight the robustness and adaptability of BevDrive in navigating complex driving scenarios within realistic environments.

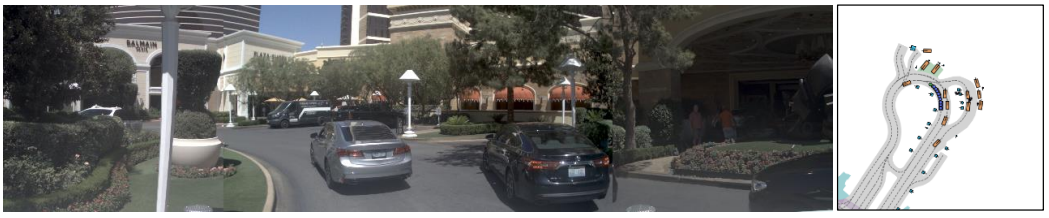
Table 7. Performance comparison of different methods on Navsim benchmark.

Method	NC ↑	DAC ↑	TTC ↑	Comf. ↑	EP ↑	PDMS * ↑
ConsVelo [47]	68.0	57.8	50.0	100	19.4	20.6
EgoStatMLPc [47]	93.0	77.3	83.6	100	62.8	65.6
VADv2 [48]	97.2	89.1	91.6	100	76.0	80.9
DrivingGPT [49]	**98.9**	90.7	**94.9**	95.6	**79.7**	82.4
BevDrive	97.7	**92.5**	92.9	**100**	78.7	**83.8**

5.3. Qualitative Analysis

To validate the effectiveness of our model, we selected several representative driving scenarios for qualitative analysis, including three typical conditions: Turn Left, Turn Right, and Keep Straight. As shown in Figure 10, we visualized the forward-facing camera images and overlaid the generated trajectories on the map for intuitive demonstration. On the map, the blue points represent the trajectories planned by our model. In all scenarios, the vehicle was able to accurately perceive the road structure and generate safe and reasonable trajectories, with the planned results aligning closely with the actual road environments. This demonstrates that our model is capable of adapting to diverse driving scenarios and further highlights its generalization ability in real-world environments.

To further evaluate the performance of our model in challenging driving scenarios, we conducted an experimental analysis in dense traffic and rainy weather conditions, as shown in Figure 11. In the dense traffic scenario, the model was able to dynamically adjust the desired trajectory of the vehicle based on the presence of obstacle vehicles ahead, coming to a stop to avoid collisions and demonstrating collision avoidance capabilities. In the rainy weather scenario, despite the camera images becoming blurred due to rainwater, the model successfully utilized perception data to accurately plan a safe straight-driving trajectory. These experimental results clearly demonstrate the robustness and reliability of our model in handling complex and challenging driving scenarios.

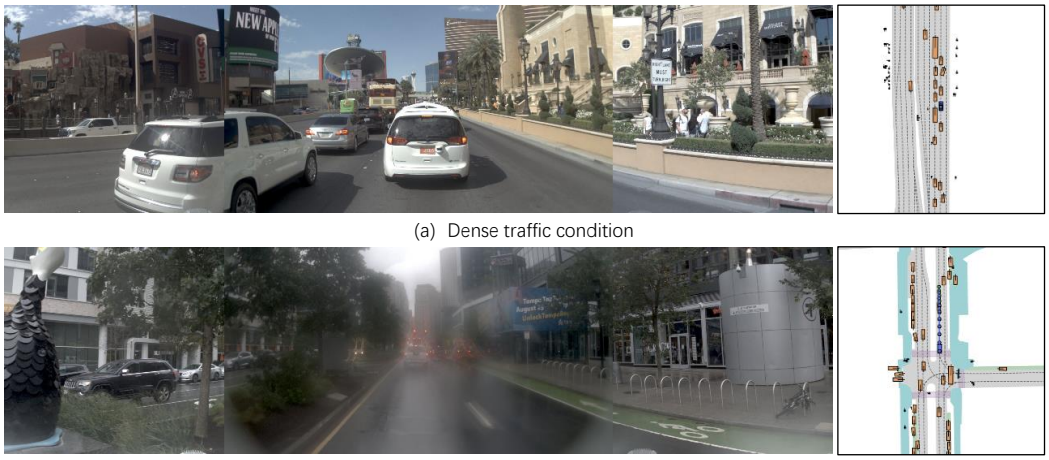


(a) Turn left

(b) Turn right

(c) Keep straight

Figure 10. Performance in representative driving scenarios.



