UC Berkeley
UC Berkeley Electronic Theses and Dissertations

Title

Perception, Planning, and World Modeling: Learning Scalable World Representations for Autonomous Driving

Permalink

https://escholarship.org/uc/item/8g03z0kb

ISBN

9798189271656

Author

Xie, Yichen

Publication Date

2026-05-01

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

Perception, Planning, and World Modeling: Learning Scalable World Representations for Autonomous Driving

By

Yichen Xie

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering Science - Mechanical Engineering

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Masayoshi Tomizuka, Chair
Professor Roberto Horowitz
Professor Jonathan Shewchuk

Spring 2026

Perception, Planning, and World Modeling:
Learning Scalable World Representations for Autonomous Driving

Copyright 2026
by
Yichen Xie

Abstract

Perception, Planning, and World Modeling:
Learning Scalable World Representations for Autonomous Driving

by

Yichen Xie

Doctor of Philosophy in Engineering Science - Mechanical Engineering

University of California, Berkeley

Professor Masayoshi Tomizuka, Chair

Autonomous driving requires intelligent systems that can understand the current environment, make decisions from raw sensory observations, and anticipate how the world may evolve over time. Traditional autonomous driving systems typically address these capabilities through a hierarchical pipeline, where perception, planning, and world modeling are developed as largely separate components. While such decomposition offers interpretability and engineering flexibility, it often suffers from information mismatch, error accumulation, and limited scalability. Recent advances in learning-based methods have increasingly blurred the boundaries between these modules, motivating a more unified perspective centered on scalable world representations.

This dissertation studies autonomous driving through this lens. The central question of the thesis is how to represent the 3D driving environment in a way that supports scene understanding, decision-making, and future world modeling. Rather than proposing a single monolithic system, the dissertation approaches this question through a sequence of methods that together form a coherent progression from perception to planning to world modeling.

First, Chapter 2 investigates efficient multi-sensor perception and introduces SparseFusion, a 3D object detection framework that performs LiDAR-camera fusion directly on sparse instance-level representations. Next, Chapter 3 studies unsupervised representation learning from raw driving logs and proposes Cohere3D, which leverages long-term temporal correspondence to learn coherent instance representations transferable across perception, mapping, prediction, and planning tasks. Chapter 4 addresses end-to-end motion planning through S4-Driver, a scalable self-supervised framework that directly predicts future trajectories from multi-view camera inputs using a sparse spatio-temporal volume representation. The dissertation then turns to world modeling. Chapter 5 proposes X-Drive, a multi-sensor generative framework for jointly synthesizing LiDAR point clouds and multi-view images with cross-modality consistency. Finally, Chapter 6 presents RAYNOVA, a geometry-agnostic world

model for multi-view video generation that captures future scene evolution through unified 4D spatio-temporal reasoning in continuous ray space.

Taken together, these contributions support a common thesis that progress in autonomous driving benefits better ways of representing the world. Across perception, planning, and world modeling, this dissertation shows that efficient, structured, and scalable representations can significantly improve both performance and generality in autonomous driving systems.

To those who strive to make the world better.

Contents

Contents **ii**

List of Figures **v**

List of Tables **ix**

1 Introduction **1**
1.1 Background and Motivations 1
1.2 Dissertation Outline and Contributions 4

I Perception: Understanding the Current World **6**

2 SparseFusion: Fusing Multi-Modal Sparse Representations for Multi-Sensor 3D Object Detection **7**
2.1 Introduction 7
2.2 Related Work 10
2.3 Methodology 12
2.4 Experiments 15
2.5 Conclusion 22

II Planning: Learning Actionable Representations **23**

3 Cohere3D: Exploiting Temporal Coherence for Unsupervised Representation Learning of Vision-based Autonomous Driving **24**
3.1 Introduction 24
3.2 Related Works 26
3.3 Methodology 27
3.4 Experiments 32
3.5 Conclusions 37

4 S4-Driver: Scalable Self-Supervised Driving Multimodal Large Language Model with Spatio-Temporal Visual Representation 38
4.1 Introduction . . . 38
4.2 Related Works . . . 40
4.3 Waymo Open Motion Dataset for Planning . . . 41
4.4 Enhancing MLLM for Motion Planning . . . 43
4.5 Experiments . . . 49
4.6 Conclusion and Future Work . . . 53

IIIWorld Modeling: Imagining the Future Evolution 54

5 X-Drive: Cross-Modality Consistent Multi-Sensor Data Synthesis for Driving Scenarios 55
5.1 Introduction . . . 55
5.2 Related Works . . . 57
5.3 Preliminary . . . 58
5.4 Methodology . . . 60
5.5 Experiments . . . 65
5.6 Conclusions . . . 69

6 RAYNOVA: Geometry-Free Auto-Regressive 4D World Modeling with Unified Spatio-Temporal Representation 70
6.1 Introduction . . . 70
6.2 Related Works . . . 73
6.3 Methodology . . . 74
6.4 Experiments . . . 80
6.5 Conclusion . . . 85

7 Final Words 86
7.1 Summary of Contributions . . . 86
7.2 Future Work . . . 86
7.3 Concluding Remarks . . . 87

Bibliography 88

A Appendix for Chapter 2 "SparseFusion: Fusing Multi-Modal Sparse Representations for Multi-Sensor 3D Object Detection" 111
A.1 Additional Experiments . . . 111
A.2 Architecture Details . . . 114

B Appendix for Chapter 4 "S4-Driver: Scalable Self-Supervised Driving Multimodal Large Language Model with Spatio-Temporal Visual Representation" 119
B.1 Implementation Details 119
B.2 Behavior-wise Model Performance 123
B.3 Additional Qualitative Results 123
B.4 Additional Ablation Studies 124

C Appendix for Chapter 5 "X-Drive: Cross-Modality Consistent Multi-Sensor Data Synthesis for Driving Scenarios" 127
C.1 Ablation Study of Classifier-Free Guidance Scales 127
C.2 Implementation Details 127
C.3 Limitation and Future Works 128
C.4 Qualitative Results 129

D Appendix for Chapter 6 "RAYNOVA: Geometry-Free Auto-Regressive 4D World Modeling with Unified Spatio-Temporal Representation" 131
D.1 Implementation Details 131
D.2 Additional Ablation Study 132
D.3 Additional Qualitative Results 133
D.4 Limitation and Future Work 133

List of Figures

1.1 Learning scalable world representations from raw sensor data instead of relying on strictly separated hierarchical stages. 2
1.2 Scalable driving data. Human driving logs with raw sensor data are scalable because they share similar embodiment, environment structure, and action space. 3

2.1 Compared to existing fusion algorithms, SparseFusion achieves state-of-the-art performance as well as the fastest inference speed on nuScenes *test set*. † : Official code of AutoAlignV2 uses flip as test-time augmentation. ‡ : We use BEVFusion-base results in the official repository of BEVFusion to match the input resolutions of other methods. § : Swin-T is adopted as image backbone. 8
2.2 Overall framework of SparseFusion. It fuses sparse candidates from LiDAR and camera modalities to obtain a multi-modality instance-level representation in the unified LiDAR space. 12
2.3 Cross-modality information transfer. We transfer geometric information from LiDAR to camera modality, and semantic information from camera to LiDAR modality. 14
2.4 Qualitative results of SparseFusion on nuScenes *validation set*. 16
2.5 Percentage of objects recalled by each modality. We do not count those objects recalled by both or neither. 19
2.6 Instance-level feature interaction in the fusion stage. Orange boxes are detected after the fusion stage with high confidence in the BEV space. Blue and green dots denote all instances from the LiDAR and camera branches separately. Orange boxes are connected with blue/green dots with blue/green lines. The attention strength is represented by the darkness and thickness of the lines. 20

3.1 Temporal coherence of instance representations. Our algorithm encourages instance representations in the BEV space to be coherent across time and viewpoints. 25
3.2 Overview of Cohere3D. We construct long-term instance-level temporal correspondence via unlabeled LiDAR point clouds, and design a contrastive learning framework to encourage the long-term coherence of instance representations. . . 27
3.3 LiDAR-guided temporal correspondence construction. Instances are identified from point clouds within each frame via clustering. The Hungarian algorithm connects every two adjacent frames to generate long-term correspondence. . . . 29

3.4 Qualitative comparison of SOLOFusion perception model without (left) vs with Cohere3D pretraining (right). 35
3.5 Qualitative comparison of VAD planning model without (left) vs with Cohere3D pretraining (right). 35
3.6 LiDAR-guided long-term correspondence. It reliably tracks each instance over long-term historical frames even if it is captured by different cameras at each time step. 36

4.1 S4-Driver scaling up with abundant unlabeled driving logs (4.1a). Different from past efforts that improve MLLM planning via multi-task learning and CoT reasoning (4.1b), S4-Driver focuses on self-supervised motion planning (4.1c) without human annotations. While a vanilla PaLI approach with self-supervised training achieves good performance, the full S4-Driver with spatio-temporal visual representation achieves the best performance and scalability (4.1a). 39
4.2 High-level behavior commands for motion planning. 42
4.3 Statistical distributions of WOMD-Planning-ADE. We show the distributions of ego-vehicle trajectories, velocities, accelerations and each behavior scenario sizes in the validation set. 42
4.4 Overview of our proposed S4-Driver algorithm. We enhance the PaLI model for motion planning by incorporating meta-decision, spatio-temporal visual representation, and multi-decoding aggregation. 43
4.5 A roadmap for enhancing MLLM for planning. The techniques are adopted step-by-step from top to bottom, while shadow items are not adopted in the subsequent steps. We show the performance on WOMD-Planning-ADE (Sec. 4.3) after including each module. 44
4.6 Qualitative results of motion planning. We show the *front left, front, front right* cameras for each case. 51
4.7 Meta-decision prediction accuracy for each behavior. 52
4.8 Distribution of sparse volumes along x- and y-axes. 52

5.1 X-Drive simultaneously generates high-quality multi-view images and point clouds with cross-modality consistency, which is impossible for previous single-modality generative models. 56
5.2 Overview of our proposed X-Drive framework. We design a dual-branch diffusion model architecture to generate multi-modality data. Cross-modality epipolar condition modules (Fig. 5.3) are inserted between branches to enhance the cross-modality consistency. 59
5.3 Cross-modality epipolar condition module. We perform mutual conditions locally between LiDAR and camera modalities based on epipolar lines on multi-view image and range image latents. 63

5.4 Cross-modality consistency qualitative results for multi-modality generation and conditional cross-modality generation. Colors of point clouds refer to different depths. Well-matched regions between point clouds and multi-view images are highlighted with red circles. . . . 67
5.5 Key points correspondence between adjacent synthetic camera images. Our proposed method (bottom) can bring higher multi-view consistency than the baseline (top). . . . 68
5.6 Scene-level and object-level controllable generation by changing the input conditions. . . . 69
5.7 Effect of CFG scales. . . . 69

6.1 Demonstrations of RayNova as Versatile World Foundation Model. . . . 71
6.2 Overview of RayNova Framework. RayNova is composed of dual-casual (scale and time) blocks. The local scale attention and local cross attention works on each image indepedently, while the global causal attention works across multi-view and multiframe images enhanced with a unified ray-level relative position embedding for better spatio-temporal consistency. . . . 74
6.3 Dual-Causality for Multi-View Video Generation. Green arrows represent the causal dependency, while the darkness indicates the topological order of autoregression (from light to dark). . . . 75
6.4 Ablation Study on Scale Causality. Conditioning on all scales in history hurts the modeling of dynamics, while conditioning only on same scale is insufficient for temporal coherence. . . . 83
6.5 Ablation Study on Model Size. Large scale model can bring significantly better visual quality. . . . 84

A.1 Qualitative results of SparseFusion on nuScenes *validation set*. . . . 112
A.2 Instance-level feature interaction in the fusion stage. Orange boxes are objects detected after the fusion stage with high confidence in the BEV space. Blue and green dots denote all instances from the LiDAR and camera branches separately. Orange boxes are connected with blue/green dots with blue/green lines. The strength of attention is represented by the darkness and thickness of the lines. . 113

B.1 Example input prompt and target output on WOMD-Planning-ADE. . . . 121
B.2 Additional qualitative results of motion planning. We show the *front left, front, front right* cameras for each case. . . . 123

C.1 Effect of CFG scales. . . . 127
C.2 Additional multi-modality data synthesis results using our proposed X-Drive. . . 130

D.1 Zero-Shot to Unseen Waymo Open Dataset Camera Configuration. . . . 134
D.2 Novel View Synthesis with Camera Rotation and Shifting. . . . 135
D.3 Multi-View Video Generation with Eight Cameras. . . . 136

D.4 Multi-View Video Generation with Six Cameras. 137

List of Tables

2.1 For each LiDAR-camera fusion method, we identify the *LiDAR candidates* and *camera candidates* that are used, the *transformation* process used to fuse these candidates into a unified space, and the *fusion outputs* generated to represent 3D scenes using information from both modalities. Based on these components, we categorize the methods into the following categories: **Dense+Sparse→Dense** approaches relate the sparse region proposals in images to a dense frustum point cloud and fuse them into a dense point space. **Dense+Dense→Dense** approaches align each point feature in the point cloud to the corresponding pixel feature in the image and represent the 3D scenes using dense features such as point/BEV features. **Sparse+Dense→Sparse** approaches generate sparse queries by detecting instance features in the point cloud and then applying cross-attention with dense image features. **Dense+Dense→Sparse** approaches predict objects using queries that combine dense features from each modality. **Sparse+Sparse→Sparse (ours)** extracts sparse instance features from each modality and directly fuses them to obtain the final sparse instance features used for detection. 9
2.2 Comparison with existing methods on nuScenes *validation set* and *test set*. . . . 16
2.3 Performance breakdown. We show the detection results in different parts of SparseFusion on the *validation set*. 'L' and 'C' refer to the LiDAR and camera modalities, respectively. 'ST' and 'GT' refer to semantic transfer and geometric transfer, respectively. 'VT' is the view transformation module. 18
2.4 Results on nuScenes *validation set* with stronger image backbones. We do not use any test-time augmentations or model ensembles. The inference speed is measured on a single NVIDIA A6000 GPU. † : CMT adopts flash-attention for transformer acceleration. 19
2.5 Results on nuScenes *validation set* for LiDAR or camera sensor malfunctions. The models are finetuned on the in-domain data in both cases. 20
2.6 Ablation study experiments to justify our design of each module. 21

3.1 3D object detection using SOLOFusion 33
3.2 3D object detection using BEVDet 34
3.3 Motion prediction using VAD-tiny 34
3.4 End-to-end planning using VAD-tiny 35

3.5 Ablation study results 37

4.1 Comparisons between WOMD-Planning-ADE and existing counterparts. WOMD-Planning-ADE has a significantly larger size and more comprehensive behavior-wise metrics. 41
4.2 NuScenes benchmark results. Most previous works require human annotations as additional supervision. S4-Driver* is pretrained on WOMD-Planning-ADE, while many prior MLLM-based methods also utilize extra data and labels due to the limited scale of nuScenes. 49
4.3 WOMD-Planning-ADE benchmark results. "*" denotes methods pretrained on the internal dataset. 50
4.4 Ablation studies on planning performance scalabity of our methods with regard to model and data sizes. 52
4.5 Ablation studies on MLLM inputs. 53
4.6 Ablation studies on sparse volume resolutions. 53

5.1 Quantitative comparison with driving data generation algorithms. For each column, the best value is highlighted by **bold**, and the second best one is denoted by underline. 66
5.2 Object-level control. 67
5.3 Ablation studies of X-Drive modules. 69

6.1 Multi-view Video Generation Performance. 80
6.2 Fidelity to Object Condition 82
6.3 Fidelity to Map Condition 82
6.4 Motion Cues in Generated Videos 82
6.5 Novel View Synthesis Performance with Camera Shifts. 83
6.6 Ablation Study on Camera Ray. 83
6.7 Ablation Study on Scale Causality and Model Size. 84

A.1 Category-wise performance on nuScenes *validation set* including the overall NDS, mAP, and AP for each category. 111

B.1 Behavior-wise sliced metrics on WOMD-Planning-ADE benchmark. "*" denotes methods with internal data pretraining. 122
B.2 Ablation studies on camera configurations. 124
B.3 Ablation studies on attention bias. 125
B.4 Ablation studies on multi-decoding aggregation. 126
B.5 Ablation studies on motion trajectory tokenization. 126

D.1 Ablation Study on Spatio-Temporal Module. 132
D.2 Ablation Study on Random Errors in Training. 132

Acknowledgments

Five years ago, I set foot on this continent for the first time, carrying both excitement and uncertainty about the Ph.D. journey ahead. Today, as I write this dissertation and arrive at the end of that journey, I find it hard to believe how quickly the years have passed. Five years are long enough to hold every kind of emotion: happiness and sadness, hope and discouragement, confidence and doubt. Yet five years are also astonishingly short, as if everything happened only yesterday. Looking back, the most valuable things I have gained are not only knowledge or publications, but also the courage, resilience, and calmness to face challenges. None of this growth would have been possible without the generous guidance and unwavering support of my advisors, mentors, peers, and family.

First and foremost, I would like to express my deepest gratitude to Professor Masayoshi Tomizuka. It has been my great fortune to have him as my Ph.D. advisor. Professor Tomizuka gave me the freedom to explore cutting-edge research topics while also offering steadfast support during difficult times and unexpected setbacks. What I have learned from him extends far beyond any specific research topics. More importantly, I have learned from his attitude, his philosophy, and his lifelong pursuit of excellence.

I also wish to express my sincere gratitude to my dissertation committee members, Professor Roberto Horowitz and Professor Jonathan Shewchuk, for the time and care they devoted to reviewing and strengthening this dissertation. I am also thankful to Professor Negar Mehr and Professor Anil Aswani for serving on my qualifying exam committee, where their thoughtful questions and constructive discussions helped shape the direction of my later research.

Beyond these faculty members, I would especially like to thank my research mentors. Dr. Hao-Shu Fang, my first research mentor during my undergraduate, also worked closely with me throughout my Ph.D. journey and taught me what research truly means and how to move a project forward. Dr. Chenfeng Xu led me into the field of 3D vision and taught me to think critically and identify the questions that matter most. Dr. Mingyu Ding provided steady support across multiple projects, especially during challenging and uncertain moments. Dr. Zheng Wu guided me step by step in the field of robotics and reinforcement learning. Dr. Wei Zhan offered invaluable guidance in autonomous driving research. I am also deeply grateful to my mentors during my internships at Cruise, Waymo, Applied Intuition, and Apple: Xin Huang, Hongge Chen, Tong He, Runsheng Xu, Yihan Hu, Depu Meng, Haofeng Chen, and Shreyash Pandey. It was their patience and support that helped to start my carrer.

I extend my heartfelt thanks to all my collaborators and co-authors, who shared precious insights and produced fruitful discussions : Professor Jianyu Chen, Professor Nhat Ho, Professor Kurt Keutzer, Professor Yong Jae Lee, Professor Wenzhao Lian, Professor Stefan Schaal, Professor Federico Tombari, Professor Junchi Yan, Professor Xiaokang Yang, Mazen Abdelfattah, Dragomir Anguelov, Ryan Brigden, Yuning Chai, Letian Chen, Yanjiang Guo, Quentin Herau, Eric Higgins, Jyh-Jing Hwang, Jingwei Ji, Zhaoqi Leng, Hubert Lin, Han Lu, Yiren Lu, Katie Luo, Gregory P. Meyer, Chensheng Peng, Alexander T. Pham, Marie-Julie Rakotosaona, Patrick Rim, Mingxing Tan, Yixiao Wang, Eric M. Wolff, Cheng-En Wu,

Jianwen Xie, and Jiezhi Yang. My sincere appreciation also goes to my other peers at the MSC Lab, including Jiachen Li, Shiyu Jin, Yujiao Cheng, Jessica Leu, Chen Tang, Yiyang Zhou, Huidong Gao, Hengbo Ma, Changhao Wang, Xinghao Zhu, Jinning Li, Lingfeng Sun, Catherine Weaver, Wu-te Yang, Xiang Zhang, Akio Kodaira, Chenran Li, Ran Tian, Yiheng Li, Ruihai Wu, Qiyang Qian, Wei-Jer Chang, Keita Kobashi, Jen-Wei Wang, Yuxin Chen, Boyuan Liang, John Viljoen, Yukata Shimizu, Pengcheng Wang, and Wenjie Li. They made our shared journey full of inspirations and happiness everyday.

Lastly, I would like to express my deepest gratitude to my family for their unconditional love and support. There were many difficulties and uncertainties throughout these five years. My parents, Mr. Zhentao Xie and Mrs. Xiaoshu Wang, always listened to me and encouraged me, despite the thousands of miles between us. My beloved girlfriend, and also my labmate, Shuqi Zhao always gave me understanding, care, and love even while carrying the pressures of her own Ph.D. journey. Her presence brought light to my world during its darkest hours. Finally, I also thank myself for having the courage to persist, the confidence to continue, and the willingness to grow throughout this journey.

School days behind, the world is ahead.

Chapter 1

Introduction

1.1 Background and Motivations

The development of autonomous driving systems represents one of the most challenging and transformative problems in modern artificial intelligence and robotics. Over the past decades, substantial progress has been made in both academia and industry, leading to large-scale deployments of autonomous vehicles in constrained environments and widespread adoption of advanced driver-assistance systems in consumer vehicles. These advances have been driven by rapid improvements in sensing hardware, computation, and data-driven learning algorithms.

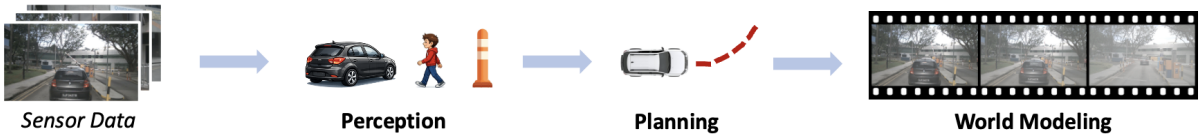
At a fundamental level, an autonomous driving system must continuously answer three essential questions:

- What is happening in the current scene?
- What should be done in response?
- What will happen in the future?

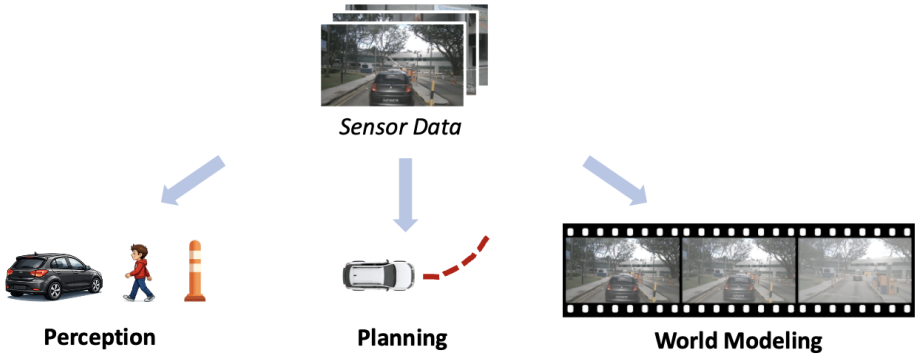
These questions correspond to perception, planning, and world modeling, which together form the backbone of modern autonomous driving systems.

1.1.1 Hierarchical Autonomous Driving Pipeline

Traditionally, these capabilities are implemented as separate modules in a hierarchical pipeline (Fig. 1.1a). Perception systems infer the state of surrounding dynamic agents and static scene structure from multi-modal sensor inputs such as cameras and LiDAR. Planning systems then operate on the perceived scene to choose an action and generate future ego motion under task objectives and safety constraints. Finally, world modeling or simulation aims to predict how the surrounding environment may evolve under different scene layouts and ego actions, which is critical for evaluation and closed-loop development.



(a) Traditional autonomous driving systems follow a hierarchical pipeline.



(b) This dissertation focuses on learning scalable world representations directly from raw sensor data.

Figure 1.1: Learning scalable world representations from raw sensor data instead of relying on strictly separated hierarchical stages.

While such modular designs offer interpretability and engineering flexibility, they also introduce several inherent limitations:

- **Error Accumulation:** Imperfect scene understanding at the perception stage is inevitably propagated to planning and world modeling, causing downstream errors to accumulate across the pipeline.
- **Information Mismatch:** Important information is often lost between modules. For example, planning systems may not have direct access to subtle visual cues in raw sensor data, while world models may fail to generate observations that are fully compatible with perception and planning modules.
- **Data Heterogeneity:** Perception, planning, and world modeling are often trained with different data formats, labels, and supervision signals, which limits data reuse and hinders overall scalability.

1.1.2 Towards Scalable World Representations

In recent years, learning-based approaches have begun to reshape this paradigm. As a special domain within robotics, autonomous driving has access to large-scale driving logs collected

from many human drivers with multiple synchronized sensors such as cameras, LiDAR, and IMU (Fig. 1.2). This abundance of raw sensory data creates an opportunity to rethink autonomous driving from the perspective of scalable representation learning rather than strictly separated task modules. These observations motivate the following central question of this dissertation: *Can we design scalable and effective learning-based representations for perception, decision-making, and future world modeling in autonomous driving?*

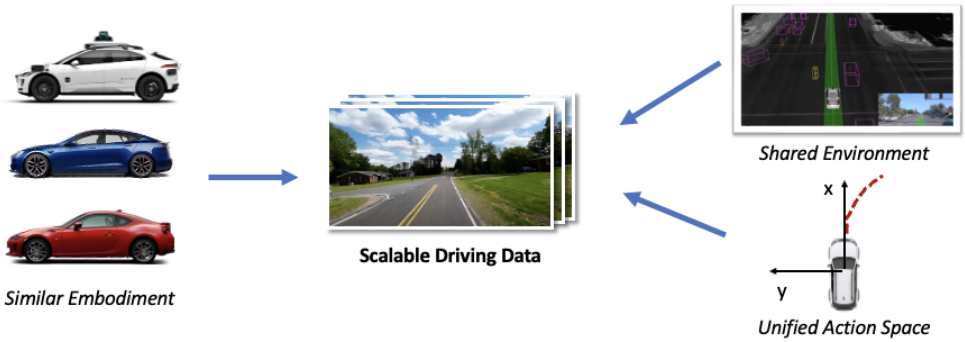


Figure 1.2: Scalable driving data. Human driving logs with raw sensor data are scalable because they share similar embodiment, environment structure, and action space.

The driving environment is inherently three-dimensional and dynamic, with strong structure in both space and time. A key insight of this dissertation is that autonomous driving systems should learn scalable world representations directly from raw sensor data, and that these representations can serve as a common foundation across perception, planning, and world modeling (Fig. 1.1b). Although different tasks may still require specialized designs, they all benefit from robust and efficient representations of the world. This perspective offers several advantages:

- **Preserving Task-Relevant Detail:** When representations are learned directly from raw sensor data, downstream modules are less constrained by the information bottlenecks imposed by earlier hand-designed stages.
- **Aligning Information Across Tasks:** End-to-end world representations provide a common interface across scene understanding, decision-making, and future generation. For example, a world model that generates realistic videos can better match the sensory inputs expected by perception and planning systems.
- **Improving Scalability:** Raw driving logs are far easier to collect at scale than task-specific annotations. Representations that can learn effectively from such data are therefore critical for robust and scalable autonomous driving.

1.2 Dissertation Outline and Contributions

This dissertation is organized around the three fundamental capabilities required by an autonomous driving system: perceiving the current scene, planning an appropriate action, and imagining how the world may evolve in the future. Rather than pursuing a single monolithic solution, this thesis studies these capabilities through five technical chapters for different aspects. Although the technical formulations differ across chapters, they are connected by a shared objective: to design scalable representations that preserve the structure of the driving environment and support rich reasoning across perception, prediction, planning, and world modeling.

- Chapter 2 begins from the perspective of perception, presenting a multi-sensor 3D object detection framework for LiDAR-camera fusion. The key idea is to move away from dense scene representations and instead operate directly on sparse instance-level candidates from each modality. The algorithm achieves both state-of-the-art detection performance and strong computational efficiency. This research has been publish in ICCV 2023 [236]

- Chapter 3 and 4 go beyond scene understanding and study the decision making and motion planning based on visual observation. Chapter 3 investigates how to learn temporally coherent instance representations from unlabeled multi-frame observations to improve the quality of visual representations for diverse downstream tasks including perception, mapping, prediction, and planning. Chapter 4 exploits the powerful multimodal large language model (MLLM) for end-to-end motion planning. To bridge the gap between pretrained 2D visual reasoning and 3D driving decision-making, this chapter introduces a sparse spatio-temporal volume representation that aggregates multi-view and multi-frame observations into a unified spatial structure. This research has been published in ICRA 2025 [233] and CVPR 2025 [237].

- Chapter 5 and 6 expands the thesis toward generative world modeling. Chapter 5 proposes a framework for cross-modality consistent synthesis of LiDAR point clouds and multi-view images by modeling the joint distribution of multiple sensor modalities through cross-modality interactions. Chapter 6 takes a step further to address the long-horizon future imagination through a geometry-agnostic multi-view video generation algorithm. It introduces a unified 4D spatio-temporal representation in continuous ray space together with a dual-causal autoregressive framework to model the future evolution of the scene itself conditioned by diverse control signals and camera configurations. This research has been published in ICLR 2025 [235] and CVPR 2026 [234].

- Chapter 7 concludes the dissertation by synthesizing the main insights from these stages, discussing their broader implications for autonomous systems, and outlining future directions toward more unified and scalable world models.

Taken together, these chapters contribute a coherent representation-centric view of autonomous driving. Across the dissertation, the proposed methods reveal a clear progression:

- from sparse instance-level representations for scene understanding,
- to structured spatial representations for decision-making,
- to continuous spatio-temporal representations for future world modeling.

This progression reflects the broader claim of the thesis: advances in autonomous driving benefit from better ways of representing the 3D world across sensing, acting, and forecasting. By designing representations that are efficient, structured, and scalable, we can build learning systems that more effectively bridge perception, planning, and world modeling.

Part I

Perception: Understanding the Current World

Chapter 2

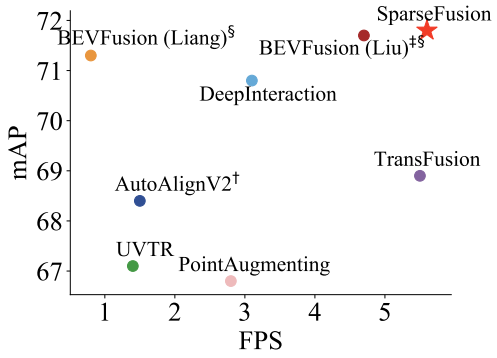
SparseFusion: Fusing Multi-Modal Sparse Representations for Multi-Sensor 3D Object Detection

This chapter begins the dissertation from the perspective of *Perception*. Before an autonomous vehicle can make decisions or reason about the future, it must first form an accurate and efficient representation of the current scene from sensor inputs. This chapter argues that such a representation should be aligned with a defining property of driving environments: the world is spatially sparse, and only a small subset of regions and objects are critical for downstream autonomy tasks. Motivated by this observation, SparseFusion performs LiDAR-camera fusion directly on sparse instance-level candidates rather than dense scene features. By transforming modality-specific candidates into a unified 3D space and fusing them with a lightweight attention mechanism, the method achieves both state-of-the-art detection performance and strong computational efficiency on nuScenes.

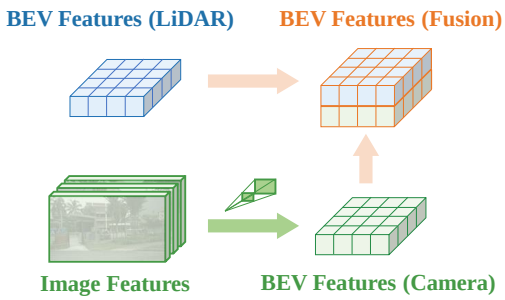
2.1 Introduction

Autonomous driving cars rely on multiple sensors, such as LiDAR and cameras, to perceive the surrounding environment. LiDAR sensors provide accurate 3D scene occupancy information through point clouds with points in the xyz coordinate space, and cameras provide rich semantic information through images with pixels in the RGB color space. However, there are often significant discrepancies between representations of the same physical scene acquired by the two sensors, as LiDAR sensors capture point clouds using 360-degree rotation while cameras capture images from a perspective view without a sense of depth. This impedes an effective and efficient fusion of the LiDAR and camera modalities. To tackle this challenge, multi-sensor fusion algorithms were proposed to *find correspondences between multi-modality data to transform and fuse them into a unified scene representation space.*

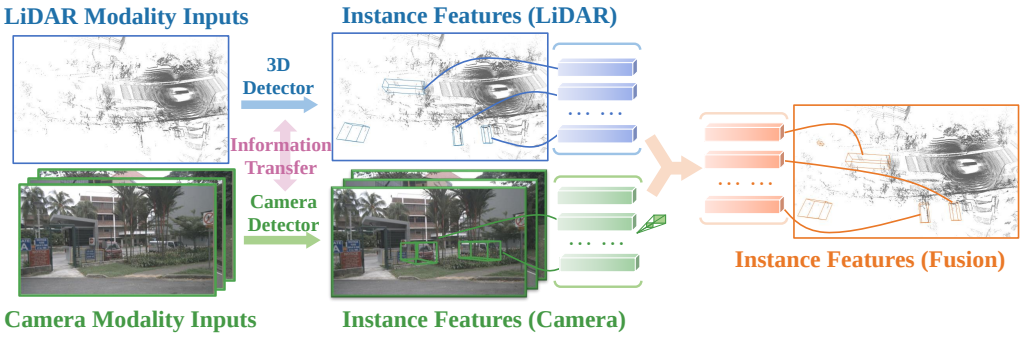
Dense representations, such as bird-eye-view (BEV), volumetric, and point representa-



(a) mAP *vs.* FPS



(b) Dense-to-dense fusion.



(c) Overview of our sparse fusion strategy. We extract instance-level features from the LiDAR and camera modalities separately, and fuse them in a unified 3D space to perform detection.

Figure 2.1: Compared to existing fusion algorithms, SparseFusion achieves state-of-the-art performance as well as the fastest inference speed on nuScenes *test set*. † : Official code of AutoAlignV2 uses flip as test-time augmentation. ‡ : We use BEVFusion-base results in the official repository of BEVFusion to match the input resolutions of other methods. § : Swin-T is adopted as image backbone.

tions, are commonly used to represent 3D scenes [128, 31, 206, 123, 209, 95]. Most previous works fuse different modalities by aligning low-level data or high-level features to yield dense features that describe the entire 3D scene, *e.g.*, as shown in Fig 2.1b. However, for the task of 3D object detection, such dense representations are superfluous since we are only interested in instances/objects, which only occupy a small part of the 3D space. Furthermore, noisy backgrounds can be detrimental to object detection performance, and aligning different modalities into the same space is a time-consuming process. For example, generating BEV features from multi-view images takes 500ms on an RTX 3090 GPU [128].

In contrast, sparse representations are more efficient, and methods based on them have achieved state-of-the-art performance in multi-sensor 3D detection [5, 30, 109, 251]. These

Table 2.1: For each LiDAR-camera fusion method, we identify the *LiDAR candidates* and *camera candidates* that are used, the *transformation* process used to fuse these candidates into a unified space, and the *fusion outputs* generated to represent 3D scenes using information from both modalities. Based on these components, we categorize the methods into the following categories: **Dense+Sparse→Dense** approaches relate the sparse region proposals in images to a dense frustum point cloud and fuse them into a dense point space. **Dense+Dense→Dense** approaches align each point feature in the point cloud to the corresponding pixel feature in the image and represent the 3D scenes using dense features such as point/BEV features. **Sparse+Dense→Sparse** approaches generate sparse queries by detecting instance features in the point cloud and then applying cross-attention with dense image features. **Dense+Dense→Sparse** approaches predict objects using queries that combine dense features from each modality. **Sparse+Sparse→Sparse (ours)** extracts sparse instance features from each modality and directly fuses them to obtain the final sparse instance features used for detection.

Category	Method	LiDAR Candidate	Camera Candidate	Transformation	Fusion Outputs
Dense+Sparse→Dense	Frustum PointNets [160]	point features	region proposals	proj. & concat.	point features
Dense+Dense→Dense	PointPainting [206]	point features	segm. output	proj. & concat.	point features
	PointAugmenting [209]	point features	image features	proj. & concat.	point features
	PAI3D [123]	point features	segm. output	proj. & concat.	point features
	BEVFusion [128]	BEV features	image features	depth. est. & proj. & concat.	BEV features
	AutoAlignV2 [31]	voxel features	image features	proj. & attn.	voxel features
Sparse+Dense→Sparse	TransFusion [5]	instance features	image features	proj. & attn.	instance features
Dense+Dense→Sparse	FUTR3D [30]	voxel features	image features	attn.	instance features
	UVTR [109]	voxel features	image features	depth. est. & proj. & attn.	instance features
	DeepInteraction [251]	BEV features	image features	proj. & attn.	instance features
	CMT [242]	BEV features	image features	attn.	instance features
Sparse+Sparse→Sparse	SparseFusion (**Ours**)	instance features	instance features	proj. & attn.	instance features

methods use object queries to represent instances/objects in the scene and interact with the original image and point cloud features. However, most previous works do not take into account the significant domain gap between features from different modalities [240]. The queries may gather information from one modality that has a large distribution shift with respect to another modality, making iterative interaction between modalities with large gaps sub-optimal. Recent work [251] mitigates this issue by incorporating modality interaction, *i.e.* performing cross-attention between features from two different modalities. However, the number of computations performed in this method increases quadratically with the dimensions of features and is thus inefficient. We categorize previous works into four groups by identifying four key components, which are outlined in Table 2.1. Further discussion of the methods in these groups is presented in Sec. 2.2.

In this paper, we propose SparseFusion, a novel method (Fig. 2.1c) that simultaneously utilizes sparse candidates and yields sparse representations, enabling efficient and effective 3D object detection. SparseFusion is the first LiDAR-camera fusion method, to our knowledge, to perform 3D detection using exclusively sparse candidates and sparse fusion outputs. We highlight a key common ground between the two modalities: an image and a point cloud that

represent the same 3D scene will contain mostly the same instances/objects. To leverage this commonality, we perform 3D object detection on the inputs from each modality in two parallel branches. Then, the instance features from each branch are projected into a unified 3D space. Since the instance-level features are sparse representations of the same objects in the same scene, we are able to fuse them with a lightweight attention module [204] in a *soft* manner. This parallel detection strategy allows the LiDAR and camera branches to take advantage of the unique strengths of the point cloud and image representations, respectively. Nevertheless, the drawbacks of each single-modality detector may result in negative transfer during the fusion phase. For example, the point cloud detector may struggle to distinguish between a standing person and a tree trunk due to a lack of detailed semantic information, while the image detector is hard to localize objects in the 3D space due to a lack of accurate depth information. To mitigate the issue of negative transfer, we introduce a novel cross-modality information transfer method designed to compensate for the deficiencies of each modality. This method is applied to the inputs from both modalities prior to the parallel detection branches.

SparseFusion achieves state-of-the-art results on the competitive nuScenes benchmark [18]. Our instance-level sparse fusion strategy allows for a lighter network and much higher efficiency in comparison with prior work. With the same backbone, SparseFusion outperforms the current state-of-the-art model [251] with 1.8x acceleration. Our contributions are summarized as follows:

- We revisit prior LiDAR-camera fusion works and identify four important components that allow us to categorize existing methods into four groups. We propose an entirely new category of methods that exclusively uses sparse candidates and representations.
- We propose SparseFusion, a novel method for LiDAR-camera 3D object detection that leverages instance-level sparse feature fusion and cross-modality information transfer to take advantage of the strengths of each modality while mitigating their weaknesses.
- We demonstrate that our method achieves state-of-the-art performance in 3D object detection with a lightweight architecture that provides the fastest inference speed.

2.2 Related Work

LiDAR-based 3D Object Detection. LiDAR sensors are commonly used for single-modality 3D object detection due to the accurate geometric information provided by point clouds. For detection in outdoor scenes, most existing methods transform unordered point clouds into more structured data formats such as pillars [101], voxels [244, 40], or range views [48, 117]. Features are extracted by standard 2D or 3D convolutional networks, based on which a detection head is used to recognize objects and regress 3D bounding boxes. Mainstream detection heads apply anchor-based [272, 101] or center-based [271] structures. Inspired by the promising performance of transformer-based methods in 2D detection, some

recent works explore transformers as feature extractors [135, 181] or as detection heads [5, 142]. Our method is agnostic to the LiDAR-based detector used in the LiDAR branch, and the default setting uses TransFusion-L [5].

Camera-based 3D Object Detection. Camera-based 3D detection methods are also being studied increasingly. Early work performs monocular 3D object detection by attaching extra 3D bounding box regression heads [254, 215] to 2D detectors. In practice, scenes are often perceived by multiple cameras from different perspective views. Following LSS [154], methods like BEVDet [85] and BEVDepth [111] extract 2D features from multi-view images and project them into the BEV space. Other methods including DETR3D [221] and PETR [126] adapt techniques from transformer-based 2D object detection methods [275, 20] to learn correspondences between different perspective views through cross-attention using 3D queries. However, as revealed in [83], there inevitably exists some ambiguity when recovering 3D geometry from 2D images. In response, recent works [84, 113, 150] also explore the positive effects of temporal cues in camera-based 3D detection. In our proposed SparseFusion, we extend deformable-DETR [275] to monocular 3D object detection and explicitly transform the regressed bounding boxes to the LiDAR coordinate space.

Multi-Modality 3D Object Detection. LiDAR and cameras provide complementary information about the surrounding environment, so it is appealing to fuse the multi-modality inputs for 3D object detection tasks. As analyzed in Tab. 2.1, existing fusion methods can be classified into four categories. Early works tend to fuse multi-modality information into a unified dense representation. Frustum PointNets [160] utilizes a **Dense**+**Sparse**→**Dense** approach that filters dense point clouds with sparse 2D regions of interest. Subsequent works explore **Dense**+**Dense**→**Dense** approaches by working directly with the dense LiDAR modality and camera modality features instead. Methods such as [206, 209] project point clouds into image perspective views and concatenate the dense image features with point features. BEVFusion [128, 116] significantly improves the performance of this line of methods by projecting dense image features into the LiDAR coordinate space using estimated per-pixel depths. AutoAlignV2 [31] also considers the soft correspondence through cross-modality attention to increase the robustness. However, we point out that dense representations are altogether undesirable for 3D object detection as they are noisy and inefficient.

Recent works have begun to utilize object-centric sparse scene representations. TransFusion [5] adopts a **Sparse**+**Dense**→**Sparse** strategy by extracting sparse instance-level features from the LiDAR modality and refining them using dense image features. Other works [30, 109, 251, 242] utilize a **Dense**+**Dense**→**Sparse** approach where queries are used to extract a sparse instance-level representation from dense BEV and image features. However, it is hard to extract information from multi-modality features with an attention operation given the large cross-modal distribution shift. To this end, UVTR [109] projects image features into the LiDAR coordinate space, CMT [242] encodes modality-specific positional information to its queries, and DeepInteraction [251] proposes cross-modality interaction. However, these methods still need to resolve the large multi-modal domain gap by stacking many transformer layers to construct a heavy decoder.

In contrast to the above methods, our method adopts the previously unexplored **Sparse**+**Sparse**→

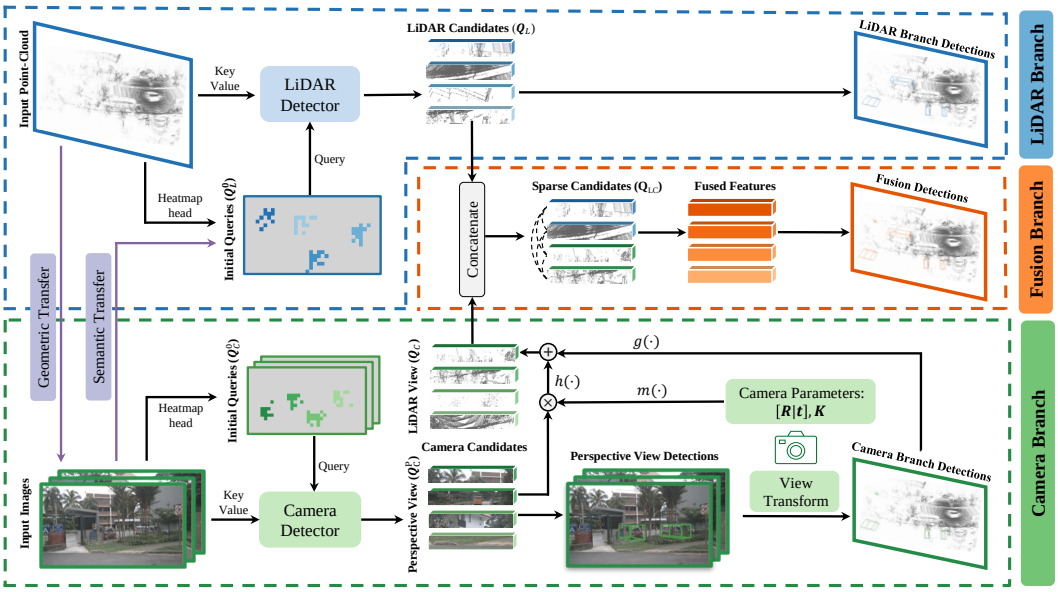


Figure 2.2: Overall framework of SparseFusion. It fuses sparse candidates from LiDAR and camera modalities to obtain a multi-modality instance-level representation in the unified LiDAR space.

Sparse approach. SparseFusion extracts sparse representations of both modalities and fuses them to generate a more accurate and semantically rich sparse representation that yields great performance while also achieving great efficiency.

2.3 Methodology

We present SparseFusion, an effective and efficient framework for 3D object detection via LiDAR and camera inputs. The overall architecture is illustrated in Fig. 2.2. We acquire sparse candidates from each modality using modality-specific object detection in the LiDAR and camera branches. The instance-level features generated by the camera branch are transformed into the LiDAR space of the instance-level features generated by the LiDAR branch. They are then fused with a simple self-attention module (Sec. 2.3.1). To mitigate the negative transfer between modalities, we apply a geometric transfer module and a semantic transfer module prior to the parallel detection branches (Sec. 2.3.2). Furthermore, we design custom loss functions for each module to ensure stable optimization (Sec. 2.3.3).

2.3.1 Sparse Representation Fusion

Acquiring candidates in two modalities. *LiDAR candidates.* We follow TransFusion-L [5] and use only one decoder layer for LiDAR modality detection. The LiDAR backbone ex-

tracts a BEV feature map from the point cloud inputs. We initialize N_L object queries $\mathbf{Q}_L^0 = \{\mathbf{q}_{L,i}^0\}_{i=1}^{N_L}, \mathbf{q}_{L,i}^0 \in \mathbb{R}^C$ as well as their corresponding reference points $\mathbf{p}_L^0 = \{\mathbf{p}_{L,i}^0\}_{i=1}^{N_L}, \mathbf{p}_{L,i}^0 \in \mathbb{R}^2$ in the BEV plane. These queries interact with the BEV features through a cross-attention layer to generate the updated object queries $\mathbf{Q}_L = \{\mathbf{q}_{L,i}\}_{i=1}^{N_L}, \mathbf{q}_{L,i} \in \mathbb{R}^C$. These updated queries $\mathbf{Q}_L$ represent the instance-level features of objects in the LiDAR modality, and we use them as the LiDAR candidates in the subsequent multi-modal fusion module. Furthermore, we apply a prediction head to each query to classify the object and regress the bounding box in LiDAR coordinate space.

Camera candidates. To generate the camera candidates, we utilize a camera-only 3D detector with images from different perspective views as inputs. Specifically, we extend deformable-DETR [275] with 3D box regression heads. We also initialize N_C object queries $\mathbf{Q}_C^0 = \{\mathbf{q}_{C,i}^0\}_{i=1}^{N_C}, \mathbf{q}_{C,i}^0 \in \mathbb{R}^C$ along with their corresponding reference points $\mathbf{p}_C^0 = \{\mathbf{p}_{C,i}^0\}_{i=1}^{N_C}, \mathbf{p}_{C,i}^0 \in \mathbb{R}^2$ on the image. For each perspective view v, queries on its image interact with the corresponding image features using a deformable attention layer [275]. The outputs of all perspective views comprise the updated queries $\mathbf{Q}_C^P = \{\mathbf{q}_{C,i}^P\}_{i=1}^{N_C}, \mathbf{q}_{C,i}^P \in \mathbb{R}^C$. We use these queries as the camera candidates in the subsequent multi-modal fusion module. We provide further details of our architecture, the initialization method, and the prediction heads for the two modalities in our supplementary materials.

Transformation After acquiring the candidates from each modality, we aim to transform the candidates from the camera modality to the space of the candidates from the LiDAR modality. Since the candidates from the camera modality are high-dimensional latent features that are distributed differently than the candidates from the LiDAR modality, a naive coordinate transformation between modalities is inapplicable here. To address this issue, we disentangle the representations of the camera candidates. Intrinsically, a camera candidate is an instance feature that is a representation of a specific object's class and 3D bounding box. While an object's class is view-invariant, its 3D bounding box is view-dependent. This motivates us to focus on transforming high-dimensional bounding box representations.

We first input the candidate instance features into the prediction head of the camera branch. We label the outputted bounding boxes as $\mathbf{b}^P$. Given the extrinsic matrix $[\mathbf{R}_v|\mathbf{t}_v]$ and intrinsic matrix $\mathbf{K}_v$ of the corresponding v-th camera, the bounding boxes can be easily projected into the LiDAR coordinate system. We denote the project bounding boxes as $\mathbf{b}^L$. We encode the projected bounding boxes with a multi-layer perceptron (MLP) $g(\cdot)$, yielding a high-dimensional box embedding. We also encode the flattened camera parameter with another MLP $m(\cdot)$ to obtain a camera embedding. The camera embedding is multiplied with the original instance features as done in [111], which are then added to the box embedding, given by

$$\mathbf{q}_{C,i}^L = g(\mathbf{b}_i^L) + h(\mathbf{q}_{C,i}^P \cdot m(\mathbf{R}_v, \mathbf{t}_v, \mathbf{K}_v)), \tag{2.1}$$

where $h(\cdot)$ is an extra MLP to encode the query features in the perspective view. $h(\cdot)$ aims to preserve view-agnostic information while discarding view-specific information. Afterward, $\mathbf{Q}_C^L = \{\mathbf{q}_{C,i}^L\}$ is passed through a self-attention layer to aggregate information from multiple cameras to get the updated queries $\mathbf{Q}_C$ which represent the image modality instance features

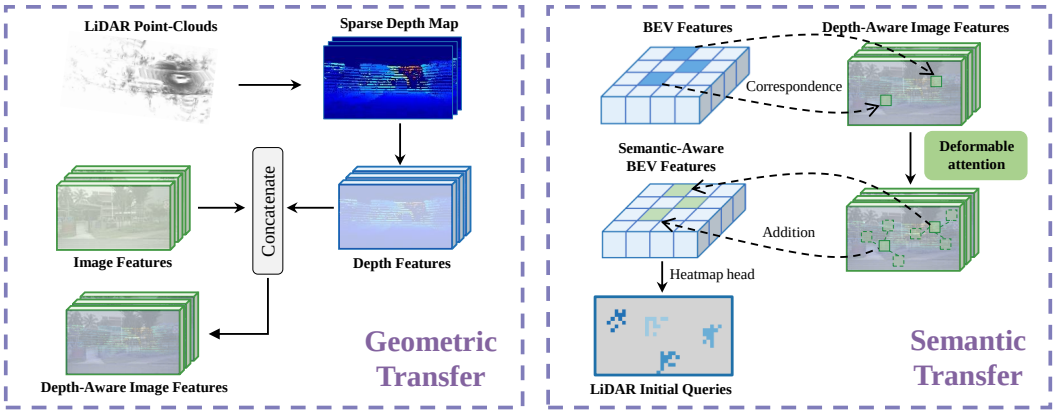


Figure 2.3: Cross-modality information transfer. We transfer geometric information from LiDAR to camera modality, and semantic information from camera to LiDAR modality.

in the LiDAR space.

Sparse candidate fusion. Our parallel modality-specific object detection provides sparse instance candidates $\mathbf{Q}_L = \{\mathbf{q}_{L,i}\}_{i=1}^{N_L}$ and $\mathbf{Q}_C = \{\mathbf{q}_{C,i}\}_{i=1}^{N_C}$ from the LiDAR and camera modalities respectively. After the above transformation of the camera candidates into LiDAR space, candidates from both modalities represent bounding boxes in the same LiDAR coordinate space, as well as the view-invariant categories. We now concatenate the candidates together:

$$\mathbf{Q}_{LC} = \{\mathbf{q}_{LC,i}\}_{i=1}^{N_L+N_C} = \{f_L(\mathbf{q}_{L,i})\}_{i=1}^{N_L} \cup \{f_C(\mathbf{q}_{C,i})\}_{i=1}^{N_C} \tag{2.2}$$

where $f_L(\cdot), f_C(\cdot)$ are learnable projectors. Afterward, we make novel use of a self-attention module to fuse the two modalities. Despite the simplicity of self-attention, the inherent intuition is novel: the modality-specific detectors encode the advantageous aspects of their respective inputs, and the self-attention module is able to aggregate and preserve the information from both modalities in an efficient manner. The output of the self-attention module is used for final classification and regression of the bounding boxes.

2.3.2 Cross-Modality Information Transfer

Although we aim to utilize the advantages of both modalities, we must address that the modalities also have their own disadvantages that can result in negative transfer between modalities. For example, the LiDAR detector struggles to capture rich semantic information, while the camera detector struggles to capture accurate geometric and depth information. To mitigate negative transfer, we propose novel geometric and semantic information transfer modules, as illustrated in Fig. 2.3, that we apply prior to the modality-specific detectors.

Geometric transfer from LiDAR to camera. We project each point in the LiDAR point cloud input to multi-view images to generate sparse multi-view depth maps. These multi-view depth maps are inputted into a shared encoder to obtain depth features, which are then

concatenated with the image features to form depth-aware image features that compensate for the lack of geometric information in camera inputs. The depth-aware image features are used as the input to the camera branch.

Semantic transfer from camera to LiDAR. We project the points in the LiDAR point cloud input to the image inputs, which yields sparse points on the image features. We perform max-pooling to aggregate the resulting multi-scale features, and we combine them with the BEV feature through addition. The concatenated features serve as the queries and interact with the multi-scale image features through deformable-attention [275]. The updated queries replace the original queries in the BEV features, which results in the semantic-aware BEV features, which are used for query initialization.

2.3.3 Objective Function

We apply the Gaussian focal loss [120] to the initialized queries of both modalities, given by

$$\mathcal{L}_{init} = \mathcal{L}_{GFocal}(\hat{\mathbf{Y}}_L, \mathbf{Y}_L) + \mathcal{L}_{GFocal}(\hat{\mathbf{Y}}_C, \mathbf{Y}_C), \tag{2.3}$$

where $\hat{\mathbf{Y}}_L, \hat{\mathbf{Y}}_C$ are the dense predictions of category-wise heatmaps of the LiDAR and camera modalities, respectively, and $\mathbf{Y}_L, \mathbf{Y}_C$ are the corresponding ground-truths.