(a) Dense traffic condition

(b) Rainy weather condition

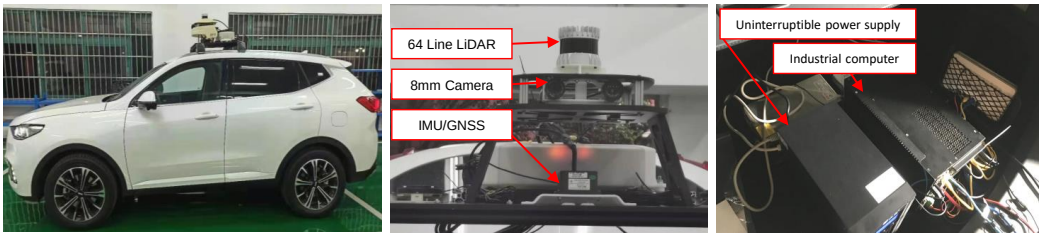
Figure 11. Performance in challenging driving conditions.

6. Experiments using Real Autonomous Vehicle

In this section, we demonstrate the capability of BevDrive to handle complex tasks in real-world driving scenarios by presenting real-world driving experiments conducted using a full-scale autonomous vehicle.

6.1. Implementation Details

The setup of the physical autonomous vehicle is shown in Figure 12. We equipped a full-scale autonomous vehicle with a 64-line LiDAR sensor (10 Hz, FOV: $\pm 22.5^{\circ}$), two video cameras (10 Hz, resolution: $1920 \times 1200 \times 3$, focal length: 8 mm and 25 mm), a GPS/IMU localization module with real-time kinematic (RTK) correction signals, and a powerful industrial computer running a real-time dataset collection program. In this study, we used three kinds of raw data from these devices. Front-view images and LiDAR point clouds were acquired at 10 frames per second to record the environmental conditions of the video cameras and LiDAR sensor on top of the vehicle. In addition, accurate localization and attitude data were recorded by the combined GPS/inertial solution at a frequency of 50 Hz.



(a) Our autonomous vehicle (b) Sensors (c) Industrial computer

Figure 12. Our autonomous vehicle and its equipped devices. (**a**) Our full-scale autonomous vehicle equipped with a LiDAR sensor, cameras, and an integrated navigation module. (**b**) The sensors equipped on our autonomous vehicle. (**c**) The industrial computer and power system equipped on our autonomous vehicle.

The real scenario dataset records urban driving conditions in Guangzhou, China. Our dataset includes scenarios with various weather and illumination conditions. The vehicle was driven manually, and drivers were asked to drive moderately and smoothly. Our final dataset covers a driving distance of 80 km and contains 100,000 frames of driving data. To better understand the distribution of driving scenarios in our dataset, we categorized them into three main types: Turn Right, Turn Left, and Keep Straight. The Keep Straight category was further divided into four subcategories: driving on straight roads, driving on curved

roads, crossing intersections, and slowing down for collision avoidance. As illustrated in Figure 13, the distribution of driving scenarios in our dataset is as follows: Turn Right constitutes 15.8%, Turn Left accounts for 17.0%, and Keep Straight makes up 67.2%. Within the Keep Straight category, the data are further broken down into driving on straight roads (52.5%), driving on curved roads (25.7%), crossing intersections (11.7%), and slowing down for collision avoidance (10.1%).