Then, we apply the loss function for the detectors of the LiDAR and camera modalities, as well as the view transformation of the camera candidates and the candidate fusion stage. Firstly, the predictions of each modality-specific detector are independently matched with the ground-truth using the Hungary algorithm [100]. The object classification is optimized with focal loss [120] and the 3D bounding box regression is optimized with L1 loss. For the camera modality detector, the ground-truth bounding boxes are in separate camera coordinates. For all other detectors, ground-truth bounding boxes are in LiDAR coordinates. The detection loss can be represented as

$$\mathcal{L}_{detect} = \gamma \mathcal{L}_{detect}^{camera} + \mathcal{L}_{detect}^{trans} + \mathcal{L}_{detect}^{LiDAR} + \mathcal{L}_{detect}^{fusion}. \tag{2.4}$$

Our entire network is optimized using $\mathcal{L} = \alpha \mathcal{L}_{init} + \beta \mathcal{L}_{detect}$. In our implementation, we empirically set $\gamma = 1, \alpha = 0.1$, and $\beta = 1$ to balance different terms.

2.4 Experiments

2.4.1 Dataset and metrics

We follow previous work [251, 242, 116, 31] to evaluate our method on the nuScenes dataset [18]. It is a challenging dataset for 3D object detection, consisting of 700/150/150 scenes for training/validation/test. It provides point clouds collected using a 32-beam LiDAR and six images from multi-view cameras. There are 1.4 million annotated 3D bounding boxes for objects from 10 different classes. We evaluate performance using the nuScenes detection

Table 2.2: Comparison with existing methods on nuScenes *validation set* and *test set*.

Methods	Modality	LiDAR Backbone	Camera Backbone	*validation set* NDS	*validation set* mAP	*test set* NDS	*test set* mAP
FCOS3D [215]	Camera	-	ResNet-101 [68]	41.5	34.3	42.8	35.8
PETR [126]	Camera	-	ResNet-101 [68]	44.2	37.0	45.5	39.1
CenterPoint [254]	LiDAR	VoxelNet [272]	-	66.8	59.6	67.3	60.3
TransFusion-L [5]	LiDAR	VoxelNet [272]	-	70.1	65.1	70.2	65.5
PointAugmenting [206]	LiDAR+Camera	VoxelNet [272]	DLA34 [255]	-	-	71.0	66.8
FUTR3D [30]	LiDAR+Camera	VoxelNet [272]	ResNet-101 [68]	68.3	64.5	-	-
UVTR [109]	LiDAR+Camera	VoxelNet [272]	ResNet-101 [68]	70.2	65.4	71.1	67.1
TransFusion [5]	LiDAR+Camera	VoxelNet [272]	ResNet-50 [68]	71.3	67.5	71.6	68.9
AutoAlignV2 [31]	LiDAR+Camera	VoxelNet [272]	CSPNet [208]	71.2	67.1	72.4	68.4
BEVFusion [116]	LiDAR+Camera	VoxelNet [272]	Dual-Swin-T [115]	72.1	69.6	73.3	71.3
BEVFusion [128]	LiDAR+Camera	VoxelNet [272]	Swin-T [127]	71.4	68.5	72.9	70.2
DeepInteraction [251]	LiDAR+Camera	VoxelNet [272]	ResNet-50 [68]	72.6	69.9	73.4	70.8
CMT [242]	LiDAR+Camera	VoxelNet [272]	ResNet-50 [68]	70.8	67.9	-	-
SparseFusion (ours)	LiDAR+Camera	VoxelNet [272]	ResNet-50 [68]	**72.8**	**70.4**	**73.8**	**72.0**

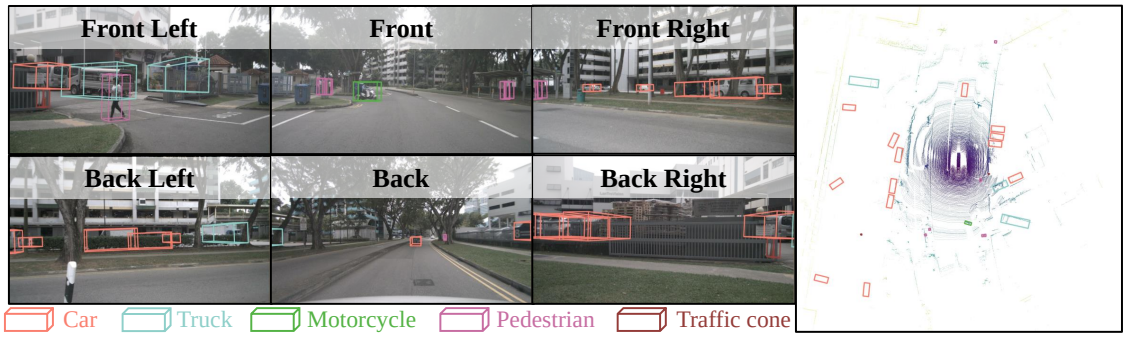


Figure 2.4: Qualitative results of SparseFusion on nuScenes *validation set*.

score (NDS) and mean average precision (mAP) metrics. The final mAP is averaged over distance thresholds 0.5m, 1m, 2m, and 4m on the BEV across 10 classes. NDS is a weighted average of mAP and other true positive metrics including mATE, mASE, mAOE, mAVE, and mAAE.

2.4.2 Implementation Details

Our implementation is based on the MMDetection3D framework [37]. For the camera branch, we use ResNet-50 [68] as the backbone and initialize it with the Mask R-CNN [67] instance segmentation network pretrained on nuImage [18]. The input image resolution is 800×448. For the LiDAR branch, we apply VoxelNet [272] with voxel size $(0.075m, 0.075m, 0.2m)$. The detection range is set as $[-54m, 54m]$ for the XY-axes and $[-5m, 3m]$ for the Z-axis.

Our LiDAR and camera modality detectors both include only 1 decoder layer. The query numbers for the LiDAR and camera modalities are set as $N_L = N_C = 200$, so our fusion stage can detect at most 400 objects per scene.

Since our framework disentangles the camera detector and the LiDAR detector, we can conveniently apply data augmentation separately to the LiDAR inputs and camera inputs. We apply random rotation, scaling, translation, and flipping to the LiDAR inputs, and we apply random scaling and horizontal flipping to the camera inputs. Our training pipeline follows previous works [5, 128, 251]. We first train TransFusion-L [5] as our LiDAR-only baseline, which is used to initialize our LiDAR backbone and LiDAR modality detector. This LiDAR-only baseline is trained for 20 epochs. Afterward, we freeze the pretrained LiDAR components and train the entire fusion framework for 6 epochs. For both training stages, we use the AdamW optimizer [129] with one-cycle learning rate policy [185]. The initial learning rate is 10^{-4} and the weight decay is 10^{-2}. The hidden dimensions in the entire model except the backbones are 128. For both training stages, we adopt CBGS [274] to balance the class distribution. We train our method on four NVIDIA A6000 GPUs with batch size 16.

2.4.3 Results and Comparison

We report results on the nuScenes *validation* and *test sets*, without using any test-time augmentations or model ensembles. As shown in Tab. 2.2, SparseFusion significantly improves our LiDAR-only baseline, TransFusion-L [5], by +3.6% NDS and +6.3% mAP on the *test set*, due to the additional use of camera inputs. More importantly, SparseFusion sets a new state of the art on both the *validation set* and *test set*, outperforming prior works including those using stronger backbones. It is noteworthy that SparseFusion demonstrates a 0.4% NDS and 1.0% mAP improvement over the most recent state-of-the-art [251] while also achieving a 1.8x speedup (5.6 FPS vs. 3.1 FPS) on A6000 GPU. It can be seen in Fig. 2.1a that, in addition to superior performance, SparseFusion also provides the fastest inference speed. We also demonstrate the performance of SparseFusion by visualizing some qualitative results in Fig. 2.4.

2.4.4 Analysis

Performance Breakdown SparseFusion performs 3D object detection in both the LiDAR and camera branch separately. Tab. 2.3 shows the detection performance in different parts of SparseFusion including the LiDAR branch, camera branch (before and after view transformation), and the fusion branch. We notice that our LiDAR branch detection results notably surpass the LiDAR-only baseline TransFusion-L [5] since the proposed semantic transfer can compensate for the weakness of point cloud inputs. In comparison with the state-of-the-art single-frame camera detector PETR [126] (six decoder layers), our camera branch achieves much better performance with just one decoder layer, owing to the depth-aware features from the proposed geometric transfer. Besides, our view transformation module not only trans-

forms the instance features from the camera coordinate space into the LiDAR coordinate space, but it also slightly improves the camera branch detection performance by aggregating multi-view information. With this strong performance, the modality-specific detectors in each branch would not cause negative transfer during fusion. We also consider direct Non-Maximum Suppression (NMS) to combine the detection results of LiDAR and camera branches, which is even worse than the single LiDAR branch. Therefore, it is also important to make the instance-level features from the two branches interact with each other in the fusion stage.

Table 2.3: Performance breakdown. We show the detection results in different parts of SparseFusion on the *validation set*. 'L' and 'C' refer to the LiDAR and camera modalities, respectively. 'ST' and 'GT' refer to semantic transfer and geometric transfer, respectively. 'VT' is the view transformation module.

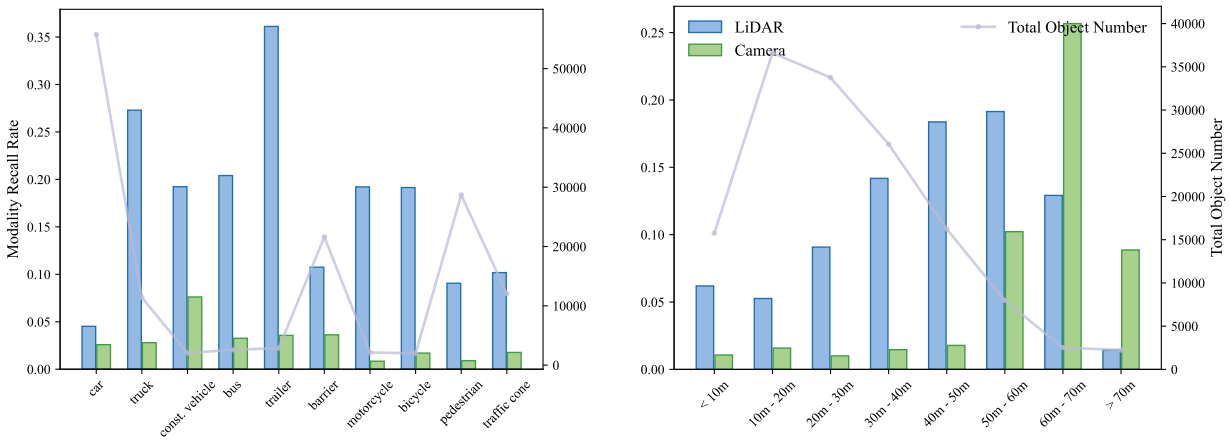
Methods	Modality	NDS	mAP
TransFusion-L [5]	L	70.1	65.1
LiDAR branch	L + ST	**71.8**	**68.4**
PETR (ResNet50) [126]	C	38.1	31.3
Camera branch (before VT)	C+GT	43.5	40.6
Camera branch (after VT)	C+GT	**44.3**	**41.5**
LiDAR/camera branches NMS	L+C	71.5	68.2
SparseFusion	L+C	**72.8**	**70.4**

Strong Image Backbone. We incorporate stronger Swin-T [127] backbone into SparseFusion to match some previous work [116, 242]. Tab. 2.4 compares the performance of different methods on the nuScenes *validation set*. We do not include any test-time augmentations or model ensembles. Although multi-modality detection relies more on the LiDAR inputs, SparseFusion can still benefit from a stronger image backbone and beat all the counterparts.

Modality-Specific Object Recall. The parallel detectors in the LiDAR and camera branches enable us to determine which modality recalls each object. Given $N_L + N_C$ predictions, we know that the first N_L instances come from the LiDAR-modality detector, while the last N_C are from the camera modality. An object is recalled if a bounding box with correct classification is predicted within *a radius of two meters* around its ground-truth position. In Fig. 2.5, we demonstrate the percentage of objects in the nuScenes *validation set* recalled by exactly one modality. We observe that each modality can compensate for the weakness of the other to some extent. Although the LiDAR modality is typically more powerful, the camera modality plays an important role in detecting objects from classes such as construction vehicles and barriers. Furthermore, the camera modality is useful for detecting objects at far distances where point clouds are sparse or the LiDAR feature map cannot cover.

Table 2.4: Results on nuScenes *validation set* with stronger image backbones. We do not use any test-time augmentations or model ensembles. The inference speed is measured on a single NVIDIA A6000 GPU. † : CMT adopts flash-attention for transformer acceleration.

Methods	Image Backbone	Input Resolution	NDS	mAP	FPS
BEVFusion [116]	Dual-Swin-T	1600×900	72.1	69.6	0.8
CMT [242]	VoVNet-99	1600×640	<u>72.9</u>	70.3	3.8†
SparseFusion	ResNet-50	800×448	72.8	<u>70.4</u>	**5.6**
SparseFusion	Swin-T	800×448	**73.1**	**71.0**	<u>5.3</u>



(a) Objects of different classes.

(b) Objects at different distances.

Figure 2.5: Percentage of objects recalled by each modality. We do not count those objects recalled by both or neither.

Cross-Modality Sparse Representation Interaction. Fig. 2.6 visualizes the instance feature interaction in the sparse fusion stage. The strength of attention between instance-level features is reflected by the thickness and darkness of lines. We notice that most objects can aggregate multi-modality instance-level features during fusion. Although the strongest interactions exist mainly between neighboring instances, it is interesting that features from the camera modality are also able to deliver strong interactions with the instances at the distant range. This could be a result of the shared semantics among objects in images.

Robustness to Sensor Malfunction. With the relatively independent modality-specific detectors, SparseFusion is robust to sensor malfunctions. We evaluate SparseFusion in two different cases: (1) The LiDAR scan range is limited to $(-90°, 90°)$ and (2) The most important front camera is missing while other cameras are preserved. SparseFusion outperforms

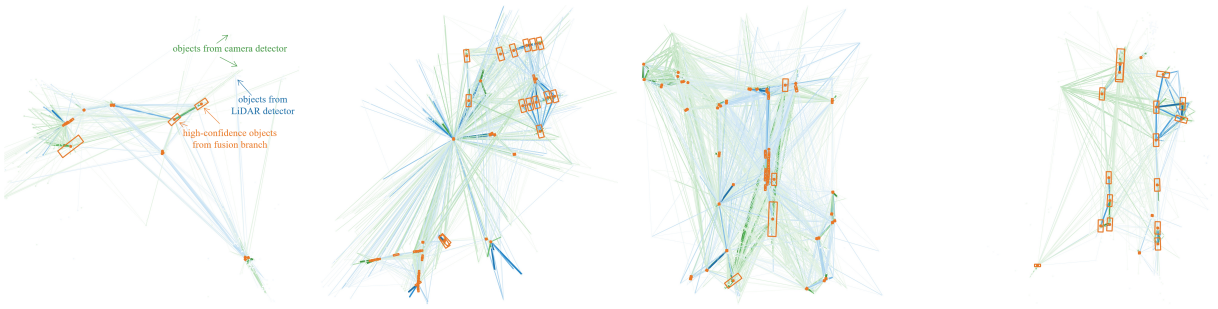


Figure 2.6: Instance-level feature interaction in the fusion stage. Orange boxes are detected after the fusion stage with high confidence in the BEV space. Blue and green dots denote all instances from the LiDAR and camera branches separately. Orange boxes are connected with blue/green dots with blue/green lines. The attention strength is represented by the darkness and thickness of the lines.

Table 2.5: Results on nuScenes *validation set* for LiDAR or camera sensor malfunctions. The models are finetuned on the in-domain data in both cases.

Methods	Sensor Malfunction	NDS	mAP
TransFusion [5]	Limited LiDAR Field $(-90°, 90°)$	49.2	31.1
BEVFusion [116]		54.9	45.5
SparseFusion		**61.2**	**54.3**
TransFusion [5]	Missing Front Camera	70.1	65.3
BEVFusion [116]		70.7	65.9
SparseFusion		**72.1**	**69.2**

previous methods [5, 116] with a large gap under both scenarios, demonstrating great robustness to sensor malfunctions.

2.4.5 Ablation Studies

In this section, we study the effect of using alternatives for the different modules in SparseFusion. For our ablation studies, we train on a *1/5 split* of nuScenes *training set* and evaluate on the full nuScenes *validation set*.

Sparse Fusion Strategy. We compare our self-attention module for sparse feature fusion with other fusion methods. In addition to our self-attention module in Sec. 2.3.1, three alternatives are considered: 1) Instance-level candidates from the two branches are directly fed into an MLP without any cross-instance aggregation. 2) LiDAR instance candidates are used as queries and camera instance candidates are used as keys/values. A cross-attention

Table 2.6: Ablation study experiments to justify our design of each module.

(a) Sparse fusion strategies.

Fusion Strategy	NDS	mAP
MLP	67.8	64.5
Cross-Attention	68.6	65.8
Optimal Transport	68.0	65.6
Self-Attention	**68.8**	**66.4**

(b) Information transfers.

Geometric	Semantic	NDS	mAP
✔	✗	67.7	64.2
✗	✔	68.4	65.7
✔	✔	**68.8**	**66.4**

(c) Parallel modality-specific detectors.

Seq. Pos.	Seq. Feat.	NDS	mAP
✔	✗	68.1	65.3
✔	✔	67.9	64.2
✗	✗	**68.8**	**66.4**

module is then used to fuse multi-modality instance features. 3) We make novel use of optimal transport in LiDAR-camera fusion [153]. We propose to learn a distribution transformation from camera candidates to LiDAR candidates through optimal transport. Then, we can directly fuse them by concatenating the candidates of two branches along the channel dimension. More details about this method are provided in appendix.

The results in Tab. 2.6a show that only cross-attention achieves competitive performance to self-attention. Yet, we observe that it relies so heavily on the output of the LiDAR branch that the camera branch is not fully utilized to compensate for the weaknesses of the LiDAR branch. The MLP strategy has limited performance as it does not fuse cross-instance and cross-modality information. Despite the impressive progress of optimal transport in other fields, it fails to learn the correspondences between the instance features of the two modalities and thus has limited performance. In contrast, self-attention is simple, efficient, and effective.

Information Transfer. We then ablate the geometric and semantic transfer between the LiDAR and camera modalities. The results in Tab. 2.6b show that the fusion performance benefits from both transfers. This also validates that the disadvantages of both modalities result in negative transfer and that our proposed information transfer modules are indeed effective in mitigating this issue. The semantic transfer module especially improves the final performance since it compensates for the LiDAR modality's lack of semantic information, which is critical for 3D detection.

Despite the importance of the information transfer, SparseFusion trained on the full training set can reach 71.5 NDS on the *validation set* even if we remove both the semantic and geometric transfers. This still outperforms many other advanced algorithms [5, 128], reflecting the robustness of our fusion stage to the modality-specific weakness.

View Transformation. As explained in Sec. 2.3.1, we transform the sparse representations of both modalities into one unified space. To validate the effectiveness of this approach, we ablate the view transformation of camera candidates into the LiDAR coordinate space. This results in a more straightforward method where we simply obtain the predictions of two modalities and directly fuse them using self-attention. Ablating the view transformation drops performance from 66.4% mAP and 68.8% NDS to 65.6% mAP and 68.3% NDS, respectively. This demonstrates that the view transformation is indeed helpful to overall performance.

Parallel Detectors. In addition to the ablation studies on the modules of the proposed pipeline, we also study alternatives to the structure of the pipeline. SparseFusion uses separate 3D object detectors for the LiDAR and camera modalities in parallel for extracting instance-level features to address the cross-modality domain gap. Alternatively, we consider a sequential pipeline where the camera detector runs after the LiDAR detector. The camera-modality detector inherits the output queries from the LiDAR detector. We consider two variants of this inheritance: 1) using the 3D position and instance features for the initial query of the camera modality; 2) using the 3D position but initializing the instance features from the corresponding image features. The camera detector follows the structure of the PETR [126] decoder (one layer). Tab. 2.6c shows that both sequential structures yield notably inferior performance, justifying our use of parallel modality-specific detectors.

2.5 Conclusion

We revisit previous LiDAR-camera fusion works and propose SparseFusion, a novel 3D object detection method that utilizes the rarely-explored strategy of fusing sparse representations. SparseFusion extracts instance-level features from each modality separately via parallel 3D object detectors and then treats the instance-level features as the modality-specific candidates. Afterward, we transform the candidates into a unified 3D space, and we are able to fuse the candidates with a lightweight attention module. Extensive experiments demonstrate that SparseFusion achieves state-of-the-art performance on the nuScenes benchmark with the fastest inference speed. We hope SparseFusion will serve as a powerful and efficient baseline for further research into this field.

Part II

Planning: Learning Actionable Representations

Chapter 3

Cohere3D: Exploiting Temporal Coherence for Unsupervised Representation Learning of Vision-based Autonomous Driving

This chapter begins the *Planning* part of the dissertation by addressing a prerequisite for learning-based decision-making: the ability to build temporally coherent representations from raw driving observations. Before a system can produce reliable actions, it must first learn features that remain stable across changing viewpoints, object motions, and long temporal horizons. Cohere3D tackles this challenge through unsupervised representation learning from multi-frame visual observations, using raw LiDAR point clouds only to establish long-term temporal correspondence during pretraining. By encouraging the same instance to maintain a consistent representation over time while remaining distinguishable from others, Cohere3D learns transferable visual features that improve perception, mapping, prediction, and planning under limited supervision.

3.1 Introduction

Recent work in vision-based autonomous driving has demonstrated promising performance in detection [85, 111], map reconstruction [173, 118], prediction [56, 92], and planning [82, 93]. Despite the rich semantics provided by images, the absence of depth information poses a critical challenge in recovering 3D geometries from a single-frame camera input. In addition, overlapping is limited between fields of view for each camera in autonomous driving scenarios, which hinders the utilization of stereo approaches. In this case, multi-frame inputs become helpful by aggregating observations of same instances (or regions) from different perspective views at different time steps to determine their 3D states. However, the dynamic nature of driving scenes poses challenge in extracting reliable instance-level information across tempo-

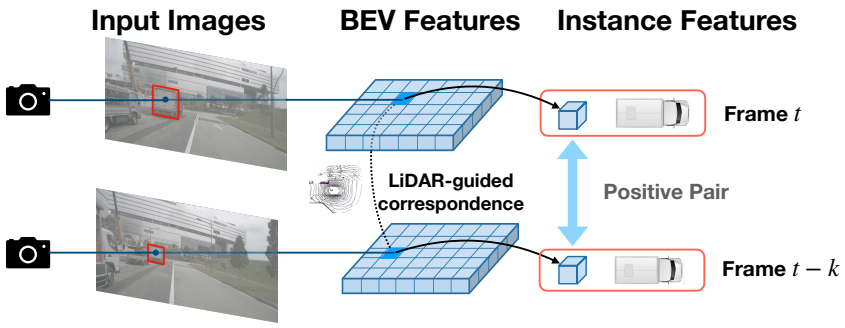


Figure 3.1: Temporal coherence of instance representations. Our algorithm encourages instance representations in the BEV space to be coherent across time and viewpoints.

ral frames. Varying relative positions between objects and cameras cause notable variations in their appearance in images, thereby impeding consistent instance identification over time. This inconsistency complicates establishing instance-level correspondence across multiple frames, thus becoming a fundamental obstacle to the effective fusion of multi-frame inputs. Hence, learning *long-term coherent local representations resilient to changes in observation viewpoint and distance* becomes imperative to facilitate temporal fusion.

Unfortunately, fine-grained annotations are needed to learn this robust representation, since all objects and their temporal correspondences must be specified. Given the immense human effort, only a small portion of collected data can be annotated. In this case, unsupervised learning provides a practical solution for exploiting all the unlabeled data. Previous exploration in this field focuses on LiDAR inputs [**nunes2023temporal**, 253]. Their representation learning algorithms are also limited to short time windows, whereas long-term temporal coherence is critical for vision-based autonomous driving.

For this reason, we propose a novel contrastive learning method, *Cohere3D*, designed to produce long-term coherent instance representations (Fig. 3.1) for vision-based autonomous driving tasks. Our approach leverages the richness of unlabeled sensor data while avoiding the constraints posed by the lack of annotations. Using commonly available raw LiDAR point clouds, our method constructs a long-term temporal correspondence for each instance. This correspondence serves as a foundation for extracting instance-level representations from vision-based bird's-eye-view (BEV) feature maps. It thus enhances the model's ability to maintain instance representation consistency across frames.

Our method focuses on the extraction and learning of long-term temporally coherent representation, a critical aspect but often overlooked in previous representation learning methods for driving tasks. By temporally aligning representations of the same instance and differentiating between distinct instances, our framework achieves notable improvement in both data efficiency and overall performance for downstream vision-based tasks such as detection, mapping, prediction, and planning. Our main contributions are as follows:

1. We give an insight into the long-term coherence of local representation to learn a robust representation for vision-based autonomous driving.

2. We introduce a simple, yet effective pretraining framework Cohere3D. It emphasizes temporal local correspondences through contrastive learning and is generic for various vision-based autonomous driving tasks.

3. We demonstrate superior performance and data efficiency of our algorithm in multiple key tasks, including detection, mapping, prediction, and planning.

3.2 Related Works

Camera-based Autonomous Driving Cameras are cheap and ubiquitous sensors commonly used in autonomous driving. To combine multi-view information in 3D perception tasks, some prior works [85, 111, 128] transform image features from different perspective views into a unified bird's-eye-view space through an LSS-style [154] projection strategy, while others [221, 126, 236] utilize a set of 3D queries to extra cross-view knowledge with transformer-based networks [20]. Since single-frame camera inputs lack reliable depth cues, some perception algorithms [84, 113, 150] exploit several temporally adjacent frames to recover the 3D geometry. Beyond perception tasks, recent work also develops end-to-end pipelines based on camera inputs. ViP3D [56] and PIP [92] predict motion trajectories using visual information. UniAD [82] and VAD [93] apply a joint training strategy to assist end-to-end planning.

High-quality 3D annotations are essential for all the above algorithms. Orthogonal to these works, we propose a novel unsupervised representation learning algorithm that makes full use of the abundant unlabeled sensor data.

Pretraining for Perception and Prediction Unsupervised learning algorithms [66, 55, 65, 6, 238] learn expressive representation without any human annotations. Some pioneering research adapts contrastive learning [232, 87] or masked autoencoder [148, 262] frameworks to 3D point clouds data, but most of them are constrained to a single-object or indoor data. To handle large outdoor scenes in autonomous driving scenarios, ProposalContrast [253] devises a region-level contrastive learning algorithm. BEV-MAE [121] and GeoMAE [198] follow MAE [65] to predict the missed information in the BEV space. Despite the increasing popularity of cameras in autonomous driving, most methods only consider point clouds as input. Differently, BEVDepth [111] and DD3D [149] highlight the importance of monocular depth estimation for vision-based 3D perception, but they neglect the holistic structure of surrounding scenes. Occ-BEV [141] predicts the occupancy of each voxel in the 3D space with image inputs. However, due to its lack of semantics and temporal cues, it provides limited benefits to complex perception tasks. Besides, the above algorithms ignore the prediction and planning tasks that require strong temporal reasoning capabilities. For prediction, previous pretraining works [110, 239] focus on the traditional two-stage motion prediction pipeline,

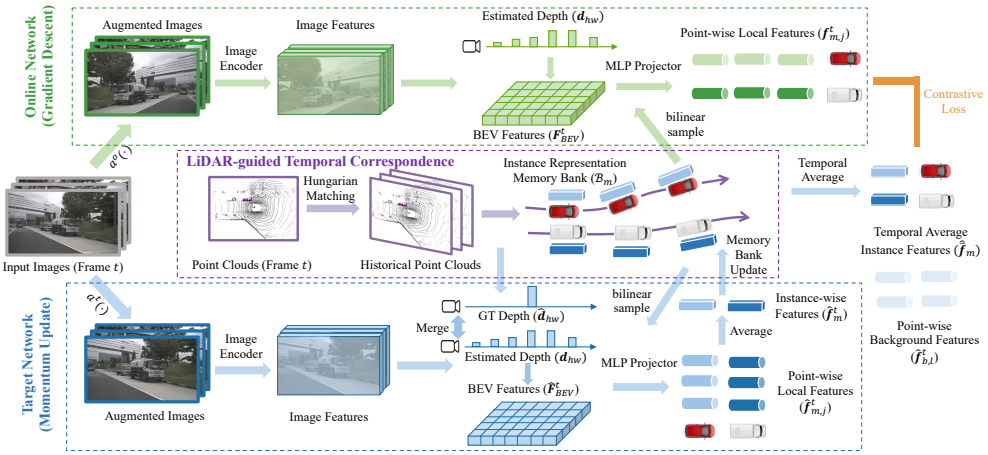


Figure 3.2: Overview of Cohere3D. We construct long-term instance-level temporal correspondence via unlabeled LiDAR point clouds, and design a contrastive learning framework to encourage the long-term coherence of instance representations.

which requires manually annotated historical trajectories and semantic maps as inputs and does not take end-to-end prediction approaches into account.

In contrast to these works, we emphasize the long-term temporal coherence of instance-level representations and propose a novel unsupervised learning algorithm targeted for diverse vision-based autonomous driving tasks.

Representation Learning from Temporal Cues Previous research [220, 201, 210, 59] uses the temporal information inside videos for unsupervised representation learning. In particular, several papers share common insights with us to develop contrastive learning algorithms to utilize fine-grained temporal correspondence. They construct the temporal correspondence using visual features [220], optical flow [62], or cycle-consistency [91]. However, they are limited to 2D tasks for image or video processing. TARL [**nunes2023temporal**] utilizes the temporal cues for the pretraining of LiDAR-based 3D perception models, but their clustering algorithm can only retrieve short-term temporal correspondences. In contrast, our algorithm targets the long-term temporal correspondence to learn a temporally coherent instance representation.

3.3 Methodology

In this section, we introduce our novel unsupervised representation learning algorithm Cohere3D. Our approach utilizes unlabeled multi-view images as the primary input. Additionally, during the pretraining phase, we incorporate raw LiDAR point clouds as supplementary guidance. It is important to note that these point clouds are not required during the inference stage of the downstream tasks.

The methodology is structured into several components: First, we formulate our method in Sec. 3.3.1. Next, Sec. 3.3.2 details how we construct temporal correspondences. It guides the extraction of temporal instance features, as elaborated in Sec. 3.3.3. Our approach is underpinned by a contrastive learning framework, described in Sec. 3.3.4. An overview of Cohere3D is shown in Fig. 3.2.

3.3.1 Formulation of Instance Feature Coherence

Camera-based perception often resorts to multiple frames to recover 3D object states. In essence, this process recovers the 3D state based on observations of the same instances from multiple perspective views at different time steps:

$$\mathbf{O}_m^{3D,t} = g\left(\left[h\left(\mathbf{O}_m^{2D,t-k}\right)\right]_{k=0}^{K}\right), \tag{3.1}$$

where $\mathbf{O}_m^{3D,t}$ is the 3D state of instance m at time step t, $\mathbf{O}_m^{2D,t}$ is the visual pattern of the instance m from the image $\mathcal{I}^t$ at frame t, $h(\cdot)$ encodes the visual patterns at each time step separately, and $g(\cdot)$ recovers the 3D object states by combining multi-frame information.

In this case, it is necessary to find the instance-wise correspondence $\left[\mathbf{O}_m^{2D,t-k}\right]_{k=0}^{K}$, *i.e.* we identify the same instance m across multiple time steps. Due to the motion of the ego-vehicle and surroundings, instance shapes and appearances in 2D images can vary significantly at different time steps. Thus, it is hard to correlate each instance with instances at other frames across a long-term temporal sequence.

To ensure the effective fusion of multi-frame inputs, we should encourage the model $h(\cdot)$ to learn coherent features robust to temporal changes, facilitating the instance-wise temporal correspondence in downstream tasks. At each frame t, we project the image features to the BEV space as $\mathbf{F}_{BEV}^t$ [154], and extract the feature $\mathbf{f}_m^t$ corresponding to each instance $\mathbf{O}_m^{2D,t}$ from the BEV feature map. The goal of our approach is to make the same instance representation $[\mathbf{f}_m^{t-k}]_{k=0}^K$ temporally coherent and distinguish the representations of different objects $\{\mathbf{f}_m^t\}_{m=1}^M$. Concretely, we compute the temporal average representation $\bar{\mathbf{f}}_m$ for each instance m over multiple historical frames as follows:

$$\bar{\mathbf{f}}_m = \frac{1}{K+1}\sum_{k=0}^{K}\mathbf{f}_m^{t-k}. \tag{3.2}$$

Our representation learning encourages the instance feature $\mathbf{f}_m^t$ to be close to the temporally average representation $\bar{\mathbf{f}}_m$ and different from other instance representations $\bar{\mathbf{f}}_n, n \neq m$.

3.3.2 LiDAR-guided Temporal Correspondence

Given the varying appearance of each instance at different frames, it is difficult to find the instance-level temporal correspondence from pure vision inputs without any human annotations. Alternatively, we build the long-term temporal correspondence using raw LiDAR

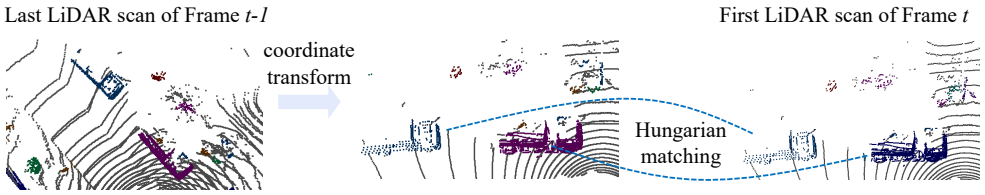


Figure 3.3: LiDAR-guided temporal correspondence construction. Instances are identified from point clouds within each frame via clustering. The Hungarian algorithm connects every two adjacent frames to generate long-term correspondence.

point clouds. LiDAR sensors have a higher frequency than cameras, so there may be multiple LiDAR scans within the small interval between two camera frames. For the camera image captured at time step t, which we refer to as frame t, we collect all LiDAR scans between time steps $t-1$ and t, to create the combined point cloud for frame t, denoted as $\mathcal{P}^t$. We apply a two-stage strategy to retrieve the long-term temporal correspondence (Fig. 3.3). In the first stage, we identify and distinguish different instances within $\mathcal{P}^t$ through intra-frame clustering. In the second stage, the identified instances are matched to generate inter-frame instance-wise correspondences.

Instance Identification We apply the method in [71] to remove the ground points (black points in Fig. 3.3) from $\mathcal{P}^t$ to obtain point clouds $\hat{\mathcal{P}}^t$. The points in $\hat{\mathcal{P}}^t$ are clustered with the HDBSCAN [19] into several clusters $\hat{\mathcal{P}}^t = \bigcup_{m=1}^{M} \mathcal{C}_m^t$, as customary in [**nunes2023temporal**, 229]. Although $\hat{\mathcal{P}}^t$ includes points from all LiDAR scans between time steps $t-1$ and t, we only consider the points from the first scan immediately after time step $t-1$ and the last scan immediately before t, denoted as $\mathcal{C}_m^{t,s}$ and $\mathcal{C}_m^{t,e}$. We would discard invalid cluster v if its point number is fewer than a threshold τ_n in either $\mathcal{C}_v^{t,s}$ or $\mathcal{C}_v^{t,e}$. Each remaining valid cluster $\mathcal{C}_m^t$ is identified as a single instance. The center location for an instance m may change between the first and last LiDAR scans, denoted as $\mathbf{c}_m^{t,s}$ and $\mathbf{c}_m^{t,e}$, which indicates the intra-frame motion of this instance.

Long-term Matching Given the instance-wise motions from $\{\mathbf{c}_m^{t,s}\}_{m=1}^{M}$ to $\{\mathbf{c}_m^{t,e}\}_{m=1}^{M}$ in a single frame t, we connect them to the previous frame $t-1$ by finding the correspondence between $\{\mathbf{c}_m^{t,s}\}_{m=1}^{M}$ at the beginning of frame t and the instance centers $\{\mathbf{c}_{m'}^{t-1,e}\}_{m'=1}^{M'}$ at the end of the frame $t-1$. Note that point clouds corresponding to different frames are in different coordinate systems due to the motion of the ego-vehicle. Given the rotation $\mathbf{R}$ and translation $\mathbf{p}$ of the ego-car between frame $t-1$ and t, the instance centers $\{\mathbf{c}_{m'}^{t-1,e}\}_{m'=1}^{M'}$ should be transformed into the coordinate of frame t:

$$\hat{\mathbf{c}}_{m'}^{t-1,e} = \mathbf{R}^T \mathbf{c}_{m'}^{t-1,e} - \mathbf{R}^T \mathbf{p}. \tag{3.3}$$

We follow [8, 76] to use Hungarian algorithm [100] to find the correspondence between $\{\mathbf{c}_m^{t,s}\}_{m=1}^{M}$ and $\{\hat{\mathbf{c}}_{m'}^{t-1,e}\}_{m'=1}^{M'}$. We set a threshold of $\tau_d = 0.5m$ for the distance between the matched instances, where the valid matching should satisfy $||\hat{\mathbf{c}}_{m'}^{t-1,e} - \mathbf{c}_m^{t,s}||_2 \leq \tau_d$. This

threshold can tolerate the possible object motion over a small interval ($< 0.05s$) between the two scans, while avoiding instance mismatching. To handle the instance disappearance or appearance, we pad empty instances with maximal distance to all the other instances. In this way, we combine several short-term intra-frame trajectories across adjacent frames to achieve the K-frame long-term trajectories $[\mathcal{C}_m^{t-k}]_{k=1}^K$ for each instance by connecting $\hat{\mathbf{c}}_{m'}^{t-k-1,e}$ and $\mathbf{c}_m^{t-k,s}$ frame by frame.

3.3.3 Temporal Instance Feature

For each frame t, we sample N_F points uniformly on the BEV space from all the clusters $\bigcup_{m=1}^M \mathcal{C}_m^t$ as $\bigcup_{m=1}^M \mathcal{S}_m^t$ where $\mathcal{S}_m^t \subseteq \mathcal{C}_m^t$ and $\sum_{m=1}^M |\mathcal{S}_m^t| = N_F$. Given the vision-based BEV feature map $\mathbf{F}_{BEV}^t$, we bilinearly sample features $\mathbf{f}_{m,j}^t$ for each point $\mathbf{s}_{m,j}^t \in \mathcal{S}_m^t$. At each frame t, we compute the instance-level representation by averaging the point-level features belonging to the same instance, *i.e.*,

$$\mathbf{f}_m^t = \frac{1}{N_m} \sum_{j=1}^{N_m} \mathbf{f}_{m,j}^t, \tag{3.4}$$

where $N_m = |\mathcal{S}_m^t|$ is the number of sampled points belonging to the instance m. For each instance, we maintain a memory bank $\mathcal{B}_m = [\mathbf{f}_m^{t-k}]_k$ for its representation at different time steps. The memory bank for each instance is created at the time step when it first appears and will be updated at each following time step as long as this instance is matched at the current frame t. Next, the temporal average representation $\bar{\mathbf{f}}_m$ of instance m is calculated following Eq. 3.2 based on the instance representation in the memory bank $\mathcal{B}_m$.

The foreground objects only account for a small part of images. To exploit semantics from the whole image, we also sample N_B background points uniformly on the BEV space not occupied by any valid clusters. The point features are extracted as $\{\mathbf{f}_{b,l}^t\}_{l=1}^{N_B}$ through bilinear sampling. Due to their static nature, the background representations are more stable. Therefore, we skip the temporal averaging operation.

3.3.4 Depth-aware Contrastive Learning

After obtaining instance-level features through LiDAR-guided temporal correspondence, we apply contrastive learning to encourage coherent instance representations, including some new techniques to ensure the stability and robustness of the unsupervised pretraining as follows.

Architecture We apply a Siamese network framework [55] with an online network $f(\cdot)$ and a target network $\hat{f}(\cdot)$. Both networks include an image backbone to encode the input images, a view transformation module to project the image features into the BEV space, and a BEV encoder to encode the BEV features. The online network is updated by gradient descent, while the target network uses the exponential moving average of the online network. For the input image $\mathcal{I}^t$ of frame t, we generate two randomly augmented copies $a^o(\mathcal{I}^t)$ and $a^t(\mathcal{I}^t)$

with a set of data augmentations $\mathcal{A}$, $a^o, a^t \in \mathcal{A}$. They are fed into the online and target networks separately, which output the BEV features $\mathbf{F}^t_{BEV}$ and $\hat{\mathbf{F}}^t_{BEV}$, where

$$\mathbf{F}^t_{BEV} = f(a^o(\mathcal{I}^t)), \quad \hat{\mathbf{F}}^t_{BEV} = \hat{f}(a^t(\mathcal{I}^t)). \tag{3.5}$$

Depth-aware Representation Learning Stable representations in the target network play an important role in contrastive learning [217]. However, due to the depth ambiguity of 2D inputs, it is unreliable to transform the image features from the perspective view to the BEV space. We follow [154, 85, 150] to project the image features with the estimated depth distribution $\mathbf{d}_{hw} = \left[p^1_{hw}, p^2_{hw}, \ldots, p^D_{hw}\right]$ for each pixel (h, w) *w.r.t.* a set of discrete depths defined by $[d_0+\Delta, d_0+2\Delta, \ldots, d_0+D\Delta]$ and $\sum_{i=1}^{D} p^i_{hw} = 1$. To get more reliable BEV representation during pretraining, we combine the ground-truth depth distribution $\hat{\mathbf{d}}_{hw}$ from the LiDAR sensor with the estimated depth distribution $\mathbf{d}_{hw}$, where $\hat{\mathbf{d}}_{hw} = \left[\hat{p}^1_{hw}, \hat{p}^2_{hw}, \ldots, \hat{p}^D_{hw}\right]$ is a one-hot distribution with only $\hat{p}^k_{hw} = 1$ if $d_0+k\cdot\Delta$ is the ground-truth depth. We name this index k_{GT} and merge the ground-truth depth and estimated depths by setting $p^{k_{GT}}_{hw} = 1$ in $\mathbf{d}_{hw}$ and then re-normalizing the merged depth distribution to ensure $\sum_{i=1}^{D} p^i_{hw} = 1$. We only merge the estimated and ground-truth depth for the target network while the online network only uses the estimated depth distribution. It equips the target network with positionally more accurate BEV features, enabling a more reliable self-supervision for pretraining.

View Transformation Mask We add a dropout mask during view transformation by setting a percentage r of the estimated depth distribution $\left[p^1_{hw}, p^2_{hw}, \ldots, p^D_{hw}\right]$ to zero. This is only adopted to the online network, since the target network requires a more stable representation for self-supervision. This encourages the online network to learn stronger BEV representation to compensate for the missing information.

Contrastive Loss Function Our basic goal is to encourage the instance-wise representation to be invariant to temporal changes. We extract point-level features at the current time step t from the BEV features of online network and target network respectively as in Sec. 3.3.3. The instance feature $\hat{\mathbf{f}}^t_m$ is derived from target network point-level features $\hat{\mathbf{f}}^t_{m,j}$ for each instance m following Eq. 3.4, while its temporal average $\hat{\bar{\mathbf{f}}}_m$ is calculated with Eq. 3.2. We get inspiration from [189] to apply a point-to-instance contrastive loss as follows:

$$\mathcal{L} = \frac{1}{N_F} \sum_{\mathbf{s}^t_{m,j} \in \mathcal{S}^t_m} \frac{\exp(\mathbf{f}^{tT}_{m,j} \hat{\bar{\mathbf{f}}}_m / \tau)}{\sum_{n=1}^{M} \exp(\mathbf{f}^{tT}_{m,j} \hat{\bar{\mathbf{f}}}_n / \tau) + \sum_{l=1}^{N_B} \exp(\mathbf{f}^{tT}_{m,j} \hat{\mathbf{f}}^t_{b,l} / \tau)}, \tag{3.6}$$

where τ is the temperature set to 0.1 and all the features are normalized to $||\mathbf{f}||_2 = 1$. This loss will make the local features belonging to the same instances close to each other across the long-term temporal sequence and distant from other instance features as well as background features.

3.4 Experiments

This section presents a comprehensive evaluation of Cohere3D. We begin by detailing the experimental setup in Sec. 3.4.1. Our approach is then compared against established baselines to demonstrate its effectiveness in vision-based autonomous driving tasks. Specifically, we evaluate our method on 3D object detection (Sec. 3.4.2), HD map construction (Sec. 3.4.3), motion prediction (Sec. 3.4.4), and planning (Sec. 3.4.5). Additionally, we provide an analysis of our algorithm in Sec. 3.4.6 and ablation studies in Sec. 3.4.7.

3.4.1 Implementation Details

Dataset We use nuScenes dataset [18] for both pretraining and finetuning. As the most common benchmark for vision-based driving tasks, it includes 700/150/150 scenes for training/validation/test. It provides point clouds collected by a 32-beam LiDAR and multi-view images from six cameras. For pretraining, we only use the raw point clouds and images from the training set without any annotations.
Unsupervised Pretraining In the pretraining, we follow the network architecture of short-term SOLOFusion [150] (ResNet50 backbone [68]) excluding detection heads. Different from 2D representation learning [66, 55], strong data augmentations potentially bring ambiguity in 3D geometry recovery from images [12] in our pretraining. Thus, we only apply weak data augmentations (same as [150]) during pretraining, including random crops, scaling, and rotations of the images in combination with random rotation and scaling for the BEV representation. We sample $N_F = N_B = 1000$ foreground and background points at each frame. The momentum of the target network is set as 0.99. To learn the long-term representation coherence, we use at most $K = 16$ (8 seconds) historical frames. A dropout percentage of $r = 0.3$ is adopted in the view transformation of the online network. We pretrain the model for 24 epochs using Adam optimizer [99] with batch size 32 and learning rate 1e-4.
Supervised Finetuning We initialize the network for different tasks with our pretrained online network weight for task-specific supervised finetuning. We follow the training strategies in their official settings for each task model.

3.4.2 Object Detection

To verify the effectiveness of our method on perception, we finetune a vision-based detector, SOLOFusion [150], for camera-based 3D detection, using our pretrained weights. We consider both short-term and long-term variants of the SOLOFusion model, where the former only considers one historical frame but the latter takes the past 16 frames as inputs. We finetune the model using $10\%, 20\%, 50\%, 100\%$ of the training set separately, and report the mAP and NDS metrics [18] on the nuScenes validation set in Tab. 3.1.

The most important advantage of unsupervised learning is to improve data efficiency with limited annotation by utilizing unlabeled data. Not surprisingly, for both short-term and long-term variants, Cohere3D significantly outperforms the baselines especially when only a

Table 3.1: 3D object detection using SOLOFusion

Finetune Percent	Pretraining	Short-term model NDS	Short-term model mAP	Long-term model NDS	Long-term model mAP
10%	ImageNet [150]	21.3	15.6	20.6	13.2
	Cohere3D (ours)	**24.3**	**19.4**	**22.9**	**16.4**
20%	ImageNet [150]	24.3	20.0	28.5	23.1
	Cohere3D (ours)	**25.8**	**22.3**	**32.3**	**25.0**
50%	ImageNet [150]	30.8	28.1	40.9	32.7
	Cohere3D (ours)	**32.4**	**29.1**	**43.1**	**33.6**
100%	ImageNet [150]	39.1	34.4	49.7	40.6
	BYOL [55]	37.2	33.2	48.9	39.7
	MonoDepth2 [54]	36.4	32.1	47.3	37.4
	Occ-BEV [141]	39.0	34.3	49.2	40.4
	Cohere3D (ours)	**41.3**	**35.1**	**50.2**	**40.8**

subset of the training data is used for finetuning. The pretraining also provides performance gains even if we finetune the model on the full training set, where the pretraining does not introduce extra data but exploits the inherent temporal cues. We also compare Cohere3D with several other pretraining baselines. For fairness, we do not compare against methods that pretrain on large-scale external datasets like DD3D [149]. Our pretraining outperforms other counterparts, including BYOL [55], a contrastive learning algorithm for 2D tasks, and MonoDepth2 [54] and Occ-BEV [141] that consider geometric information but ignore semantics.

Cohere3D pretraining can also help improve models that only use a single current frame as input. For example, we transfer our pretrained backbone to BEVDet [85] which does not include temporal inputs. Tab. 3.2 shows the improvement brought by our pretraining when we finetune the model on both 10% and 100% of nuScenes training set.

3.4.3 HD Map Construction

Apart from detecting foreground objects, we examine our learned representation of the background through HD map construction task. We apply MapTR [118] and finetune it on the full nuScenes training set using our pretrained weights. Our pretraining improves the mAP metric from 50.3 to **51.2** in comparison to ImageNet pretraining. This improvement indicates that Cohere3D can also help the model to better understand the background scene such as the roadgraph.

Table 3.2: 3D object detection using BEVDet

Pretraining	10%		100%	
	NDS	**mAP**	**NDS**	**mAP**
ImageNet [85]	11.7	8.0	35.0	28.3
BYOL [55]	12.1	6.7	34.0	26.4
MonoDepth2 [54]	10.3	8.0	33.9	26.1
Occ-BEV [141]	13.6	9.1	36.0	28.4
Cohere3D (ours)	**16.2**	**10.8**	**36.1**	**28.6**

Table 3.3: Motion prediction using VAD-tiny

Pretraining	**Perception**		**Motion Prediction**	
	NDS	**mAP**	**EPA** (car)	**EPA** (pedestrian)
ImageNet [93]	39.0	27.0	59.8	29.0
Cohere3D (ours)	**41.4**	**27.9**	**63.1**	**33.1**

3.4.4 Motion Prediction

Learning a coherent instance representation over time not only helps to determine the current 3D object states, but also brings strong temporal reasoning ability, such as predicting future object motions. To verify this, we apply a vision-based prediction and planning model, VAD [93], as the task model for camera-based motion prediction task, where we initialize the model with our pretrained weights. The VAD-tiny model is finetuned on 100% nuScenes full training set. This model outputs both perception and end-to-end motion prediction results. The perception performance is measured by mAP and NDS, while the prediction performance is evaluated using the End-to-end Prediction Accuracy (EPA) [56] metric, as shown in Tab. 3.3. Consistent with Sec. 3.4.2, our pretraining leads to a performance gain for perception task. The end-to-end motion prediction also benefits from Cohere3D pretraining, which encourages instance-level coherence over long-term temporal frames. Without using any extra data, Cohere3D greatly lifts the motion prediction performance.

3.4.5 End-to-end Planning

Our pretraining benefits the downstream task models in predicting the current (Sec. 3.4.2) and future (Sec. 3.4.4) 3D states of the surrounding environments. This improved scene understanding facilitates the planning task. We finetune VAD-tiny [93] model for end-to-end planning on 100% nuScene training set. Planning performance is measured by the L_2

Table 3.4: End-to-end planning using VAD-tiny

Pretraining	L_2 (m)			Collision (%)		
	$1s$	$2s$	$3s$	$1s$	$2s$	$3s$
ImageNet [93]	0.46	0.76	1.12	0.21	0.35	0.58
Cohere3D (ours)	**0.28**	**0.52**	**0.84**	**0.11**	**0.26**	**0.56**

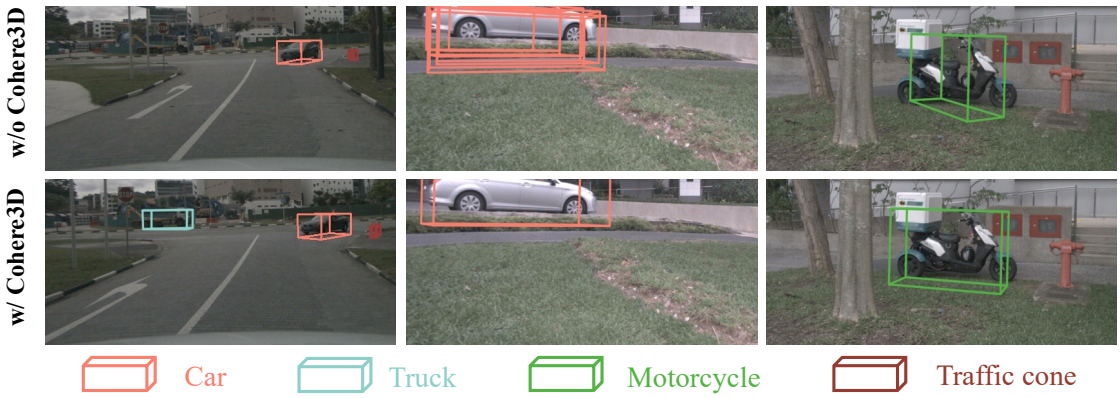


Figure 3.4: Qualitative comparison of SOLOFusion perception model without (left) vs with Cohere3D pretraining (right).

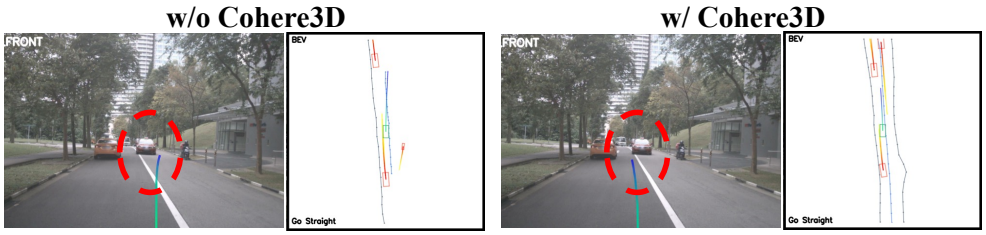


Figure 3.5: Qualitative comparison of VAD planning model without (left) vs with Cohere3D pretraining (right).

error and collision rates. Tab. 3.4 demonstrates that Cohere3D notably boosts planning performance by better estimating the current and future states of surroundings.

3.4.6 Analysis

Qualitative Results Fig. 3.4 and 3.5 show some qualitative improvement of Cohere3D pretraining for perception and planning. For perception, Cohere3D is effective at improving

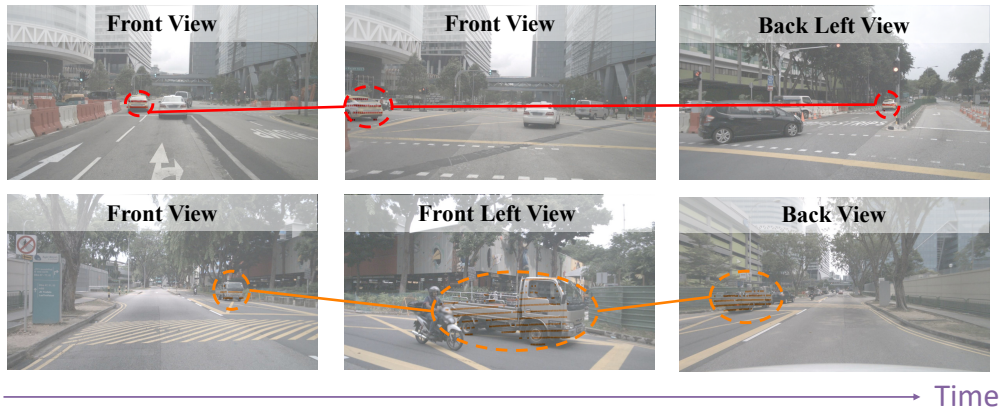


Figure 3.6: LiDAR-guided long-term correspondence. It reliably tracks each instance over long-term historical frames even if it is captured by different cameras at each time step.

recall rates, reducing false positives, and recovering object poses in 3D space. For planning, Cohere3D helps to reduce conflicts through a better understanding of surroundings.