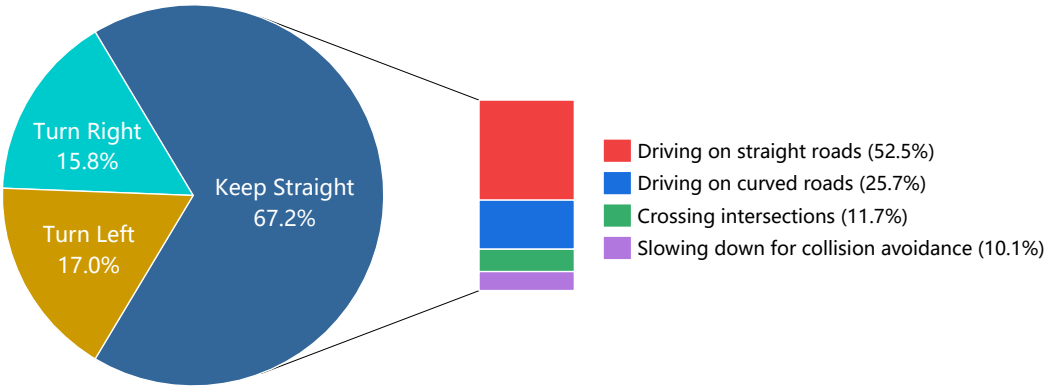


Figure 13. Statistics of our dataset. We categorized the driving scenarios in our dataset into three main types: Turn Right, Turn Left, and Keep Straight. For the Keep Straight category, we further divided the data into four subcategories: driving on straight roads, driving on curved roads, crossing intersections, and slowing down for collision avoidance.

6.2. Quantitative Analysis

The results in Table 8 demonstrate that the proposed BevDrive consistently achieves lower L2 errors across all time horizons (0.5 s, 1 s, 1.5 s, and 2 s) compared to the baseline methods, Late Fusion and Transfuser. Specifically, BevDrive achieves a significantly lower average L2 error of 0.08 m, outperforming Transfuser (0.12 m) and Late Fusion (0.16 m). This highlights the superior decision-making and trajectory planning capabilities of the proposed method, which remain reliable even in real-world scenarios. The reduced short-term and long-term L2 errors validate the robustness and accuracy of the algorithm in practical autonomous driving environments.

Table 8. L2 error comparison across methods at different time horizons.

Method	L2 Error (m)				
	0.5 s	**1 s**	**1.5 s**	**2 s**	**Avg.**
Late Fusion	0.06	0.09	0.16	0.31	0.16
Transfuser	0.05	0.07	0.11	0.23	0.12
BevDrive	**0.04**	**0.05**	**0.08**	**0.13**	**0.08**

6.3. Qualitative Analysis

To show the performance of our proposed method in real driving scenarios, we visualize the driving trajectories and the corresponding motion states of the ego vehicle in several typical scenarios, including keeping straight, turning right, turning left, following the lane, stopping at a red light, and going at a green light, as shown in Figure 14. In general, we can see that our BevDrive is able to accurately generate collision-free trajectories under various driving conditions. As shown in Figure 14a, the vehicle can run smoothly with a high driving speed in the straight driving scenario. As shown in Figure 14b, the planned trajectories in the turning scenarios are reliable and smooth. As shown in Figure 14c, our model reasonably slows down as a human driver would when following a vehicle. Moreover, our model can stop at an intersection when the traffic light is red, as shown in

Figure 14d, and start up again after the traffic light turns green, as shown in Figure 14e. These results demonstrate that our method is capable of controlling the vehicle to drive stably in real-world driving scenarios.