LiDAR-Guided Long-term Correspondence Our pretraining algorithm emphasizes the long-term coherence of instance representation. The representation quality heavily depends on the accuracy of long-term correspondence constructed in the pretraining stage. Fig. 3.6 provides some examples of long-term correspondence guided by the LiDAR point clouds. Our method can retrieve reliable instance trajectories without any explicit human annotations over multiple historical frames for those dynamic instances. This provides a solid foundation for our temporal contrastive learning.

3.4.7 Ablation Studies

We justify our design by examining the role of each module during pretraining. We start from our key component, long-term temporal coherence over instance representations. Then, we discuss the role of each design decision in the contrastive learning framework. All the experiments in this section finetune the short-term version SOLOFusion [150] for 3D object detection task on the full nuScenes training set.

Long-term Temporal Coherence In the pretraining, we construct a memory bank for the historical instance features to encourage the temporal coherence of instance representations. We try different numbers of historical frames in Tab. 3.5a. Without temporal information, a clear performance drop is observed. This verifies that long-term temporal coherence is crucial for the downstream task. Results also show that more historical frames can lead to stronger representation and bring better performance to the downstream task.

Contrastive Learning Modules We also examine the role of other modules in our contrastive learning framework, as summarized in Tab. 3.5b. The results show that it is important to merge the ground-truth depth with the estimated depth in the view transformation

Table 3.5: Ablation study results

(a) Temporal coherence

Historical Frame	NDS	mAP
0	39.9	34.8
1	40.1	34.6
4	41.1	34.9
16	**41.3**	**35.1**

(b) Contrastive learning modules

Bkg. Points	Merged Depth	VT Mask	NDS	mAP
	✓	✓	39.8	35.0
✓		✓	38.3	34.8
✓	✓		40.2	34.7
✓	✓	✓	**41.3**	**35.1**

module of the target network (Sec. 3.3.4). Since the depth estimation from image inputs is fragile, the ground-truth depth from the LiDAR point clouds can notably reduce the positional ambiguity in the target network's BEV features. Secondly, the extraction of background point-level features (Sec. 3.3.3) makes full use of semantics in the input images outside of the foreground objects, which helps to distinguish the foreground from the background. Finally, extra dropout in the estimated depths for view transformation (Sec. 3.3.4) results in stronger local representation. Therefore, all the above designs contribute meaningfully to the final performance of our algorithm.

3.5 Conclusions

We propose a novel contrastive learning algorithm, Cohere3D, for unsupervised pretraining of vision-based 3D autonomous driving tasks. It encourages the long-term temporal coherence of instance features, effectively and robustly handling changes in observation viewpoints. By utilizing unlabeled LiDAR point clouds, Cohere3D learns long-term coherent instance-level representation without human annotations. Through comprehensive experiments, we show that our algorithm not only improves data efficiency, but also significantly boosts the final performance of downstream perception, mapping, prediction, and planning tasks.

Chapter 4

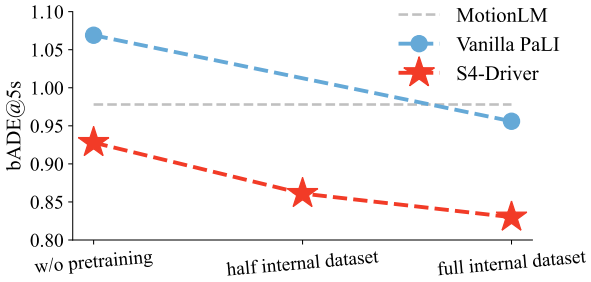
S4-Driver: Scalable Self-Supervised Driving Multimodal Large Language Model with Spatio-Temporal Visual Representation

This chapter continues the *Planning* part of the dissertation by moving from transferable representation learning to direct motion planning. Different from Chapter 3 that learns temporally stable features from scratch using raw driving logs, this chapter transfers out-of-domain knowledge into autonomous driving system. S4-Driver addresses this problem through a scalable self-supervised motion planning framework built on multimodal large language models and enhanced with a sparse spatio-temporal volume representation. By aggregating multi-view and multi-frame observations into a unified 3D structure, the model bridges large-scale visual pretraining with driving-specific spatial reasoning and directly predicts future trajectories without human-annotated planning labels.

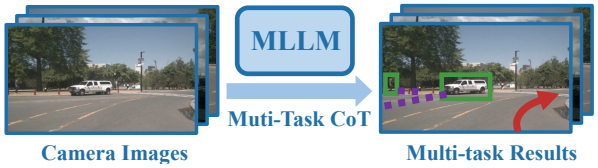
4.1 Introduction

The exploration of end-to-end autonomous driving dates back to the 1980s [158], where the motion planning model directly predicts control signals or future waypoints based on raw sensor inputs. Due to limited robustness, these early attempts struggle in complex urban situations. Recent neural network capability advances have led to renewed interest in this field [82, 93, 227], especially given the strong generalization ability of Multimodal Large Language Models (MLLMs) [28, 194, 146].

However, vanilla application of MLLMs to end-to-end motion planning can hardly exploit their strong visual understanding and reasoning capabilities given the significant difference between motion planning and MLLM pretraining tasks, resulting in inferior planning performance [199, 241]. To narrow this gap, as shown in Fig. 4.1b, previous methods resort to



(a) Pretraining data scaling-up of S4-Driver.



(b) Multi-task learning framework.



(c) Self-supervised framework.

Figure 4.1: S4-Driver scaling up with abundant unlabeled driving logs (4.1a). Different from past efforts that improve MLLM planning via multi-task learning and CoT reasoning (4.1b), S4-Driver focuses on self-supervised motion planning (4.1c) without human annotations. While a vanilla PaLI approach with self-supervised training achieves good performance, the full S4-Driver with spatio-temporal visual representation achieves the best performance and scalability (4.1a).

either *multi-task learning*, which incorporates diverse perception and prediction tasks into training and inference, or *supervised perception pretraining*, which utilizes the pretrained autonomous driving perception models as visual tokenizers. However, fine-grained human annotations serve as the bottleneck for both strategies. In contrast, pure self-supervised approaches, while capable of learning directly from sensor inputs and utilizing the abundant unlabeled driving logs, often under-perform the state of the art.

To challenge this status-quo, we aim to boost the performance of self-supervised motion planning orthogonal to prior work (Fig. 4.1c). It only requires raw sensor data and ego-vehicle trajectories for self-supervised model training, without the need of expensive human labels. First of all, we identify the following two main obstacles in this direction:

- **Suboptimal representation format.** MLLMs [28, 29] are typically designed for tasks in the 2D image plane. This image space representation limits their 3D reasoning ability from the combination of multi-view images.
- **Limited data scale.** Although it is one of the most widely adopted benchmarks for end-to-end planning, nuScenes [18] only contains fewer than 1k sequences and lacks driving behavior diversities. This limited scales leads to severe over-fitting issues in the finetuning of MLLMs with billion-parameter size.

In this paper, we handle the above challenges to better explore the potential of MLLMs. Firstly, we propose **S4-Driver**, a simple yet effective Scalable Self-Supervised motion plan-

ning method with **S**patio-temporal visual representation. Built upon a general multimodal large language model [29, 28] , we directly predict ego-vehicle waypoints from camera images, eliminating the need for intermediate perception and prediction tasks, and thus facilitating the scale-up of model pretraining leveraging massive unannotated driving logs (Fig. 4.1a). To address the obstacle of suboptimal visual format, we propose a novel sparse volume representation that enables aggregating visual information from multi-view and multi-frame images. Moreover, this spatio-temporal representation is designed to work with frozen features from the perspective-view vision encoder. It boosts 3D spatio-temporal reasoning capabilities of the model on motion planning, and seamlessly preserves world knowledge in the pretrained visual embeddings of MLLMs. Secondly, to rigorously evaluate our method with sufficient training data, we also utilize large-scale WOMD-Planning-ADE benchmark [47] with in-house camera sensor data besides the popular nuScenes dataset [18]. WOMD-Planning-ADE is approximately 100× larger than nuScenes, so it serves as a more comprehensive benchmark to evaluate model performance in a scalable manner.

In summary, our contributions include:

- We advocate for a simple self-supervised end-to-end motion planning with MLLMs, solely relying on ego-vehicle trajectory supervision. This approach simplifies the system architecture and facilitates data scaling for training.
- We propose a novel sparse volume representation to effectively aggregate multi-view and multi-frame visual information, enhancing the 3D spatio-temporal reasoning capabilities of the MLLM for motion planning.
- Our algorithm achieves state-of-the-art performance on both nuScenes and WOMD-Planning-ADE benchmarks. It also demonstrate great scalability with pretraining.

4.2 Related Works

Multimodal Large Language Models (MLLMs). MLLMs with both language and images modalities have garnered significant attention in recent years [162, 3, 106, 1, 194]. Prior research has focused on integrating powerful Large Language Models (LLMs) [265, 35, 265] with advanced image encoders [49, 162, 260] such as LLaVA [124], PaLI [29, 28], PaliGemma [9], and InstructBLIP [38]. Through instruction tuning or multi-modal fine-tuning, these models demonstrate impressive performance in multimodal understanding and reasoning. Current developments are leveraging increasingly larger multimodal datasets to further enhance their capabilities in complex perception and generalization tasks [4, 146, 230]. However, despite their strengths, these models have shown limitations in 3D spatial reasoning [75, 34, 137], posing challenges for their application in autonomous driving.

End-to-end Autonomous Driving. While the key final step of autonomous driving is planning, traditional systems are often assembled with multiple modules for perception [128, 236, 118] and prediction [203, 143, 266] tasks. To mitigate the information loss and error

Table 4.1: Comparisons between WOMD-Planning-ADE and existing counterparts. WOMD-Planning-ADE has a significantly larger size and more comprehensive behavior-wise metrics.

Benchmark	Size		Data Format		Metrics	
	Sequence	Length	Frequency	Planning horizon	Sample-wise	Behavior-wise
BDD-X [98]	7k	77 h	10 Hz	control signal	✓	✗
nuScenes [18]	1k	5.5 h	2 Hz	3s	✓	✗
WOMD-Planning-ADE	103k	574 h	10 Hz	5s	✓	✓

accumulation across modules, end-to-end driving systems [21, 80, 82, 93, 267, 114] utilize a unified model to predict the ego-vehicle future waypoints or control signals directly from raw sensor inputs. While prioritizing planning, these systems typically still incorporate perception and prediction, requiring explicit supervision for each. Although some prior works [177, 36, 228] have explored pure motion planning without any intermediate tasks, they struggle in complex urban scenarios due to modeling capability limitations.

MLLMs for Driving. The superior reasoning and generalization abilities of large-scale models are desirable properties for autonomous driving. Some efforts transform the driving scenario into large language model textual prompts [178, 134, 26] or directly process camera images with vision-language models [241, 179, 44]. However, their potential is limited to the small scale of existing benchmarks [18], which only allow partial finetuning [78]. Meanwhile, closed-loop simulators [45] struggle to provide realistic sensor data for end-to-end tasks. Consequently, multi-task joint finetuning [183, 199, 241] or chain-of-though (CoT) [225] inference [133, 136, 134] are widely adopted to simplify the reasoning despite the extra annotation requirements brought by perception and prediction modules. Alternatively, some work [197, 44] integrates pretrained perception models [113] to extract BEV features and sends to language models as vision tokens. Recently, EMMA [88] leverages powerful Gemini [194] for self-supervised motion planning. Additionally, they develop a mixture of training tasks, including motion planning, 3D object detection, and road graph estimation as well as additional reasoning process for trajectory generation. In contrast, our work focuses on the enhancement of self-supervised motion planning without extra human labels.

4.3 Waymo Open Motion Dataset for Planning

For the large-scale training and evaluation of planning algorithms with large models, we design a WOMD-Planning-ADE benchmark based on WOMD dataset [47], and compare with the existing datasets in Tab. 4.1.

Overview. It contains $103k$ real-world driving scenarios, covering diverse urban and sub-urban scenes. They are further divided into $9s$ samples with $1s$ history and $8s$ future.

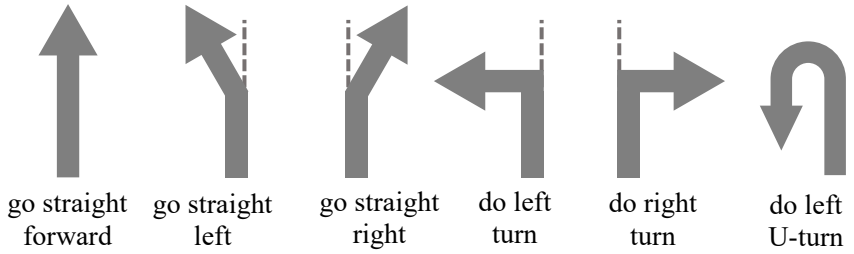
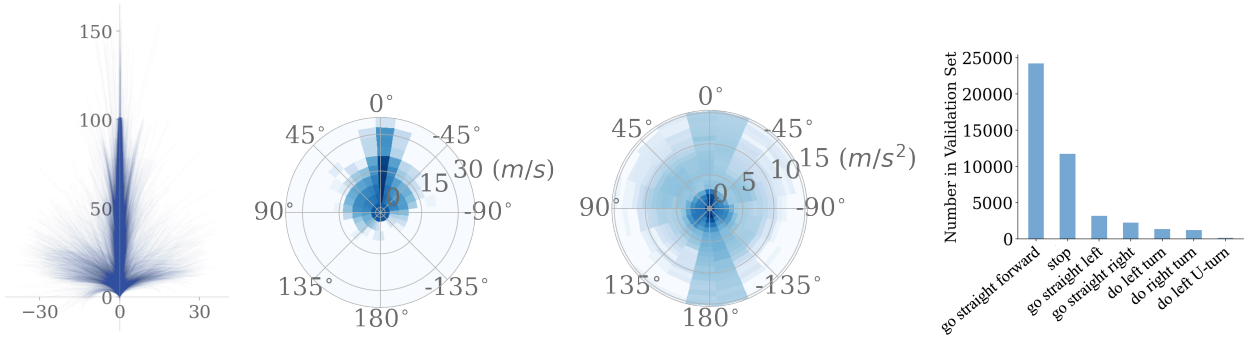


Figure 4.2: High-level behavior commands for motion planning.



(a) Trajectory (b) Velocity (c) Acceleration (d) Behavior

Figure 4.3: Statistical distributions of WOMD-Planning-ADE. We show the distributions of ego-vehicle trajectories, velocities, accelerations and each behavior scenario sizes in the validation set.

For end-to-end planning evaluation, besides the ego-vehicle trajectories in each sample as groundtruth, we also have the following key items in the dataset:

(In-house) Camera Data. Most end-to-end planning methods [93, 183, 241, 199] relies on the camera images as model inputs. In our dataset, each frame contains images captured by eight multi-view cameras.

High-level Behavior Commands. Like navigation systems for humans (*e.g.* Google Map), the end-to-end planning system also requires navigation signals to indicate the direction to go. We consider six high-level behavior commands (Fig. 4.2), namely *go straight forward, go straight left, go straight right, do left turn, do right turn, do left U-turn.* They can cover diverse driving situations in real world, *e.g. "go straight right"* describes the case to leave the high-way. Instead of simply considering the last step position [21, 82], we decide the behavior commands based on long-term future trajectories, which can handle the low-speed or stopping situations. Details of this heuristic rule are included in the supplementary materials.

Evaluation Metrics. The imbalance of data distribution is inevitable in real driving scenar-

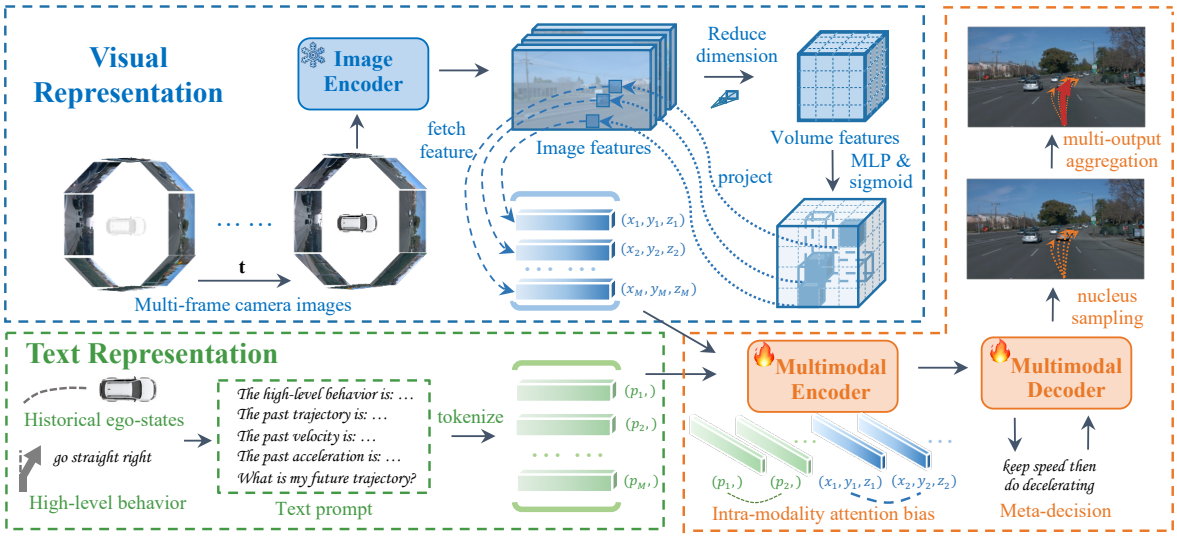


Figure 4.4: Overview of our proposed S4-Driver algorithm. We enhance the PaLI model for motion planning by incorporating meta-decision, spatio-temporal visual representation, and multi-decoding aggregation.

ios. For example, in the WOMD-Planning-ADE benchmark, moving straight and stopping account for more than 70% of all the samples, as reflected by Fig. 4.3. In this case, we argue that current widely used sample-wise average displacement error and collision rates cannot comprehensively reflect the performance of motion planning algorithms since challenging but less frequent behaviors, like turning, are overwhelmed by simple straightforward moving scenarios. To this end, we propose novel behavior-wise metrics similar to the mAP metric in prediction [47]. For example, we denote the behavior-wise average displacement error as *bADE* defined as follows.

$$bADE = \frac{1}{|\mathcal{B}|} \sum_{b \in \mathcal{B}} ADE_b \tag{4.1}$$

where ADE_b is the ADE metric for a specific behavior. Concretely, $|\mathcal{B}| = 7$ behaviors are considered – including the six high-level commands (Fig. 4.2) and an additional *stop* behavior[1]. These behavior-wise metrics make up for the shortcomings of previous sample-wise metrics by paying more attention to those scarce but safety-critical behaviors.

4.4 Enhancing MLLM for Motion Planning

In this section, we adapt a pretrained MLLM for the end-to-end motion planning task. Without loss of generality, we leverage PaLI [28]. However, these techniques are also compatible

[1]We do not include “stop" in the high-level behavior commands since navigation system cannot provide it and this may leak future information like traffic lights. Details are explained in supplementary materials.

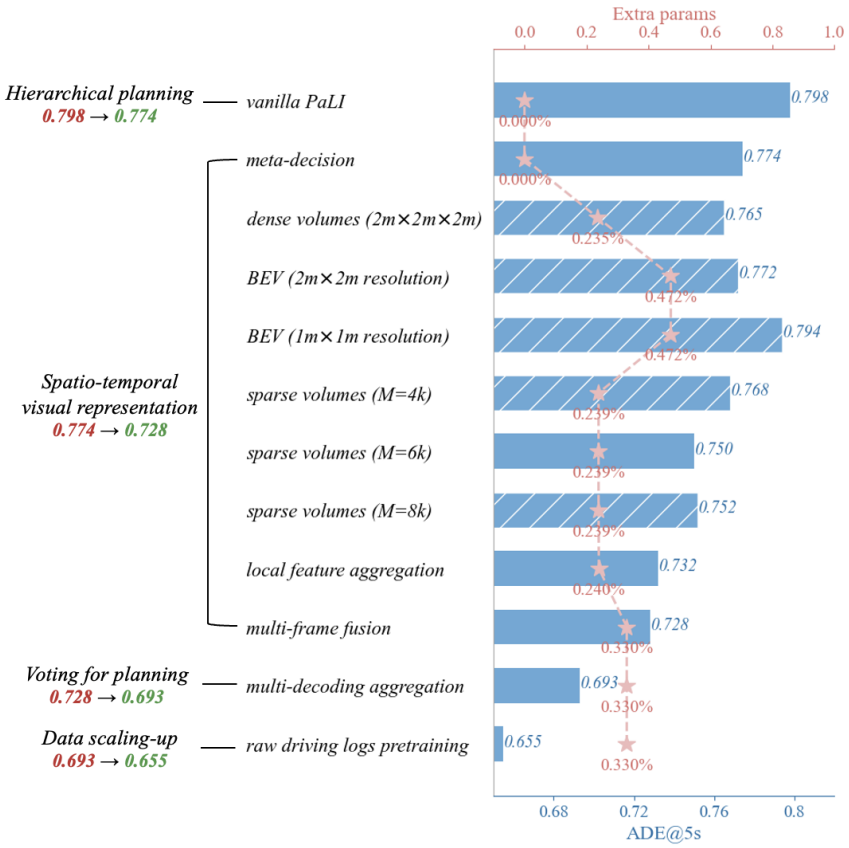


Figure 4.5: A roadmap for enhancing MLLM for planning. The techniques are adopted step-by-step from top to bottom, while shadow items are not adopted in the subsequent steps. We show the performance on WOMD-Planning-ADE (Sec. 4.3) after including each module.

with other more powerful MLLMs. We introduce a set of design improvements to the MLLM setup and demonstrate correspondingly improved performance on WOMD-Planning-ADE (Fig. 4.5).

4.4.1 Vanilla PaLI as Planner

End-to-end motion planning models determine the future trajectory $\mathcal{O}_{T_f}$ of ego-vehicle based on multi-view camera images $\mathcal{C}$ and high-level behavior command b. The future trajectory consists of the ego-vehicle's location at each future time step in the BEV coordinate, *i.e.* $\mathcal{O}_{T_f} = [(x_t, y_t)]_{t=1}^{T_f}$. The high-level behavior command is necessary to guide the driving direction, similar to a navigation system. Additionally, as verified in prior works [259, 114], the historical states of ego-vehicle $\mathcal{H}_{T_h}$ are also critical for a smooth and feasible planning result, where we consider the historical locations, velocities, and accelerations as $\mathcal{H}_{T_h} =$

$[\mathbf{l}_t, \mathbf{v}_t, \mathbf{a}_t]_{t=-1}^{-T_h}$, *i.e.*

$$\mathcal{O}_{T_f} = P(\mathcal{C}, \mathcal{H}_{T_h}, b) \tag{4.2}$$

where $P(\cdot)$ is the planning model. We provide both historical ego states and high-level commands to the model as text prompts. Locations, velocities, and accelerations are represented directly as floating numbers with two decimals. The predicted future trajectories are then extracted from the decoded text outputs of the model. Example prompts and targets are provided in the supplementary materials.

Without preceding perception and prediction tasks, the vanilla PaLI, fine-tuned in a self-supervised manner, achieves reasonable but not ideal performance in motion planning (Fig. 4.5).

4.4.2 Hierarchical Planning with Meta-Decision

Directly outputting a future trajectory without any reasoning is challenging for MLLMs. To address this, we apply a coarse-to-fine approach, inspired by Chain-of-Thought (CoT) [225]. It adopts a hierarchical planning method [89], going from from semantic decision to numeric planning. We prompt our model to provide a high-level estimation of the future acceleration state D first, effectively decomposing the motion planning task into two steps:

$$\mathcal{D} = P_{s1}(\mathcal{C}, \mathcal{H}_{T_h}, b), \quad \mathcal{O}_{T_f} = P_{s2}(\mathcal{C}, \mathcal{H}_{T_h}, b; \mathcal{D}) \tag{4.3}$$

We define D to encompass four meta-decisions: *keep stationary, keep speed, accelerate* and *decelerate*. Notably, unlike previous CoT applications in VLM-based planning that require human annotation for training, such as the scene analysis in DriveVLM [199], we incorporate these meta-decisions as a "free lunch" to simplify the motion planning process without requiring any additional annotations. The ground-truth decisions are generated by heuristic rules based on the future ego velocity and acceleration (details in the supplementary materials). Fig. 4.5 shows this simple design brings non-trivial improvements in planning performance.

4.4.3 Scene Representation in 3D Space

High-quality motion planning requires a robust understanding of the surrounding 3D scene, including both static and dynamic elements. While traditionally achieved through separate perception and prediction modules, our self-supervised end-to-end framework relies on the MLLM to implicitly learn this understanding without explicit supervision. However, despite strong 2D reasoning capabilities, MLLMs often struggle with 3D spatial reasoning due to limitations in their perspective view representation and the lack of depth-related tasks in pretraining [24, 131].

3D Visual Representation with Dense Volumes

To overcome the above limitations, we draw inspiration from previous successful perception tasks [113, 128] and adopt a 3D volume representation. The multi-view feature maps $\mathcal{F}_{2D} =$

$\{\mathbf{f}_{2D}^v\}_{v=1}^V (\mathbf{f}_{2D}^v \in \mathbb{R}^{H\times W\times C})$ are extracted by visual encoder of MLLM, where V is the number of views. We construct a 3D feature volume $\mathbf{f}_{3D}^{vol} \in \mathbb{R}^{X\times Y\times Z\times C}$ centered at the ego-vehicle based on the multi-view image features. To avoid introducing complex modules that might disrupt the pretrained MLLM and misalign visual features with the subsequent multimodal encoder-decoder, we employ a lightweight projection method, similar to SimpleBEV [64]. Specifically, for each voxel in the 3D volume, we project its (x, y, z) coordinates to each perspective view v, obtaining corresponding 2D coordiantes (u_v, v_v). We then bilinearly sample the local features from each view at these projected locations. Finally, the voxel's feature representation is computed as the average of the local semantic features from all views where the voxel projects within the image bounds. This process effectively incorporates 3D spatial information while maintaining compatibility with the pretrained MLLM.

$$\mathbf{f}_{3D,(x,y,z)}^{vol} = \mathbf{f}_{(x,y,z)}^{sem} + \text{PosEmb}(x, y, z) \tag{4.4}$$

$$\mathbf{f}_{(x,y,z)}^{sem} = \text{Avg}\left(\{\text{Bilinear}\left(\mathbf{f}_{2D}^v; (u_v, v_v)\right)\}_v\right) \tag{4.5}$$

This simple and efficient projection strategy ensures that the 3D volume features $\mathbf{f}_{3D}^{vol}$ share a similar distribution with the original multi-view features $\mathcal{F}_{2D}$. This similarity facilitates seamless integration with the subsequent multimodal encoder-decoder of the MLLM.

As shown in Fig. 4.5, this 3D volume representation leads to some improvement in motion planning performance. Alternatively, we find that reducing the Z-axis using a fully connected layer to obtain a BEV representation yields slightly worse performance since this reduction operation may introduce ambiguity to the scene representation.

Sparse Volume Representation

While the 3D volume representation effectively captures spatial information, inherently most of the surrounding 3D space is empty. Moreover, for motion planning, detailed information about objects distant from the road, such as buildings and trees, is less critical. Based on this observation, we propose a sparse volume representation to reduce the number of voxels, enabling higher resolution given memory constraints and improving efficiency.

To determine useful volumes in each scene based on their positions and semantics, we define a gate value $g_{(x,y,z)}^{vol} \in (0, 1)$ for each volume at coordinates (x, y, z). To obtain this gate, starting from multi-view image features $\{\mathbf{f}_{2D}^v\}_{v=1}^V$, we reduce its dimension through a fully connect (FC) layer as

$$\mathbf{f}_{2D}^{gate,v} = \text{FC}(\mathbf{f}_{2D}^v), \quad \mathbf{f}_{2D}^{gate,v} \in \mathbb{R}^{H\times W\times C'}, C' \ll C. \tag{4.6}$$

Then we construct a reduced-dimension volume feature $\mathbf{f}_{3D}^{gate} \in \mathbb{R}^{X\times Y\times Z\times C'}$ from $\{\mathbf{f}_{2D}^{gate,v}\}_{v=1}^V$ same as Eq. 4.4. A smaller channel number allows for larger volume resolution and is sufficient to indicate whether a volume is related to motion planning . Afterward, the gate is derived from $\mathbf{f}_{3D}^{gate}$ through a small MLP module as follows.

$$g_{(x,y,z)}^{vol} = \text{sigmoid}\left(\text{MLP}(\mathbf{f}_{3D,(x,y,z)}^{gate})\right) \tag{4.7}$$

Thus, we can easily select M volumes ($M \ll X \cdot Y \cdot Z$) with largest gate values with coordinates $\{(x_i, y_i, z_i)\}_{i=1}^{M}$.

As we do not have access to ground-truth occupancy state, we propose learning the gate values implicitly. We assume that regions with small gate values should be empty or irrelevant to planning. For these vacant space, we assign a learnable feature $\mathbf{f}_{vac} \in \mathbb{R}^C$. We expect the model can learn the gate value via the trade-off between the semantic features and this vacant feature at each 3D position, so we fetch features for selected sparse volumes as

$$\mathbf{f}^{sparse}_{3D,(x,y,z)} = \hat{\mathbf{f}}^{sparse}_{3D,(x,y,z)} + \text{PosEmb}(x, y, z) \tag{4.8}$$

$$\hat{\mathbf{f}}^{sparse}_{3D,(x,y,z)} = g^{vol}_{(x,y,z)} \cdot \mathbf{f}^{sem}_{(x,y,z)} + (1 - g^{vol}_{(x,y,z)}) \cdot \mathbf{f}_{vac} \tag{4.9}$$

where $\mathbf{f}^{sem}_{(x,y,z)}$ is same as Eq. 4.5 and $(x, y, z) \in \{(x_i, y_i, z_i)\}_{i=1}^{M}$ are selected volumes with large gate values. When the sparse volume features $\mathbf{f}^{sparse}_{3D,(x,y,z)}$ are fed into the subsequent multimodal encoder, they notably boost the planning performance (Fig. 4.5) by simultaneously providing explicit 3D spatial cues of surrounding scenes and concentrate on the important regions adaptively.

Local Feature Aggregation in 3D Space

Due to the lack of depth information, the lifting process in Eq. 4.4 or 4.8 results in duplicate volume features along each camera ray. This spatial ambiguity can be mitigated by 3D local operations like convolutions [64] or deformable attention [113] as demonstrated in previous work. However, in our MLLM framework, it is less economical to insert extra local operations. Instead, we inject some relative position bias into the existing multimodal encoder by tailoring the self-attention as follows.

$$\text{Attn}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d}} - b(D)\right)V \tag{4.10}$$

Given M sparse volumes with coordinates $\{(x_i, y_i, z_i)\}_{i=1}^{M}$, the distance matrix $D \in \mathbb{R}^{M \times M \times 3}$ is the distance along xyz-axes between each pair of sparse volumes, *i.e.* $D_{ij} = (x_j - x_i, y_j - y_i, z_j - z_i)$. The bias is computed with the function $b(\Delta x, \Delta y, \Delta z) = b_x(\Delta x) + b_y(\Delta y) + b_z(\Delta z)$, where $\Delta x, \Delta y, \Delta z$ are divided into several bins and mapped to a learnable bias value for each bin through $b_x(\cdot), b_y(\cdot), b_z(\cdot)$. We also apply a separate bias between the text tokens with 1D position $\{p_i\}$.

This relative position bias elegantly inserts local inductive bias to pretrained global self-attention modules with minimal additional costs. It leads to a notable performance gain in Fig. 4.5 by facilitating the aggregation of local information in 3D space, enhancing scene understanding and spatial reasoning.

4.4.4 Multi-frame Temporal Fusion

Multi-frame input combination helps to compensate for the lack of depth cues in camera images. We extend our sparse volume representation to aggregate mult-frame temporal information by incorporating T historical frames with an interval of $0.5s$. Given images from the total $T+1$ frames, Eq. 4.6 is applied to each frame separately to obtain the multi-view image features $\mathbf{f}_{2D,t}^{gate,v}, t = 0, -1, \ldots, -T, v = 1, \ldots, V$. After ego-motion compensation, we construct a gate feature volume $\mathbf{f}_{3D,t}^{gate}$ based on the current ego-vehicle coordinates and per-frame image features. Multi-frame volume features are concatenated along the channel dimension to generate gate value.

$$g_{(x,y,z)}^{vol} = \text{sigmoid}\left(\text{MLP}\left(\text{Concat}\left([\mathbf{f}_{3D,t,(x,y,z)}^{gate}]_{t=0}^{-T}\right)\right)\right) \tag{4.11}$$

Similar with Sec. 4.4.3, we select M volumes based on the gate value. Following Eq. 4.8, the volume features are fetched from each frame image feature map separately as $\mathbf{f}_{3D,t,(x,y,z)}^{sparse}, t = 0, \ldots, -T$, which are fused with an FC-layer as the sparse volume feature with temporal awareness.

$$\mathbf{f}_{3D,(x,y,z)}^{sparse} = \text{FC}\left(\text{Concat}\left(\left[\mathbf{f}_{3D,t,(x,y,z)}^{sparse}\right]_{t=0}^{-T}\right)\right) \tag{4.12}$$

In Fig. 4.5, temporal fusion helps to improve motion planning performance by facilitating the environment understanding.

4.4.5 Voting for Planning via Multi-Decoding

MLLMs are prone to assigning high confidence to the simple future behaviors in motion planning, *e.g.*, keeping stationary. To relieve this bias, we aggregate multiple outputs and use voting to obtain the final planning output. We resort to nucleus sampling [73] to produce multiple future trajectories for the ego-vehicle, denoted as $\{\mathcal{O}_{T_f}^k\}_{k=1}^K$. They are simply averaged to produce a unique planning result as follows: $\mathcal{O}_{T_t} = [(x_t, y_t)]_{t=1}^{T_f}$.

$$(x_t, y_t) = \left(\frac{1}{K}\sum_{k=1}^{K} x_t^k, \frac{1}{K}\sum_{k=1}^{K} y_t^k\right) \tag{4.13}$$

This unweighted average mitigates the MLLM's bias towards simple behaviors. Fig. 4.5 demonstrates the notable performance gain with this simple multi-decoding aggregation approach.

4.4.6 Scaling to Large-scale Raw Driving Logs

The self-supervised training allows our proposed S4-Driver to scale to large-scale driving logs amounts without requiring human annotations. To exploit the potential of MLLM-based planners, we pretrain the model on our internal dataset. Results in Fig. 4.5 demonstrate that S4-Driver achieves notable performance gains in challenging tail behaviors due to large-scale pretraining.

4.5 Experiments

4.5.1 Implementation Details

Model and finetuning. Unless otherwise specified, we construct our model from a pretrained PaLI3-5B model [28], which includes a ViT-G (2B) vision encoder [46] and multimodal encoder-decoder (3B) [193]. Our default setting freezes the ViT encoder and finetune only the inserted modules and the multimodal encoder-decoder. Finetuning the model on the WOMD-Planning-ADE dataset requires approximately 2.5 days on 128 Google Cloud TPU v4 using a batch size of 256 and a learning rate of 3e-3.
Datasets. We evaluate S4-Driver on both WOMD-Planning-ADE (Sec. 4.3) and nuScenes [18] benchmarks. Given the limited size of nuScenes, which is insufficient for training a multimodal large language model, we also attempt to finetune the model on nuScenes using weights pretrained on WOMD-Planning-ADE.

Table 4.2: NuScenes benchmark results. Most previous works require human annotations as additional supervision. S4-Driver* is pretrained on WOMD-Planning-ADE, while many prior MLLM-based methods also utilize extra data and labels due to the limited scale of nuScenes.

Methods	**Human Annotations**		**Extra Data**		$L_2(m)$			
	perception	prediction	data	labels	1.0s	2.0s	3.0s	$Avg_{1,2,3s}$
Multi-task end-to-end approaches								
UniAD [82]	✓	✓	✗	✗	0.42	0.64	0.91	0.66
VAD [93]	✓	✓	✗	✗	0.17	0.34	0.60	0.37
PARA-Drive [227]	✓	✓	✗	✗	0.25	0.46	0.74	0.48
Multimodal large language model approaches								
GPT-Driver [134]	✓	✓	✗	✗	0.20	0.40	0.70	0.44
DriveVLM [199]	✓	✓	✓	✓	0.18	0.34	0.68	0.40
OmniDrive [213]	✓	✓	✓	✓	0.14	0.29	0.55	0.33
Self-supervised approaches								
DriveVLM (w/o CoT) [199]	✗	✗	✓	✗	0.19	0.41	0.89	0.49
S4-Driver (ours)	✗	✗	✗	✗	0.16	0.34	0.63	0.38
S4-Driver* (ours)	✗	✗	✓	✗	**0.13**	**0.28**	**0.51**	**0.31**

4.5.2 Main Results and Comparison

NuScenes dataset. To widely compare with existing end-to-end planning methods in the community, we evaluate S4-Driver on nuScenes dataset [18]. Tab. 4.2 show that S4-

Table 4.3: WOMD-Planning-ADE benchmark results. "*" denotes methods pretrained on the internal dataset.

Methods	Sample-wise Metrics			Behavior-wise Metrics		
	ADE@1s	ADE@3s	ADE@5s	bADE@1s	bADE@3s	bADE@5s
Vanilla PaLI	0.034	0.277	0.798	0.051	0.392	1.069
S4-Driver (ours)	0.031	0.247	0.693	0.049	0.350	0.928
S4-Driver* (ours)	**0.029**	**0.233**	**0.655**	**0.044**	**0.313**	**0.830**
Models that utilize high-quality objects, tracks, and roadgraph as inputs rather than raw camera images.						
MotionLM [177]	0.045	0.266	0.697	0.066	0.388	0.978

Driver notably outperforms all the previous algorithms despite its self-supervised training format. Unlike existing works, S4-Driver does not require any perception pretraining or human annotations. This self-supervision nature enables S4-Driver to utilize all the accessible raw trajectory data, whereas these easily collectable raw driving logs offer limited benefit to other algorithms without a labeling process.

WOMD-Planning-ADE benchmark. In Tab. 4.3, we primarily compare S4-Driver with the vanilla PaLI3-5B baseline (Sec. 4.4.1) and modular algorithm MotionLM [177]. Compared with vanilla PaLI3-5B, a notable gap is witnessed for both sample-wise and behavior-wise metrics. For reference, we also adapt a latest motion prediction algorithms MotionLM [177] (enhanced internally reproduced version) for planning task by only predicting the future trajectories of ego-vehicle and injecting the high-level command to the model. Because it utilizes auto-labelled high-quality objects, tracklets, and roadgraph signals as model inputs [159], a direct comparison with our end-to-end approaches is not equitable. However, as shown in Tab. 4.3, S4-Driver still achieves favorable performance compared to MotionLM, particularly in behavior-wise metrics even if S4-Driver uses only raw camera images as inputs.

4.5.3 Analysis

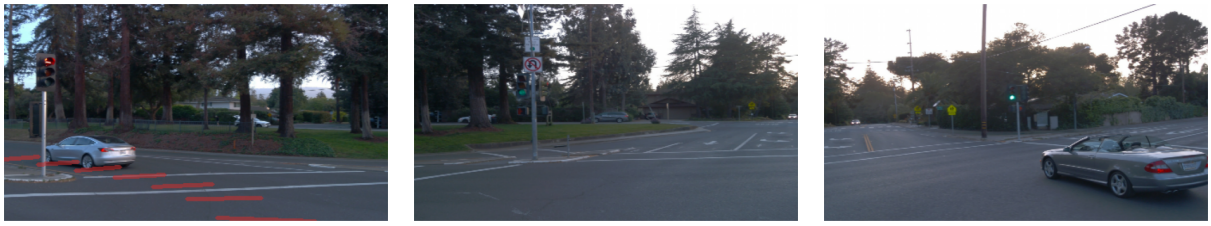
Qualitative results. Fig. 4.6 visualizes planning results in diverse scenarios. Our proposed S4-Driver can determine the future ego-behavior based on the traffic lights and road lane. It can also complete challenging behaviors and deal with different lighting conditions. We will include more visualizations in supplementary materials.

Meta-decision reliability. Fig. 4.7 demonstrates the accuracy of meta-decision prediction (Sec. 4.4.2) on WOMD-Planning-ADE validation set. Across all behaviors, the model provides reliable meta-decision estimations. Without any human annotations, this preliminary prediction can simplify the derivation of numeric motion planning.

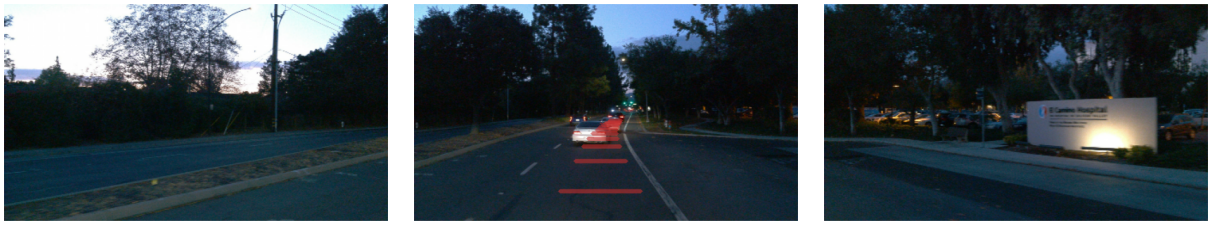
Distribution of sparse volumes. We visualize the distribution of self-supervised learned sparse volumes along x-axis and y-axis over WOMD-Planning-ADE validation set in Fig. 4.8.



(a) The ego-vehicle decelerates to wait for the red lights.



(b) The ego-vehicle conducts a U-turn at the intersection.



(c) The ego-vehicle keeps the lane under bad lighting condition.

Figure 4.6: Qualitative results of motion planning. We show the *front left, front, front right* cameras for each case.

From back to front, the sparse volumes concentrate on the close front region. From left to right, the sparse volumes cover all the regions, since there are turning scenarios, but most volumes focus on the middle region. These distributions are consistent with human driving experiences.

4.5.4 Ablation Studies

This section presents experiments conducted on the WOMD planning benchmark to clarify the roles of specific modules within our S4-Driver algorithm. More ablation studies are available in the supplementary material.

MLLM inputs. In Tab. 4.5, we analyze the roles of camera images and historical ego-states separately using the vanilla PaLI model. Consistent with [259, 114], the importance of ego-states is acknowledged. Meanwhile, different from their observations on nuScenes benchmark, camera inputs also make great differences on WOMD-Planning-ADE. We assume that WOMD-Planning-ADE covers more diverse driving scenarios including many severe changes

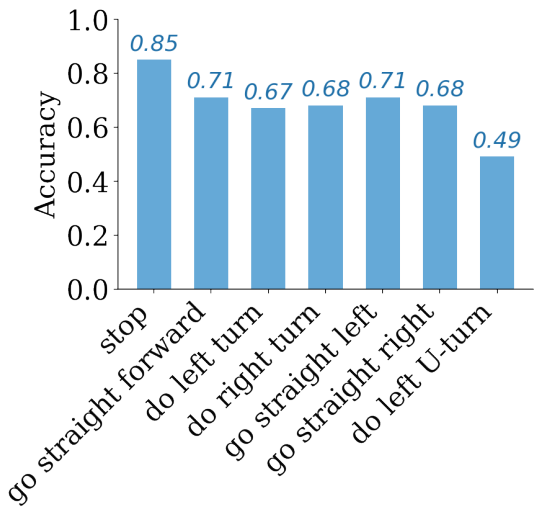


Figure 4.7: Meta-decision prediction accuracy for each behavior.

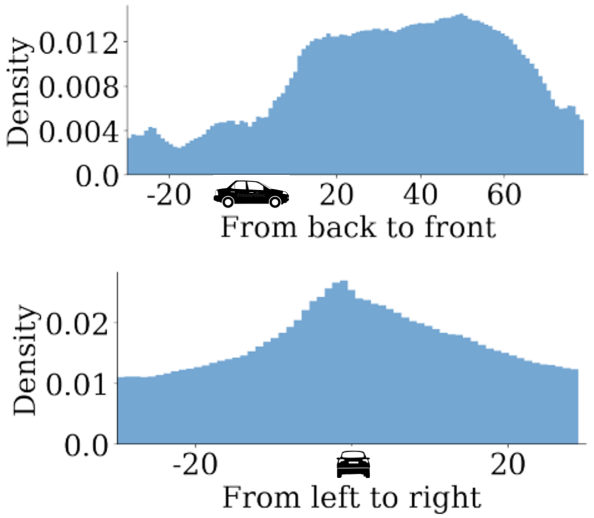


Figure 4.8: Distribution of sparse volumes along x- and y-axes.

Table 4.4: Ablation studies on planning performance scalabity of our methods with regard to model and data sizes.

MLLM	WOMD-Planning-ADE Subset (20k)		Full WOMD-Planning-ADE (400k)	
	ADE@5s	bADE@5s	ADE@5s	bADE@5s
PaLI2-3B [29]	1.241	1.846	1.035	1.522
PaLI3-5B [28]	**1.075**	**1.495**	**0.693**	**0.928**

of speed and directions, which attaches importance to the sensor data. This also showcases the advantages of WOMD-Planning-ADE for comprehensive evaluation. Tab. 4.5 also shows that random initialized model cannot converge without MLLM pretraining. Despite different domains, S4-Driver can profit from the strong reasoning ability from large-scale MLLM pretraining on general tasks.

MLLM capability. In addition to the PaLI3-5B [28] used in other parts, we also tailor PaLI2-3B [29] for motion planning with the same techniques. PaLI2-3B is weaker in parameter numbers (3B vs. 5B), architecture, pretraining manner, and input resolution (224 vs. 448). As in Tab. 4.4, S4-Driver based on PaLI2-3B demonstrates much inferior performance compared to that based on PaLI3-5B. We conduct experiments on two different training data scales on WOMD-Planning-ADE, *i.e.* 20k (nuScenes scale) vs. 400k (full WOMD-Planning-ADE). The gap is particularly large given enough training data. It also justifies the necessity to conduct experiments on large-scale dataset, which can fully unleash the potential of powerful MLLMs.

Table 4.5: Ablation studies on MLLM inputs.

Method	**ADE@5s**	**bADE@5s**
vanilla PaLI	**0.798**	**1.069**
w/o historical states	2.009	2.250
w/o camera images	1.081	1.448
w/o MLLM pretraining	*fail to converge*	

Table 4.6: Ablation studies on sparse volume resolutions.

Resolution	**ADE@5s**	**bADE@5s**
$2m \times 2m \times 2m$	0.798	1.201
$1m \times 1m \times 2m$	**0.750**	**1.005**
$1m \times 1m \times 1m$	0.765	1.051

Sparse volume resolution. Tab. 4.6 shows the results with different resolution for sparse volumes with same number of sparse volumes ($M = 6000$). In line with Fig. 4.5, low resolution leads to relatively worse performance since it limits the precision of 3D spatial reasoning. Interestingly, higher resolution along z-axis does not definitely improve the model performance since the motion planning mainly works on the xy-plane and too low sparsity ratio tends to make the optimization unstable.

4.6 Conclusion and Future Work

This paper introduces S4-Driver, a scalable self-supervised motion planning framework for autonomous driving that utilizes Multimodal Large Language Models (MLLMs). To enhance 3D reasoning in MLLMs, we propose a novel sparse volume representation, which enables improved spatio-temporal reasoning through the aggregation of multi-view and multi-frame image inputs. Additionally, we design behavior-wise metrics for comprehensive evaluation on the large-scale WOMD-Planning-ADE benchmark. S4-Driver achieves state-of-the-art performance on both nuScenes and WOMD-Planning-ADE benchmarks, without requiring any human annotations. This demonstrates the potential of self-supervised learning for end-to-end autonomous driving. Future work will involve applying our methods to other powerful MLLM architectures. Our contributions are orthogonal to previous works, such as those that explore multi-task learning or CoT reasoning [183, 213]. Combining our large-scale self-supervised learning approach with supervised finetuning on targeted small-scale labeled data may further enhance the final performance and interpretability of the system.

Part III

World Modeling: Imagining the Future Evolution

Chapter 5

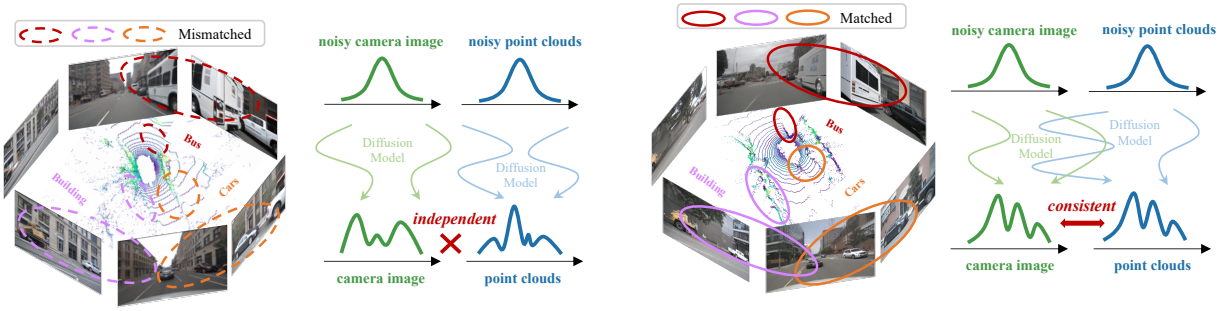
X-Drive: Cross-Modality Consistent Multi-Sensor Data Synthesis for Driving Scenarios

This chapter begins the *World Modeling* part of the dissertation by studying generative modeling for multi-sensor driving data. Once autonomous driving is viewed through the lens of scalable world representations, an important question is not only how to understand the world, but also how to generate unseen environment with specialized constraints. X-Drive addresses this challenge by modeling the joint distribution of LiDAR point clouds and multi-view images. The chapter argues that realistic world modeling for driving must preserve both per-modality realism and cross-modality consistency, since different sensors must describe the same geometry, object layout, and scene semantics. To this end, X-Drive introduces a dual-branch latent diffusion architecture with cross-modality conditioning based on epipolar geometry, enabling aligned and controllable generation across LiDAR and camera views.

5.1 Introduction

Autonomous driving vehicles perceive the world with multiple sensors of different kinds, where LiDAR and cameras play crucial roles by capturing point clouds and multi-view images. They provide complementary geometric measurements and semantic information about the surrounding environment, significantly benefiting tasks such as object detection [128, 236], motion planning [186], scene reconstruction [86, 269], and self-supervised representation learning [247, 233]. However, these advancements hinge on access to *large amounts of aligned multi-modality data*, specifically well-calibrated LiDAR and multi-view camera inputs that describe the same scene.

Scaling up the collection of such high-quality multi-modality data is costly and a non-trivial effort. The high-quality sensors are expensive, and the calibration process demands intensive human efforts. Additionally, real-world driving data suffers from the severe long-



(a) Single-modality images and point clouds synthesized by separate models

(b) Multi-modality images and point clouds jointly generated by our proposed

Figure 5.1: X-Drive simultaneously generates high-quality multi-view images and point clouds with cross-modality consistency, which is impossible for previous single-modality generative models.

tailed distribution problem, making it an obstacle for corner case collection such as in extreme weather conditions. This raises a natural question: *Can we use a controllable way to synthesize aligned multi-modality data?*

Given the success in other fields [168, 10, 125], generative models offer a promising solution. Current research focuses either on synthesizing point clouds [279, 79, 164] or multi-view images [51, 218, 226], with limited attention paid to generating multi-modality data. A simple combination of these single-modality algorithms results in serious cross-modality mismatches in the synthetic scenes (Fig. 5.1a). Such inconsistencies create ambiguities and even contradictory inputs or supervision signals, thus hindering the performance of downstream tasks.

Cross-modality consistency serves as the key desiderata of multi-modality data generation. However, there are several challenges in the generation of consistent LiDAR and camera data. Firstly, synthetic point clouds and multi-view images must be spatially aligned in all the local regions since they describe the same driving scene, *i.e.* the shapes and layouts of both foregrounds and backgrounds must be matched. Secondly, unlike those 2D pixel-level tasks [261], point clouds and multi-view images have distinct geometrical spaces and data formats. Multi-view images are represented by RGB values in camera perspective views, while point clouds are XYZ coordinates in the 3D space. Thirdly, the 3D spatial information is ambiguous during generation for both point clouds and multi-view images without reliable point location or pixel depth in the denoising process.

In this paper, we fill in this gap by proposing X-Drive, a novel framework for the joint generation of LiDAR and camera data, as demonstrated in Fig. 5.1b. We design a dual-branch architecture with two latent diffusion models separately dedicated to the synthesis of point clouds and multi-view images, while a key cross-modality condition module enhances *cross-modality consistency* between them. To model the joint distribution and ensure local

spatial alignment, we perform the cross-modality conditions locally with only corresponding regions from the other modality considered. An explicit transform bridges the two different geometrical spaces, converting the cross-modality condition to match each other's noisy latent space. To handle the positional ambiguity, we resort to a 3D-aware design based on epipolar lines on range images and multi-view images, allowing the cross-attention module to adaptively determine the cross-modality correspondence without explicit 3D positions. In consequence, X-Drive is able to exploit existing single-modality data as conditions to seamlessly generate data in the other modality. Additionally, X-Drive enhances controllability by introducing 3D bounding boxes for geometrical layout control and text prompts for attribute control (*e.g.* weather and lighting), enabling more flexible, fine-grained, and precise control over both modalities.

Extensive experiments demonstrate the great ability of X-Drive in generating realistic multi-modality sensor data. It notably outperforms previous specialized single-modality algorithms in the quality of both synthetic point clouds and multi-view images. More importantly, for the first time, it demonstrates reliable cross-modality consistency in synthetic scenes with comprehensive and flexible conditions. Our contributions are summarized as follows.

- We introduce X-Drive, a dual-branch multi-modality latent diffusion framework that, for the first time, enables controllable and reliable synthesis of aligned LiDAR and multi-view camera data.
- Our cross-modality epipolar condition module bridges the geometrical gap under spatial ambiguity between point clouds and multi-view images, significantly enhancing modality consistency.
- Extensive experimental results demonstrate the effectiveness of X-Drive, with notable MMD (for point cloud generation) and FID (for multi-view generation) improvements, establishing a new state-of-the-art for remarkable cross-modality consistency.