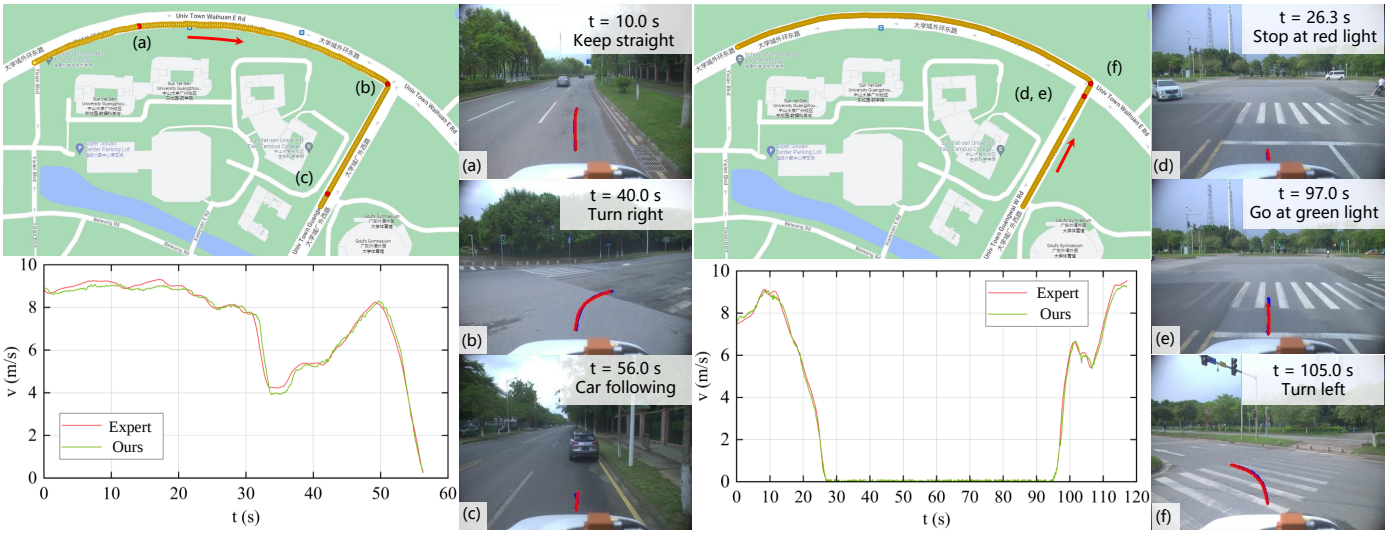


Figure 14. Visualization of trajectories in typical driving scenarios. BevDrive demonstrates its ability to generate collision-free trajectories in a variety of driving scenarios, including lane following (a), turning (b, f), collision avoidance (c) and traffic light reaction (d, e).

7. Discussion

Although BevDrive demonstrates outstanding performance in perception, sensor fusion, and motion planning, there are certain limitations that should be addressed in future work. The current framework relies solely on single-frame sensor data for decision-making, which restricts its ability to capture and model dynamic changes in the driving environment. This reliance on static, frame-by-frame processing may limit its effectiveness in highly dynamic scenarios, where understanding the temporal motion patterns of surrounding agents is essential for precise trajectory prediction and efficient obstacle avoidance. To address this limitation, future work will investigate architectures capable of long-term temporal modeling, including Temporal Transformers and Long Short-Term Memory (LSTM) networks. By integrating historical frame data (e.g., past trajectories, images, and LiDAR information), the system can more accurately predict the behaviors and intentions of surrounding vehicles and pedestrians. Leveraging this temporal information for motion planning can significantly enhance the obstacle avoidance capabilities of end-to-end autonomous driving models, ultimately improving their robustness and adaptability in real-world dynamic driving environments.

In addition, BevDrive currently relies on LiDAR geometric measurement data during the BEV construction process. The supervision of depth information is primarily based on 3D measurements from LiDAR, which may limit the adaptability of the system to a broader range of scenarios. To address this issue and further enhance the robustness and flexibility of the system, we will also explore BEV construction methods that do not rely on LiDAR supervision, thereby reducing the dependency on 3D measurements from LiDAR.

8. Conclusions

In this work, we proposed BevDrive, a novel end-to-end autonomous driving framework that unifies camera–LiDAR fusion and motion planning within a BEV representation. The framework integrates three key modules: the bidirectionally guided BEV feature construction module, the dual-attention BEV feature fusion module, and the BEV-based

motion planning decoder. The bidirectionally guided BEV feature construction module leverages a dual-stream network to process image and LiDAR data independently, using depth-guided image BEV construction and image-guided LiDAR BEV construction to transform cross-view features into a unified BEV space, effectively eliminating spatial discrepancies between modalities. The dual-attention BEV feature fusion module combines local and global features using hybrid attention mechanisms to achieve comprehensive multi-modal integration. The BEV-based motion planning decoder consists of joint motion queries, attention-based feature interaction, and an output fusion strategy, enabling precise and reliable vehicle control by dynamically refining motion plans and integrating trajectory and control commands. Extensive evaluations demonstrate that BevDrive achieves accurate and collision-free motion planning, highlighting its robustness and potential for real-world autonomous driving applications.

Author Contributions: Conceptualization, Z.Y. and X.T.; methodology, Z.Y. and Y.L.; software, Z.Y. and Y.L.; validation, J.L. and Y.W.; formal analysis, Z.Y. and Y.W.; investigation, Y.L. and Y.W.; resources, X.T.; data curation, Z.Y.; writing—original draft preparation, Z.Y. and Y.W.; writing—review and editing, Z.Y. and Y.W.; visualization, Z.Y.; supervision, X.T.; project administration, X.T.; funding acquisition, X.T. All authors have read and agreed to the published version of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: The research is supported by the Shenzhen Science and Technology Program under Grant KJZD20231023100204008.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Conflicts of Interest: Author Mr. Yuandong Lyu was employed by the company Li Auto Inc. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Han, L.; Wang, S.; Wang, Z.; Jin, L.; He, X. Method of 3D voxel prescription map construction in digital orchard management based on LiDAR-RTK boarded on a UGV. *Drones* **2023**, *7*, 242. [CrossRef]
2. Zhang, X.; Fan, Z.; Shen, Y.; Li, Y.; An, Y.; Tan, X. MAEMOT: Pretrained MAE-Based Antiocclusion 3-D Multiobject Tracking for Autonomous Driving. *IEEE Trans. Neural Netw. Learn. Syst.* **2024**, *Early Access*.
3. Zhu, M.; Gong, Y.; Tian, C.; Zhu, Z. A Systematic Survey of Transformer-Based 3D Object Detection for Autonomous Driving: Methods, Challenges and Trends. *Drones* **2024**, *8*, 412. [CrossRef]
4. Shi, L.; Wang, L.; Zhou, S.; Hua, G. Trajectory unified transformer for pedestrian trajectory prediction. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 2–3 October 2023; pp. 9675–9684.
5. Yang, B.; Fan, F.; Ni, R.; Wang, H.; Jafaripournimchahi, A.; Hu, H. A multi-task learning network with a collision-aware graph transformer for traffic-agents trajectory prediction. *IEEE Trans. Intell. Transp. Syst.* **2024**, *25*, 6677–6690.
6. Gui, X.; Huang, T.; Shao, H.; Yao, H.; Zhang, C. Fiptr: A simple yet effective transformer framework for future instance prediction in autonomous driving. In Proceedings of the European Conference on Computer Vision, Milan, Italy, 29 September–4 October 2024; pp. 19–35.
7. Li, X.; Gong, X.; Chen, Y.H.; Huang, J.; Zhong, Z. Integrated path planning-control design for autonomous vehicles in intelligent transportation systems: A neural-activation approach. *IEEE Trans. Intell. Transp. Syst.* **2024**, *25*, 7602–7618. [CrossRef]
8. Xie, G.; Fang, L.; Su, X.; Guo, D.; Qi, Z.; Li, Y.; Che, J. Research on Risk Avoidance Path Planning for Unmanned Vehicle Based on Genetic Algorithm and Bezier Curve. *Drones* **2025**, *9*, 126. [CrossRef]
9. Eskandari, M.; Savkin, A.V.; Deghat, M. Kinodynamic Model-Based UAV Trajectory Optimization for Wireless Communication Support of Internet of Vehicles in Smart Cities. *Drones* **2024**, *8*, 574. [CrossRef]
10. Daoud, M.A.; Mehrez, M.W.; Rayside, D.; Melek, W.W. Simultaneous feasible local planning and path-following control for autonomous driving. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 16358–16370.
11. Lee, S.; Hwang, S.; Kim, H.S. T–S Fuzzy Observer-based Output Feedback Lateral Control of UGVs Using a Disturbance Observer. *Drones* **2024**, *8*, 685. [CrossRef]