5.2 Related Works

Conditional generation with diffusion models. Diffusion models [72, 43, 156, 151] exhibit remarkable ability in generating diverse images by learning a progressive denoising process. They yield state-of-the-art results in various tasks such as text-to-image generation [168, 144], text-to-video generation [184, 258], and instructional image editing [13, 138]. Beyond text condition, several works emerges with the competence in managing additional forms of control signals. [261] integrates spatial conditions to a pretrained text-to-image diffusion model via efficient finetuning. [112, 264] condition the image synthesis on the fine-grained geometrical annotations to facilitate downstream tasks like 2D object detection. [125, 175] introduce 3D-aware diffusion models for novel view synthesis based on single-view

image inputs. Unlike these prior work for image domain, our framework simultaneously generates point clouds and multi-view images conditioned on text prompts and 3D bounding boxes.

Cross-modality data generation. Beyond image domain, generative models are extended to the synthesis of multi-modality data. [81, 180] leverage video diffusion model to estimate sequential depths. [7] formulates various vision tasks as next token prediction by extracting diverse visual information as unified visual tokens. The cross-modality synthesis between videos and audios also draws great attention. Efforts are devoted to the video-to-audio [276], audio-to-video [22], bi-directional [25, 63], and joint multi-modality [169] generation. However, in all the above cases, there exists clear cross-modality alignments, such as the pixel-to-pixel spatial correspondence for depth estimation and frame-to-frame temporal alignment between audios and videos. In contrast, point clouds and multi-view images are in different geometrical spaces without clear one-to-one local correlation between noisy multi-modality samples.

Generation of vehicle sensor data. Panoramic driving environment is usually perceived by the multi-view cameras and LiDARs on the ego-vehicle. Alternative to the laborious collection and annotation process, many works resort to the fast-growing generative models for the synthesis of either camera [192, 248, 51] or LiDAR [279, 79, 278] sensor data. For multi-view images [248, 218, 51] or videos [226, 130], previous work leverages pretrained image diffusion models [168] by incorporating extra inter-view modules to ensure the multi-view consistency. For point clouds, [279, 164, 79] adapt latent diffusion models for the generation of range images [48, 105] since this range-view representation shares a similar format with RGB images. However, all above methods concentrate on the single-modality data, either LiDAR or camera sensor. Despite various control signals, there exists no guarantee for the cross-modality consistency between point clouds and multi-view images generated by independent single-modality models. To this end, we propose to synthesize consistent multi-modality data in a joint manner.

5.3 Preliminary

Latent Diffusion Models. Diffusion models [72, 187] learn to approximate a data distribution $p(\mathbf{x}_0)$ by iteratively denoising a random Gaussian noise for T steps. Typically, diffusion models construct a diffused input $\mathbf{x}_t$ through a forward process, which gradually adds Gaussian noise to the data according to a variance schedule $\beta_1, \ldots, \beta_T$.

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}), \quad q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}). \tag{5.1}$$

Then, the reverse process learns to recover the original inputs by fitting $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$.

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t), \quad p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)). \tag{5.2}$$

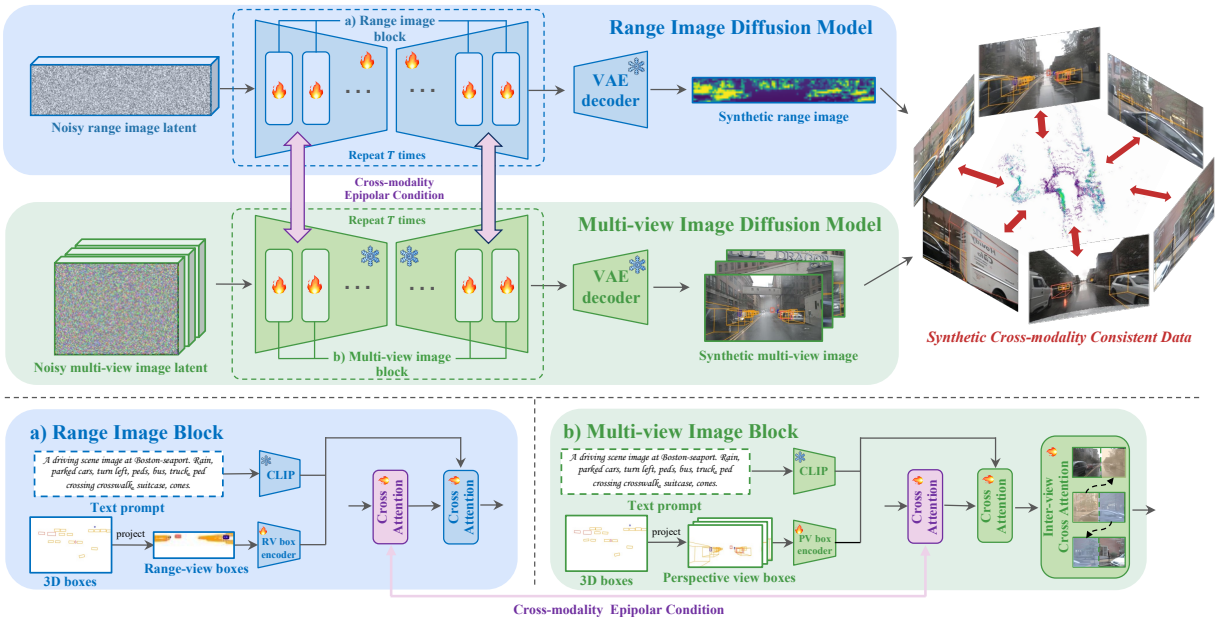


Figure 5.2: Overview of our proposed X-Drive framework. We design a dual-branch diffusion model architecture to generate multi-modality data. Cross-modality epipolar condition modules (Fig. 5.3) are inserted between branches to enhance the cross-modality consistency.

Latent diffusion models [168] perform the diffusion process on the latent space instead of the input space to handle the high dimensional data. Specifically, it maps input $\mathbf{x}$ with an encoder $\mathcal{E}$ into the latent space as $\mathbf{z} = \mathcal{E}(\mathbf{x})$. The latent code $\mathbf{z}$ can be reconstructed to the input as $\hat{\mathbf{x}} = \mathcal{D}(\mathbf{z})$ through a decoder $\mathcal{D}$. The forward and reverse processes of latent diffusion models are similar with the original diffusion models by substituting $\mathbf{x}$ with latent $\mathbf{z}$ in Eq. 5.1 and Eq. 5.2.

Range Image Representation. Compatible with the sampling process of LiDAR sensor, range image $\mathcal{R} \in \mathbb{R}^{H_r \times W_r \times 2}$ is a dense rectangular representation for LiDAR point clouds, where H_r and W_r are the numbers of rows and columns with one channel representing the range and the other representing intensity. The rows reflect the laser beams while the columns indicate the yaw angles. For each point with Cartesian coordinate (x, y, z), it can be transformed to the spherical coordinates (r, θ, ϕ) through the following projection.

$$r = \sqrt{x^2 + y^2 + z^2}, \quad \theta = \arctan(y, x), \quad \phi = \arctan(z, \sqrt{x^2 + y^2}) \tag{5.3}$$

where r is the range, θ is the inclination, and ϕ is the azimuth. The range image is produced by quantizing the θ and ϕ with factors s_θ and s_ϕ. For each point with spherical coordinate (r_i, θ_i, ϕ_i), we have $\mathcal{R}(\lfloor \theta_i / s_\theta \rfloor, \lfloor \phi_i / s_\phi \rfloor) = (r_i, i_i)$. The range r_i and intensity i_i are normalized to $(0, 1)$.

5.4 Methodology

In this section, we present our novel framework, X-Drive, for joint generation of point clouds and multi-view images. The overall architecture is depicted in Fig. 5.2. We extend the vanilla diffusion models to modeling the joint distribution of multi-modality data in Sec. 5.4.1. Practically, it is achieved by our proposed framework with dual-branch diffusion models in Sec. 5.4.2. For the cross-modality consistency, we propose a cross-modality epipolar condition module in Sec. 5.4.3 . Our method can also be tailored for cross-modality conditional generation in a zero-shot manner as in Sec. 5.4.4.

5.4.1 Joint Multi-modality Generation

Beyond single-modality diffusion models in Sec. 5.3, we give a formulation to our proposed X-Drive for multi-modality data generation. It aims to approximate a joint distribution for paired data $(\mathbf{r}_0, \mathcal{C}_0)$ that describes a specific driving scene, where $\mathbf{r}_0$ is the range image representation of point clouds and $\mathcal{C}_0 = \{\mathbf{x}_0^v\}_{v=1}^V$ denotes multi-view images from V cameras at different perspective views.

Given their similar rectangular latent formats, we set a shared noise schedule $\beta_1, \ldots, \beta_T$ for range images and multi-view images. The forward processes of either range image or multi-view images only depend on its own modality, equivalent with independent forward processes for each modality.

$$q(\mathbf{r}_t|\mathbf{r}_{t-1}) = \mathcal{N}\left(\mathbf{r}_t; \sqrt{1-\beta_t}\mathbf{r}_{t-1}, \beta_t\mathbf{I}\right), \quad q(\mathcal{C}_t|\mathcal{C}_{t-1}) = \mathcal{N}\left(\{\mathbf{x}_t^v\}_v; \{\sqrt{1-\beta_t}\mathbf{x}_{t-1}^v\}_v, \{\beta_t\mathbf{I}\}_v\right). \tag{5.4}$$

In contrast, the reverse process exploits models $\epsilon_{\theta_r}, \epsilon_{\theta_c}$ to predict the noise $\boldsymbol{\epsilon}_r$ and $\boldsymbol{\epsilon}_c$ added to the range image and multi-view images separately. Given the requirements for cross-modality consistency, the noise prediction for each modality should take each other into consideration, so the reverse process for either range image or multi-view images would depend on both modalities.

$$\begin{aligned} p_{\theta_r}(\mathbf{r}_{t-1}|\mathbf{r}_t, \mathcal{C}_t) &= \mathcal{N}(\mathbf{r}_{t-1}; \mu_{\theta_r}(\mathbf{r}_t, \mathcal{C}_t, t), \Sigma_{\theta_r}(\mathbf{r}_t, \mathcal{C}_t, t)), \\ p_{\theta_c}(\mathcal{C}_{t-1}|\mathcal{C}_t, \mathbf{r}_t) &= \mathcal{N}(\mathcal{C}_{t-1}; \mu_{\theta_c}(\mathcal{C}_t, \mathbf{r}_t, t), \Sigma_{\theta_c}(\mathcal{C}_t, \mathbf{r}_t, t)). \end{aligned} \tag{5.5}$$

Intuitively, we can implement two diffusion models as ϵ_{θ_r} and ϵ_{θ_c} separately for range images and multi-view images, equipped with encoders c_{CR} and c_{RC} for the other modality as extra conditions.

$$\hat{\boldsymbol{\epsilon}}_r = \epsilon_{\theta_r}(\mathbf{r}_t, c_{CR}(\mathcal{C}_t, t), t), \quad \hat{\boldsymbol{\epsilon}}_c = \epsilon_{\theta_c}(\mathcal{C}_t, c_{RC}(\mathbf{r}_t, t), t). \tag{5.6}$$

Instead of training separate condition encoders, we can directly treat diffusion models themselves as strong encoders for each modality. Thus, ϵ_{θ_c} and ϵ_{θ_r} are adapted as the condition encoder $\epsilon'_{\theta_c}(\mathcal{C}_t, t)$ and $\epsilon'_{\theta_r}(\mathbf{r}_t, t)$ in each block. Compared to text-to-image condition, a notable characteristic of LiDAR-camera mutual condition lies in the local spatial correspondence. For example, a car captured by LiDAR should share the consistent shape and location with the

same car captured by cameras but has little direct relationship with other parts in the images. Due to their different geometrical spaces, we should rely on a spatial transformation $\mathcal{T}(\cdot)$ to explicitly align the geometrical space for cross-modality correspondence. Concretely, $\mathcal{T}_{CR}$ transforms the multi-view image conditions to the range image space, while $\mathcal{T}_{RC}$ transforms the range image conditions to the multi-view image space. As a result, the condition encoders c_{CR}, c_{RC} can be formulated as follows.

$$c_{CR}(\mathcal{C}_t, t) = \mathcal{T}_{CR}(\epsilon'_{\theta_c}(\mathcal{C}_t, t)), \quad c_{RC}(\mathbf{r}_t, t) = \mathcal{T}_{RC}(\epsilon'_{\theta_r}(\mathbf{r}_t, t)). \tag{5.7}$$

In this case, we rewrite the denoising models in Eq. 5.6 as follows.

$$\hat{\boldsymbol{\epsilon}}_r = \epsilon_{\theta_r}(\mathbf{r}_t, \mathcal{T}_{CR}(\epsilon'_{\theta_c}(\mathcal{C}_t, t)), t), \quad \hat{\boldsymbol{\epsilon}}_c = \epsilon_{\theta_c}(\mathcal{C}_t, \mathcal{T}_{RC}(\epsilon'_{\theta_r}(\mathbf{r}_t, t)), t). \tag{5.8}$$

They can be trained with a joint multi-modality objective function $\mathcal{L}_{DM-M}$.

$$\mathcal{L}_{DM-M} = \mathcal{L}_{DM-R} + \mathcal{L}_{DM-C}, \quad \mathcal{L}_{DM-R} = \mathbb{E}_{r,t,\epsilon_r}||\epsilon_r - \hat{\epsilon}_r||_2^2, \quad \mathcal{L}_{DM-C} = \mathbb{E}_{\mathcal{C},t,\epsilon_c}||\epsilon_c - \hat{\epsilon}_c||_2^2. \tag{5.9}$$

5.4.2 Dual-Branch Joint Generation Framework

Our framework is developed following Eq. 5.8, which consists of two denoising models, *i.e.*, $\epsilon_{\theta_r}(\cdot)$ and $\epsilon_{\theta_c}(\cdot)$ respectively tailored for range images (Sec. 5.4.2) and multi-view images (Sec. 5.4.2).

Diffusion Model for Range Images

Architecture. We adapt existing latent diffusion model (LDM) [168] to learning the distribution of range image $\mathbf{r_0} \in \mathbb{R}^{H_r \times W_r \times 2}$ given its similar representation format with RGB images. In the first stage, the range image is compressed into latent feature $\mathbf{z}_0^r \in \mathbb{R}^{h_r \times w_r \times c_r}$ by a downsampling factor $f_r = H_r/h_r = W_r/w_r$. In the second stage, we train an LDM for range images from scratch to learn the latent distribution of $\mathbf{z}_0^r$. Range image reflects the panoramic view of the ego-vehicle, so its left-most and right-most sides are connected. Considering this constraint, we follow LiDARGen [279] and RangeLDM [79] to replace all the convolutions in both VAE and LDM with horizontally circular convolutions [176] where the left and right sides of the range image are treated as neighbors.

In the first stage, we follow the standard protocol to train the VAE including both encoder $\mathcal{E}_r$ and decoder $\mathcal{D}_r$ by maximizing the ELBO [167]. In addition, we attach an adversarial discriminator [90, 168] to mitigate the blurriness brought by the reconstruction loss. In the second stage, the LDM conditions the synthesis of range images on the cross-modality information from the multi-view image branch (elaborated in Sec. 5.4.3) as well as the text prompt and 3D boxes. The model is trained with loss function in Eq. 5.9.

Range-view bounding box condition. Given a 3D bounding box $(\mathbf{b}_i, c_i)$, $\mathbf{b}_i = \{(x_j, y_j, z_j)\}_{j=1}^8 \in \mathbb{R}^{8\times 3}$ represents the 3D coordinates of box corners and c_i denotes its semantic category. We

project each box corner to the range view using Eq. 5.3 as $\mathbf{b}_i^r = \{(r_j, \theta_j, \phi_j)\}_{j=1}^8$. These range-view corner coordinates are passed through a Fourier embedder. Then, the concatenated position embeddings of eight corners are encoded by MLP_p^r as $\mathbf{h}_{p,i}^r$. For category label c_i, we follow [90, 51] to pool the CLIP embedding [162] of the category name as $\mathbf{h}_{l,i}^r$. The box and label embeddings are combined and encoded by another MLP_b^r as a hidden vector $\mathbf{h}_{b,i}^r$.

$$\mathbf{h}_{b,i}^r = \text{MLP}_b^r([\mathbf{h}_{l,i}^r, \mathbf{h}_{p,i}^r]), \quad \mathbf{h}_{l,i}^r = \text{Avg}(\text{CLIP}(c_i)), \quad \mathbf{h}_{p,i}^r = \text{MLP}_p^r(\text{Concat}(\text{Foruier}(\mathbf{b}_i^r))) \tag{5.10}$$

We concatenate box hidden vectors $\{\mathbf{h}_{b,i}^r\}_i$ from this range-view box encoder with text prompt hidden vectors, which guide the synthesis the latent diffusion model via a cross-attention module.

Diffusion Model for Multi-view Images

Architecture. To model the distribution of multi-view images $\mathcal{C}_0 = \{\mathbf{x}_0^v\}_{v=1}^V, \mathbf{x}_0^v \in \mathbb{R}^{H_c \times W_c \times 3}$, we tailor the latent diffusion model by introducing extra bounding box and inter-view conditions apart from the cross-modality signals from range images (details in Sec. 5.4.3). Following standard LDM [168], each image $\mathbf{x}_0^v$ is first compressed by a pretrained VAE to the latent space as $\mathbf{z}_0^{c,v}$. In the training process, we sample the Gaussian noise for each view independently and the denoising model is trained with the objective function in Eq. 5.9.

Multi-view bounding box condition. The 3D box condition module is similar with Sec. 5.4.2, but the box positions are transformed to the perspective view of each camera. For each 3D bounding box $(\mathbf{b}_i, c_i)$, we project its 3D corner coordinates $\mathbf{b}_i = \{(x_j, y_j, z_j)\}_{j=1}^8$ to corresponding camera perspective view as $\mathbf{b}_i^c = \{(u_j, v_j, d_j)\}_{j=1}^8$, where u_j, v_j are the corner location in the pixel coordinate and d_j is the depth. For each camera view v, its synthesis is only conditioned on the 3D bounding boxes with at least one corner projected into the range of its perspective view image. Then, we formulate this perspective-view box encoder similar with Eq. 5.10.

$$\mathbf{h}_{b,i}^c = \text{MLP}_b^c([\mathbf{h}_{l,i}^c, \mathbf{h}_{p,i}^c]), \quad \mathbf{h}_{l,i}^c = \text{Avg}(\text{CLIP}(c_i)), \quad \mathbf{h}_{p,i}^c = \text{MLP}_p^c(\text{Concat}(\text{Fourier}(\mathbf{b}_i^c))) \tag{5.11}$$

For each camera view, the box embeddings $\{\mathbf{h}_{b,i}^c\}_i$ are concatenated with the text prompt hidden vectors, which together guide the image generation through a cross-attention module.

Inter-view condition. It is critical to enhance the consistency across different views in the generation of multi-view images $\mathcal{C}_0 = \{\mathbf{x}_0^v\}_{v=1}^V$, so we inject inter-view cross-attention module to condition the generation of each camera view v on its left and right adjacent views vl and vr. Given the small overlapping between field-of-views of adjacent cameras, we split each image latent $\mathbf{z}_{in}^{c,v}$ into two halves horizontally. For each view v, its left half $\mathbf{z}_{in,l}^{c,v}$ attends to the right half $\mathbf{z}_{in,r}^{c,vl}$ of its left neighbor view vl, while its right half $\mathbf{z}_{in,r}^{c,v}$ depends on the left half $\mathbf{z}_{in,l}^{c,vr}$ of its right neighbor vr.

$$\mathbf{z}_{out}^{c,v} = \mathbf{z}_{in}^{c,v} + \tanh(\alpha_v) \cdot \text{Concat}_{width}\left(\left[\text{CrossAttn}(\mathbf{z}_{in,l}^{c,v}, \mathbf{z}_{in,r}^{c,vl}); \text{CrossAttn}(\mathbf{z}_{in,r}^{c,v}, \mathbf{z}_{in,l}^{c,vr})\right]\right) \tag{5.12}$$

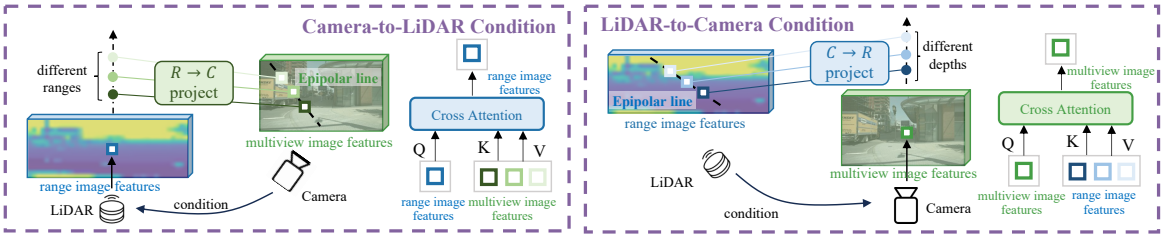


Figure 5.3: Cross-modality epipolar condition module. We perform mutual conditions locally between LiDAR and camera modalities based on epipolar lines on multi-view image and range image latents.

where α_v is a zero-initialization gate [261] for stable optimization. Compared to full cross-attention in MagicDrive [51], our split strategy significantly reduces the per-scene GPU memory cost of inter-view condition from 11GB to 3GB with better multi-view consistency.

5.4.3 Cross-Modality Epipolar Condition Module

The key to multi-modality data synthesis is to boost the cross-modality consistency, which potentially relies on cross-modality conditions. Ideally, there is an explicit point-to-pixel correspondence between point clouds and multi-view images through camera projection. However, in the denoising process, we are not aware of either the range values of range images or the depths of the multi-view images. Instead, we propose to warp the local features from range images and multi-view images based on epipolar lines through spatial transforms $\mathcal{T}_{RC}$ and $\mathcal{T}_{CR}$ (Eq. 5.7) to provide control signals for the synthesis of the other modality, illustrated in Fig. 5.3.

Camera-to-LiDAR condition. For each position (ϕ, θ) on the range image latent $\mathbf{z}^r_{in}$, we sample R points along the range axis following the linear-increasing discretization (LID) [165, 126] with range value $r_k, k = 1, 2, \ldots, R$. We transform these points $\{(\phi, \theta, r_k)\}_{k=1}^{R}$ to their 3D coordinates $\{(x_k, y_k, z_k)\}_{k=1}^{R}$.

$$x_k = r_k \cdot \cos(\phi) \cdot \sin(\theta), \quad y_k = r_k \cdot \cos(\phi) \cdot \cos(\theta), \quad z_k = r_k \cdot \sin(\phi). \tag{5.13}$$

They can be projected to the camera perspective view v with the camera parameters $(\mathbf{R}_v, \mathbf{t}_v, \mathbf{K}_v)$ as

$$d_k \begin{bmatrix} u_k & v_k & 1 \end{bmatrix}^T = \mathbf{K}_v \left(\mathbf{R}_v \begin{bmatrix} x_k & y_k & z_k \end{bmatrix}^T + \mathbf{t}_v \right), \quad k = 1, 2, \ldots, R \tag{5.14}$$

where $(u_k, v_k), k = 1, 2, \ldots, R$ are the pixel coordinates for the R points along the LiDAR ray. They form the epipolar line on the camera image $\mathbf{x}^v$ corresponding to position (ϕ, θ) of the LiDAR range image. We bilinearly sample the corresponding local features from the image latent $\mathbf{z}^{c,v}_{in}$ of view v at location (u_k, v_k). If a 3D point (x_k, y_k, z_k) is projected into more than one camera view, we would simply adopt the average local features. Extra Fourier

embedding of r_k is added to each sampled feature as an indicator for the range value along the ray. As a result, the entire spatial transform $\mathcal{T}_{CR}$ makes the camera image features spatially aligned to the range image features.

$$\mathcal{T}_{CR}(\mathbf{z}_{in}^{c,v})_{(\phi,\theta)} = \{\text{Bilinear-Sample}(\mathbf{z}_{in}^{c,v}; (u_k, v_k)) + \text{MLP}_r\left(\text{Fourier}(r_k)\right)\}_{k=1}^{R}. \tag{5.15}$$

We apply a cross-attention module to condition the range image feature $\mathbf{z}_{in,(\phi,\theta)}^{r}$ at coordinate (ϕ, θ) on the corresponding transformed local image features $\mathcal{T}_{CR}(\mathbf{z}_{in}^{c})_{(\phi,\theta)}$ on the epipolar line.

$$\mathbf{z}_{out,(\phi,\theta)}^{r} = \mathbf{z}_{in,(\phi,\theta)}^{r} + \tanh(\alpha_{cr}) \cdot \text{CrossAttn}\left(\mathbf{z}_{in,(\phi,\theta)}^{r}, \mathcal{T}_{CR}(\mathbf{z}_{in}^{c,v})_{(\phi,\theta)}\right) \tag{5.16}$$

where α_{cr} is a zero-initialization gate for the camera-to-LiDAR condition. The model learns the adaptive local correspondence between range image and multi-view images under range ambiguity.

LiDAR-to-camera condition. Inversely, we condition the synthesis of multi-view images on the LiDAR range images using a similar module. For each position (u, v) on the camera image latent $\mathbf{z}_{in}^{c,v}$ from view v, D points are sampled at different depths following the linear-increasing discretization with depth value $d_k, k = 1, 2, \ldots, D$. Each point (u, v, d_k) in the pixel coordinate is transformed to 3D coordinate (x_k, y_k, z_k) via the corresponding camera parameters $(\mathbf{R}_v, \mathbf{t}_v, \mathbf{K}_v)$.

$$\begin{bmatrix} x_k & y_k & z_k \end{bmatrix}^T = \mathbf{R}_v^T \left(\mathbf{K}_v^{-1} \begin{bmatrix} u_k \cdot d_k & v_k \cdot d_k & d_k \end{bmatrix}^T - \mathbf{t}_v \right), \quad k = 1, 2, \ldots, D \tag{5.17}$$

The 3D point coordinate is then projected to the LiDAR range view using Eq. 5.3 as (ϕ_k, θ_k, r_k). The D points $(\phi_k, \theta_k), k = 1, 2, \ldots, D$ form the epipolar line on the LiDAR range image corresponding to the pixel coordinate (u, v) of camera view v. The local range image features are extracted at each coordinate (ϕ_k, θ_k) from range image latent $\mathbf{z}_{in}^{r}$ through bilinear sampling with additional Fourier embedding for the depth value $d_k, k = 1, 2, \ldots, D$ attached. We can write the spatial transform $\mathcal{T}_{RC}$ aligning the range view features to multi-view image features as follows.

$$\mathcal{T}_{RC}(\mathbf{z}_{in}^{r})_{(u,v)} = \{\text{Bilinear-Sample}(\mathbf{z}_{in}^{r}; (\theta_k, \phi_k)) + \text{MLP}_d\left(\text{Fourier}(d_k)\right)\}_{k=1}^{D}. \tag{5.18}$$

Similar with camera-to-LiDAR condition, a cross-attention module conditions the local camera image feature $\mathbf{z}_{out,(u,v)}^{c,v}$ at coordinate (u, v) on spatially aligned range image features $\mathcal{T}_{RC}(\mathbf{z}_{in}^{c,v})_{(u,v)}$.

$$\mathbf{z}_{out,(u,v)}^{c,v} = \mathbf{z}_{in,(u,v)}^{c,v} + \tanh(\alpha_{rc}) \cdot \text{CrossAttn}\left(\mathbf{z}_{in,(u,v)}^{c,v}, \mathcal{T}_{RC}(\mathbf{z}_{in}^{r})_{(u,v)}\right) \tag{5.19}$$

where α_{rc} is a zero-initialization gate. The cross-attention module adaptively learns the local reliance of camera multi-view images on LiDAR range images without explicit depth cues.