12. Kim, J.S.; Chung, C.C. Optimal Sliding Hyperplane Design of Discrete-Time Integral Sliding-Mode Control for Automated Driving Vehicles. *IEEE Trans. Ind. Inform.* 2025, *Early Access*. [CrossRef]
13. Chen, L.; Wu, P.; Chitta, K.; Jaeger, B.; Geiger, A.; Li, H. End-to-end autonomous driving: Challenges and frontiers. *IEEE Trans. Pattern Anal. Mach. Intell.* 2024, *Early Access*.
14. Codevilla, F.; Müller, M.; López, A.; Koltun, V.; Dosovitskiy, A. End-to-end driving via conditional imitation learning. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 4693–4700.
15. Prakash, A.; Chitta, K.; Geiger, A. Multi-modal fusion transformer for end-to-end autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 7077–7087.
16. Xiao, Y.; Codevilla, F.; Gurram, A.; Urfalioglu, O.; López, A.M. Multimodal end-to-end autonomous driving. *IEEE Trans. Intell. Transp. Syst.* **2020**, *23*, 537–547.
17. Lyu, Y.; Tan, X.; Yu, Z.; Fan, Z. Sensor Fusion and Motion Planning with Unified Bird's-Eye View Representation for End-to-end Autonomous Driving. In Proceedings of the 2024 International Joint Conference on Neural Networks (IJCNN), Yokohama, Japan, 30 June–5 July 2024; pp. 1–8.
18. Hecker, S.; Dai, D.; Van Gool, L. End-to-end learning of driving models with surround-view cameras and route planners. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 435–453.
19. Codevilla, F.; Santana, E.; López, A.M.; Gaidon, A. Exploring the limitations of behavior cloning for autonomous driving. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic Korea, 27 October–2 November 2019; pp. 9329–9338.
20. Zhang, Q.; Tang, M.; Geng, R.; Chen, F.; Xin, R.; Wang, L. Mmfn: Multi-modal-fusion-net for end-to-end driving. In Proceedings of the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2022), Kyoto, Japan, 23–27 October 2022; pp. 8638–8643.
21. Cai, P.; Wang, S.; Sun, Y.; Liu, M. Probabilistic end-to-end vehicle navigation in complex dynamic environments with multimodal sensor fusion. *IEEE Robot. Autom. Lett.* **2020**, *5*, 4218–4224. [CrossRef]
22. Huang, Z.; Lv, C.; Xing, Y.; Wu, J. Multi-modal sensor fusion-based deep neural network for end-to-end autonomous driving with scene understanding. *IEEE Sens. J.* **2020**, *21*, 11781–11790.
23. Chitta, K.; Prakash, A.; Jaeger, B.; Yu, Z.; Renz, K.; Geiger, A. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *45*, 12878–12895.
24. Jaeger, B.; Chitta, K.; Geiger, A. Hidden biases of end-to-end driving models. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 2–3 October 2023; pp. 8240–8249.
25. Shao, H.; Wang, L.; Chen, R.; Li, H.; Liu, Y. Safety-enhanced autonomous driving using interpretable sensor fusion transformer. In Proceedings of The 6th Conference on Robot Learning, Auckland, New Zealand, 14–18 December 2023; pp. 726–737.
26. Wu, W.; Deng, X.; Jiang, P.; Wan, S.; Guo, Y. Crossfuser: Multi-modal feature fusion for end-to-end autonomous driving under unseen weather conditions. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 14378–14392.
27. Hussein, A.; Gaber, M.M.; Elyan, E.; Jayne, C. Imitation learning: A survey of learning methods. *ACM Comput. Surv.* **2017**, *50*, 1–35. [CrossRef]
28. Kaelbling, L.P.; Littman, M.L.; Moore, A.W. Reinforcement learning: A survey. *J. Artif. Intell. Res.* **1996**, *4*, 237–285.
29. Chen, D.; Zhou, B.; Koltun, V.; Krähenbühl, P. Learning by cheating. In Proceedings of the 2020 Conference on Robot Learning, Virtual, 16–18 November 2020; pp. 66–75.
30. Chen, D.; Krähenbühl, P. Learning from all vehicles. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 17222–17231.
31. Jaritz, M.; De Charette, R.; Toromanoff, M.; Perot, E.; Nashashibi, F. End-to-end race driving with deep reinforcement learning. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 2070–2075.
32. Zhang, Z.; Liniger, A.; Dai, D.; Yu, F.; Van Gool, L. End-to-end urban driving by imitating a reinforcement learning coach. In Proceedings of the IEEE International Conference on Computer Vision, Virtual, 11–17 October 2021; pp. 15222–15232.
33. Liang, X.; Wang, T.; Yang, L.; Xing, E. Cirl: Controllable imitative reinforcement learning for vision-based self-driving. In Proceedings of the European Conference Computer Vision, Munich, Germany, 8–14 September 2018; pp. 584–599.
34. Hu, J.; Kong, H.; Zhang, Q.; Liu, R. Enhancing scene understanding based on deep learning for end-to-end autonomous driving. *Eng. Appl. Artif. Intell.* **2022**, *116*, 105474.
35. Chitta, K.; Prakash, A.; Geiger, A. Neat: Neural attention fields for end-to-end autonomous driving. In Proceedings of the IEEE International Conference on Computer Vision, Virtual, 11–17 October 2021; pp. 15793–15803.
36. Vora, S.; Lang, A.H.; Helou, B.; Beijbom, O. Pointpainting: Sequential fusion for 3d object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 4604–4612.

37. Pan, Y.; Cheng, C.A.; Saigol, K.; Lee, K.; Yan, X.; Theodorou, E.; Boots, B. Agile Autonomous Driving using End-to-End Deep Imitation Learning. In Proceedings of the Robotics: Science and System, Pittsburgh, PA, USA, 26–30 June 2018.
38. Wu, P.; Jia, X.; Chen, L.; Yan, J.; Li, H.; Qiao, Y. Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline. *Proc. Adv. Neural Inf. Process. Syst.* **2022**, *35*, 6119–6132.
39. Ku, J.; Harakeh, A.; Waslander, S.L. In defense of classical image processing: Fast depth completion on the cpu. In Proceedings of the Conference Computer Vision, Salt Lake City, UT, USA, 18–22 June 2018; pp. 16–22.
40. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In Proceedings of the IEEE International Conference on Computer Vision, Virtual, 11–17 October 2021; pp. 10012–10022.
41. Ruby, U.; Yendapalli, V. Binary cross entropy with deep learning technique for image classification. *Int. J. Adv. Trends Comput. Sci. Eng.* **2020**, *9*, 5393–5397.
42. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
43. Lang, A.H.; Vora, S.; Caesar, H.; Zhou, L.; Yang, J.; Beijbom, O. Pointpillars: Fast encoders for object detection from point clouds. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2019; pp. 12697–12705.
44. Loshchilov, I.; Hutter, F. Decoupled Weight Decay Regularization. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
45. Chen, W.; Chen, Y.; Kong, F.; Zhang, X.; Sun, H. Motion planning using feasible and smooth tree for autonomous driving. *IEEE Trans. Veh. Technol.* **2023**, *73*, 6270–6282. [CrossRef]
46. Contributors, O. Openscene: The largest up-to-date 3d occupancy prediction benchmark in autonomous driving. In Proceedings of the Conference on Computer Vision and Pattern Recognition, Vancouver, Canada, 18–22 June 2023.
47. Dauner, D.; Hallgarten, M.; Li, T.; Weng, X.; Huang, Z.; Yang, Z.; Li, H.; Gilitschenski, I.; Ivanovic, B.; Pavone, M.; et al. Navsim: Data-driven non-reactive autonomous vehicle simulation and benchmarking. *Adv. Neural Inf. Process. Syst.* **2024**, *37*, 28706–28719.
48. Chen, S.; Jiang, B.; Gao, H.; Liao, B.; Xu, Q.; Zhang, Q.; Huang, C.; Liu, W.; Wang, X. Vadv2: End-to-end vectorized autonomous driving via probabilistic planning. *arXiv* **2024**, arXiv:2402.13243.
49. Chen, Y.; Wang, Y.; Zhang, Z. Drivinggpt: Unifying driving world modeling and planning with multi-modal autoregressive transformers. *arXiv* **2024**, arXiv:2412.18607.