5.4.4 Cross-Modality Conditional Generation

Besides multi-modality joint generation, X-Drive can work as a camera-to-LiDAR or LiDAR-to-camera conditional generation model in a zero-shot manner without targeted training. Given either one of ground-truth LiDAR range image $\mathbf{r}_0$ or multi-view images $\mathcal{C}_0$, we change Eq. 5.8 separately for camera-to-LiDAR or LiDAR-to-camera generation to synthesize the other modality data.

$$\hat{\boldsymbol{\epsilon}}_r = \epsilon_{\theta_r}(\mathbf{r}_t, \mathcal{T}_{CR}(\epsilon'_{\theta_c}(\mathcal{C}_0, 0)), t), \quad \hat{\boldsymbol{\epsilon}}_c = \epsilon_{\theta_c}(\mathcal{C}_t, \mathcal{T}_{RC}(\epsilon'_{\theta_r}(\mathbf{r}_0, 0)), t). \tag{5.20}$$

In this case, the trained diffusion model for the input modality serves as a strong encoder for the condition to enhance the adherence of generation to the input LiDAR or camera conditions.

5.5 Experiments

5.5.1 Experimental Setups

Dataset. We evaluate our method using nuScenes dataset [18]. It provides point clouds from a 32-beam LiDAR and multi-view images from 6 cameras. We follow the official setting to employ 700 driving scenes for training and 150 scenes for validation. The generation of X-Drive is conditioned on bounding boxes from 10 semantic classes as well as scene descriptions.
Evaluation Metrics. For joint multi-modality data synthesis, we evaluate both single-modality data realism and cross-modality consistency. For image modality, we report Fréchet Inception Distance (FID) for image synthesis realism. We also utilize View Consistency Score (VSC) [192] to evaluate multi-view consistency, which reflects the keypoints correspondence on overlapping regions between adjacent camera views (Fig. 5.5), and we normalize the VSC of real images to 1.0 for clarity. For point clouds modality, we measure the distribution gap between synthetic and real point clouds with Maximum Mean Discrepancy (MMD) and Jensen-Shannon divergence (JSD) following RangeLDM [79]. For cross-modality consistency, we propose a novel metric called Depth Alignment Score (DAS). We project the synthetic point clouds to multi-view images as sparse depth values and also estimate the image depth using DepthAnythingV2 [250]. The mean absolute error between the projected and estimated disparities is used as DAS metric.
Baselines. To our best knowledge, we are the first to work on the joint generation of point clouds and multi-view images. For single-modality generation, we compare with state-of-the-art generation algorithms for point clouds, *i.e.* LiDAR VAE [16], LiDARGen [279], RangeLDM [79], and multi-view images, *i.e.* BEVGen [192], BEVControl [248], MagicDrive [51], with released code or quantitative results on nuScenes dataset. Furthermore, we combine MagicDrive [51] and RangeLDM [79] respectively for images and point clouds as a multi-modality baseline.
Training Setup. Our X-Drive has a dual-branch architecture. We utilize the Stable-Diffusion V1.5 pretrained weight to initialize the multi-view image branch with other newly

Table 5.1: Quantitative comparison with driving data generation algorithms. For each column, the best value is highlighted by **bold**, and the second best one is denoted by underline.

Modality	Method	Image quality		Point clouds quality		Cross-modality
		FID ↓	VSC ↑	MMD ↓	JSD ↓	DAS ↓
C	BEVControl	25.54	-	-	-	-
	BEVGen	24.85	0.439	-	-	-
	MagicDrive	16.20	0.633	-	-	-
L→C	X-Drive (ours)	**16.01**	**0.721**	-	-	-
L	LiDAR VAE	-	-	1.1×10^{-3}	-	-
	LiDARGen	-	-	1.9×10^{-3}	0.160	-
	RangeLDM	-	-	1.9×10^{-4}	0.054	-
C→L	X-Drive (ours)	-	-	$\mathbf{1.0 \times 10^{-4}}$	**0.052**	-
C+L	MagicDrive + RangeLDM	-	-	-	-	2.32
	X-Drive (ours)	17.37	0.675	1.2×10^{-4}	**0.052**	**1.69**

added parameters randomly initialized. We follow a three-stage pipeline for the training. In the first stage, the VAE is trained from scratch for range images. In the second stage, we train the latent diffusion model in point clouds branch without cross-modality conditions using the frozen VAE. Finally, in the last stage, the entire multi-modality model is trained in an end-to-end manner with only the Stable-Diffusion V1.5 parts frozen. We follow MagicDrive [51] and RangeLDM [79] to synthesize 224×400 multi-view camera images and 32×1024 point cloud range images.

5.5.2 Quantitative Results

We report the quantitative results for both joint generation of aligned point clouds and multi-view images as well as conditional LiDAR-to-camera or camera-to-LiDAR cross-modality generation.

Joint multi-modality generation. As shown in Tab. 5.1, in comparison to specialized single-modality generation methods, X-Drive, as a multi-modality algorithm, achieves comparable or even better quality in both synthetic point clouds and multi-view images. Our FID metric is outoerformed by MagicDrive [51], since we do not have map condition which offers strong control signals. Unlike simple combination of single-modality methods, we can generate point clouds and images with cross-modality alignment, reflected by our superior DAS metric, thanks to our proposed cross-modality epipolar condition module. Extra qualitative visualizations are shown in Appendix C.4.

Conditional cross-modality generation. As mentioned in Sec. 5.4.4, X-Drive can also work as a LiDAR-to-camera or camera-to-LiDAR conditional generation model. In Tab. 5.1, for single-modality data generation, X-Drive can outperform previous baselines for both point clouds and multi-view images, demonstrating the flexibility of our proposed algorithm

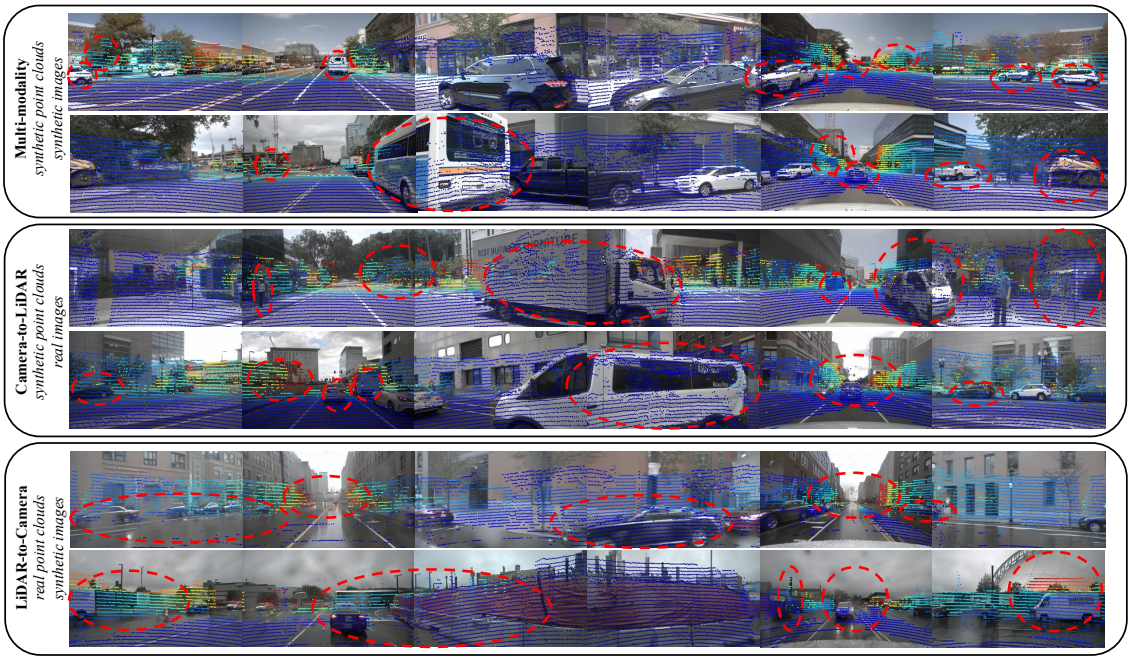


Figure 5.4: Cross-modality consistency qualitative results for multi-modality generation and conditional cross-modality generation. Colors of point clouds refer to different depths. Well-matched regions between point clouds and multi-view images are highlighted with red circles.

for cross-modality data synthesis.

Object-level controllable generation. X-Drive can adhere to the 3D bounding box conditions in the generative process. Since previous LiDAR generation methods do not show this ability [79, 164], we only compare our methods with multi-view image generation algorithm [51]. For fair comparison with single-modality method, we employ synthetic multi-view images and real point clouds from nuScenes validation set to run a pretrained SparseFusion [236] model, which is an advanced multi-sensor object detection algorithm. Tab. 5.2 shows X-Drive outperforms MagicDrive [51] *w.r.t.* the object-level fidelity.

Table 5.2: Object-level control.

Methods	mAP	NDS
Oracle	70.5	72.8
MagicDrive	65.2	69.2
X-Drive (ours)	**65.4**	**69.6**

5.5.3 Qualitative Analysis

Cross-modality consistency. We show some multi-modality generation examples to exhibit the cross-modality consistency qualitatively. As in Fig. 5.4, the projected point clouds overlap with multi-view image contents properly for both foregrounds and backgrounds. We also show the results of conditional camera-to-LiDAR and LiDAR-to-camera generation in Fig. 5.4, where the synthetic data adheres to the cross-modality LiDAR or camera conditions



Figure 5.5: Key points correspondence between adjacent synthetic camera images. Our proposed method (bottom) can bring higher multi-view consistency than the baseline (top).

well.

Multi-view consistency. In our multi-modality generation, point clouds provide a natural 3D-aware geometrical guidance for the multi-view images, benefiting the multi-view consistency. In Fig. 5.5, we visualize the keypoints matching results [190] between adjacent synthetic camera images, where better correspondence is witnessed for X-Drive than MagicDrive [51].

Scene-level and object-level control. Given the text and bounding box conditions, X-Drive generate diverse results using different control signals, as illustrated in Fig. 5.6. For scene-level control, we can target for various lighting and weathers through the text prompts, while synthetic images still remain realistic and adhere to the object layouts. For object-level control, we can edit the scene by deleting objects or inserting objects at desired locations with specific sizes and semantic categories.

5.5.4 Ablation Studies

Due to limited computational resources, we employ shorter training schedule for ablation studies.

Cross-modality condition. The cross-modality condition is critical for cross-modality consistency. As shown in Tab. 5.3, removing this module significantly hurts the cross-modality alignment although the shared bounding boxes and text prompts still provide some constraints. Given the spatial ambiguity, we resort to cross-attention along epipolar lines to adaptively learn the local correspondence in Eq. 5.16 and Eq. 5.19. Instead, simple averaging along epipolar lines instead of cross-attention is quite noisy and poses severe damage to the point clouds quality and cross-modality consistency.

Bounding box and text prompt input conditions. Although X-Drive also supports unconditional generation, these input conditions provide multi-level control signals to generate diverse outputs. Tab. 5.3 shows that text and box conditions improve the quality of synthetic point clouds and images.

Classifier-free guidance. We employ classifier-free guidance (CFG) during inference for box and text conditions. Illustrated in Fig. 5.7, it improves the realism of synthetic data. However, higher CFG scale would hurt the image and point cloud quality by increasing the contrast and sharpness.

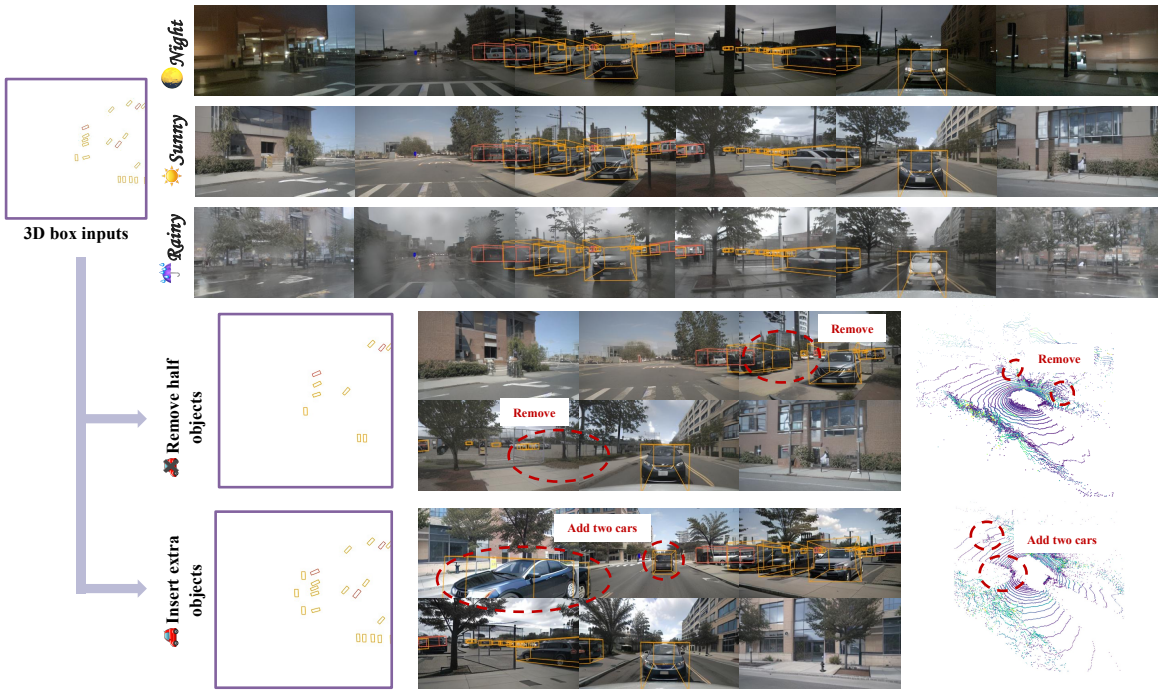


Figure 5.6: Scene-level and object-level controllable generation by changing the input conditions.

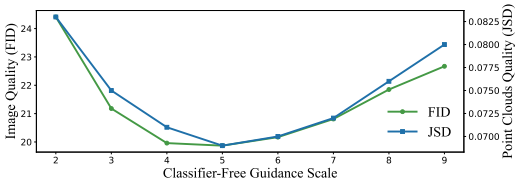


Figure 5.7: Effect of CFG scales.

Table 5.3: Ablation studies of X-Drive modules.

Methods	FID↓	JSD↓	DAS↓
w/o 3D box condition	29.60	0.091	1.76
w/o text prompt condition	25.76	0.070	1.73
w/o cross-modality	24.41	0.081	1.95
average epipolar condition	20.70	0.098	2.23
full model	**20.17**	**0.070**	**1.67**

5.6 Conclusions

We present X-Drive, a novel framework for the multi-modality generation of aligned LiDAR point clouds and multi-view camera images. By leveraging a dual-branch diffusion model, X-Drive captures the joint distribution of multi-modality data, ensuring mutual dependence between modalities. A key feature is the cross-modality condition module, which uses epipolar lines on LiDAR range images and camera views to address spatial ambiguity during the denoising process. Additionally, X-Drive supports multi-level control, enabling synthesis based on text prompts and object bounding boxes. Extensive experiments show that X-Drive generates high-quality, geometrically and semantically aligned point clouds and images, maintaining fidelity to the input conditions while accurately describing the same driving scene together.

Chapter 6

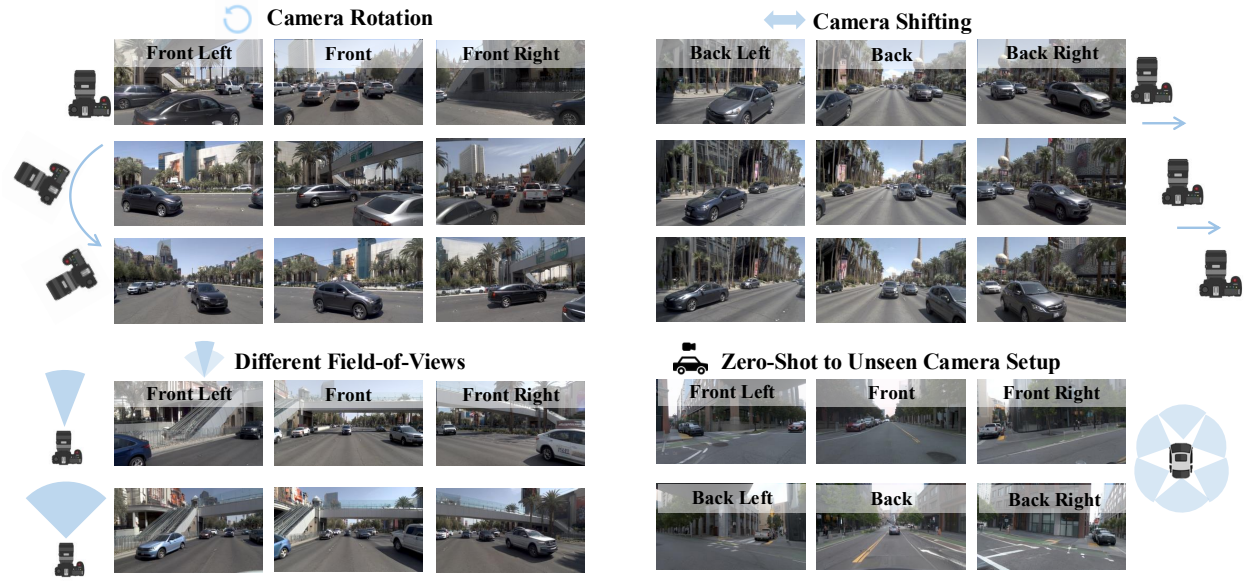
RAYNOVA: Geometry-Free Auto-Regressive 4D World Modeling with Unified Spatio-Temporal Representation

This chapter continues the *World Modeling* part of the dissertation by extending it to long-horizon full-scene future generation. This chapter focuses on the representation and generation of the temporal evolution of the world itself. RayNova addresses this problem through a geometry-free autoregressive framework for multi-view video generation. By combining dual-causal spatio-temporal modeling with an isotropic representation in continuous Plücker-ray space, the model captures multi-view, multi-frame, and long-horizon dynamics without being tied to a fixed sensor configuration or narrow prediction target. Together with a recurrent training strategy that aligns training and inference distributions.

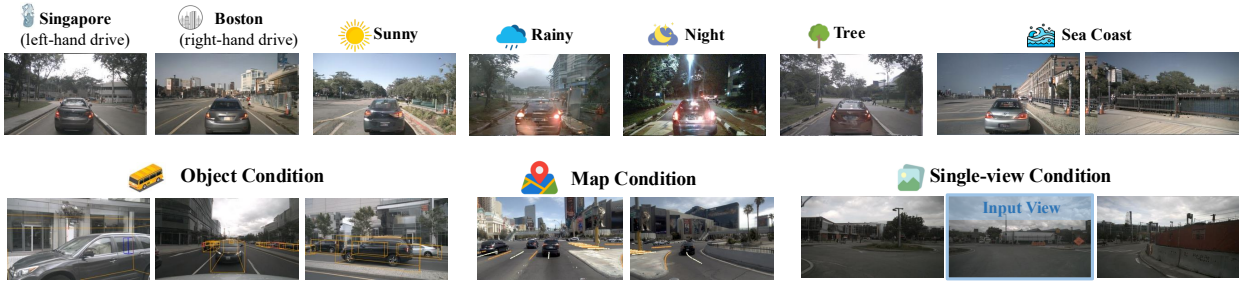
6.1 Introduction

World foundation models (WFMs) [103, 147, 15, 60, 145] aim to simulate the evolution of complex scenes within a dynamic real world governed by physical laws. In a continuous 4D spatio-temporal world, a world model is designed to generate physically plausible images or videos captured by cameras positioned at specific locations and timesteps.

To impose physical plausibility, prior work commonly considers spatial and temporal constraints separately. For spatial reasoning, some methods leverage positional correlations across multiple viewpoints, including camera adjacency [51, 226] and cross-view projection [58]. For temporal coherence, previous works assume strong multi-frame dependencies among images captured by the same cameras [226, 224, 171] and thereby exploit existing techniques from video generation, such as temporal VAEs [252, 14]. However, this decoupled design fundamentally restricts the model's flexibility to handle novel sensor configurations



(a) RayNova is not bound to any specific sensor setups, allowing flexible video generation in continuous 4D world space.



(b) RayNova takes diverse input conditions with high fidelity.



(c) RayNova can generate long-horizon multi-view videos in a large range.

Figure 6.1: Demonstrations of RayNova as Versatile World Foundation Model.

and rapid camera motions. Alternatively, some approaches construct explicit 3D scene representations (e.g. point clouds [235, 270] or BEV/volume features [263, 130]), within a unified global coordinate system to enhance spatio-temporal consistency. Although empirically effective within constrained domains, this geometry-forcing approach impedes generalization to unconstrained open-world environments beyond the distribution of training scenarios.

To overcome these limitations, we pursue a universal framework that retains physical plausibility with minimal inductive biases. We propose RayNova, a world foundation model supported by a *dual-causal autoregressive strategy* over both scale and time. The scale-wise autoregression is built up on "next-scale prediction" visual autoregressive model [196, 61], enabling the model to capture intricate visual patterns across multiple levels of abstraction. The temporal autoregression works frame-by-frame in the unified 4D spatio-temporal space to improve temporal coherence.

The core insight of RayNova lies in a novel spatio-temporal conditioning mechanism that represents the physical world with *relative positional encoding in camera ray space*. This ray-level representation is isotropic in continuous 4D space, thereby reducing dependency on specific camera configurations, motion patterns, or view overlaps. By encoding relative rather than absolute positions, it naturally supports extrapolation beyond the training distribution and generalizes to unconstrained real-world conditions. Leveraging this representation, RayNova performs coherent reasoning across multi-view and multi-frame images, accommodating diverse input conditions within a unified 4D geometric space. We highlight the following aspects of our method:

- **Versatile World Foundation Model.** Our method supports diverse input and output formats for various use cases with a single model. Video generation can be conditioned on multiple optional control signals (e.g., texts, objects, maps, images), while output formats remain flexible in terms of perspectives, resolutions, and frame rates.

- **Scalable Data-Driven Framework.** Our training paradigm is capable of ingesting heterogeneous training data from diverse sources with different sensor configurations. Unlike prior works with handcrafted biases [58, 23], RayNova does not require auxiliary supervision such as depth maps, point clouds, or optical flow.

- **Extendable Position Embedding.** Our relative ray-level positional encoding supports extrapolation beyond the training range, theoretically allowing an unbounded spatial extent. We also develop a recurrent training paradigm to mitigate distribution drift in long video generation.

- **Efficient Video Generation.** Our hierarchical multi-scale representation enables rapid progression from coarse abstractions to fine-grained details, and can be seamlessly integrated with existing acceleration techniques for visual autoregressive models [57].

We evaluate RayNova on driving datasets, demonstrating superior performance in both video generation and novel view synthesis. The results demonstrate high visual realism,

strong spatio-temporal coherence, and high fidelity to multiple control signals together with substantial computational efficiency. These advantages enhance its practicality for physical-world applications such as autonomous driving simulation.

6.2 Related Works

Video Generative Models. Recent advances in generative models have pushed the fidelity and diversity of monocular video synthesis to new heights. Early transformer-based, masked autoregressive approaches operate on discrete tokens to generate high-quality frames in sequence [243, 139, 52, 205, 257]. In parallel, diffusion-based pipelines iteratively denoise latent or pixel representations, yielding vivid motion and appearance [11, 147, 2, 157, 170, 53, 132, 252]. In just a few years, these models excel at visual quality, offering realistic and vivid generated videos. Most of these approaches focus on text-to-video tasks, generating videos conditioned on textual prompts [53, 132, 252]. Some other models also generate videos conditioned on reference images [216, 166], or reference videos [122]. In this paper, we go beyond video generation for a versatile world foundation model with flexible multi-view camera setups, ego motions, and input conditions.

3D Scene Representation Learning. To more accurately simulate the geometry and dynamics of real-world scenes [42], several approaches leverage explicit 3D priors to enhance the geometric consistency, typified by NeRF [140, 97] and Gaussian splatting [32, 245, 152, 268] variants. Despite high-quality image renderings, their inherent inductive biases limit the adaptability of models to complex scenarios beyond the training distribution. On the other hand, several approaches work on the geometry-free representation with improved generalization ability, most of which resort to camera ray space [174, 74, 96, 94]. However, despite these advances, they still rely on the camera rays in the global coordinate, which serves as another bottleneck of generalization for extrapolation beyond the training distribution. In contrast, we exploit the relative positions in the camera ray space for spatio-temporal grounding with minimal inductive biases, which boosts the video generation in a large range.

World Models for Autonomy. Autonomous driving serves as an ideal testbed for physically grounded world models due to its stringent demands for modeling complex geometric environments, dynamic actor interactions, and the need for coherent spatio-temporal representations. Early efforts [219, 77] demonstrate action-conditioned video generation but are constrained primarily to a single front-view perspective. Later studies [223, 218, 226] have extended video generation to multi-view cameras with diverse input conditions such as textual descriptions, object bounding boxes, and HD maps. To enhance geometric consistency and structural controllability, the community commonly introduces various inductive biases. Several approaches [223, 51, 226] count the spatial consistency of the mutual condition between adjacent cameras, but this design binds the model to some fixed camera configurations. Other efforts incorporate explicit 3D scene representations such as point clouds [235, 270,

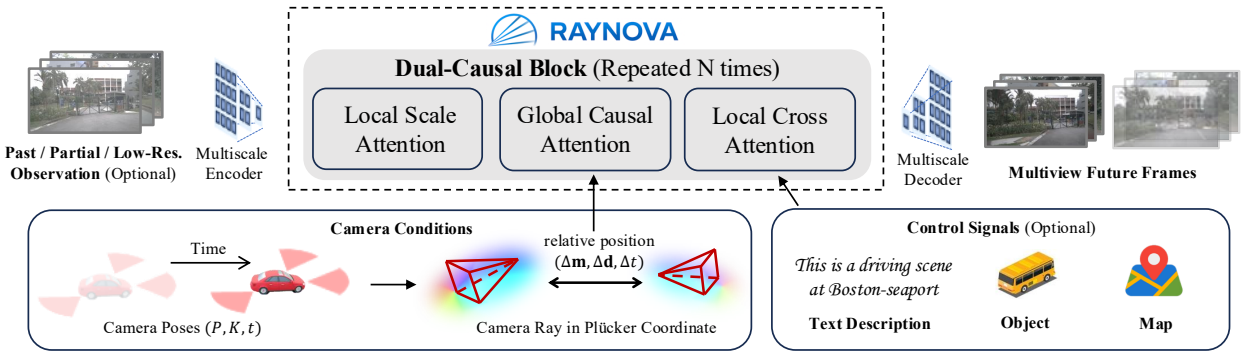


Figure 6.2: Overview of RayNova Framework. RayNova is composed of dual-casual (scale and time) blocks. The local scale attention and local cross attention works on each image indepedently, while the global causal attention works across multi-view and multiframe images enhanced with a unified ray-level relative position embedding for better spatio-temporal consistency.

263], occupancy grids [130], 3D latent features [277, 50] and Gaussian splatting [41] to ensure spatial alignment in world space. However, the reliability of the above 3D structural priors highly depends on the overlap between camera perspective views. These representations also limit the world models to a constrained 3D space, hindering generalization in the open world. In contrast, our approach uniquely learns a continuous 4D latent representation based on the relative position in the camera ray space that jointly encodes spatial and temporal dimensions, enforcing multi-view spatial consistency and frame-to-frame temporal coherence with minimal hand-crafted biases.

6.3 Methodology

RayNova, as shown in Fig. 6.2, is a versatile world foundation model with autoregression. After introducing preliminary “next-scale prediction" work (Sec. 6.3.1), we formulate the dual-causal autoregressive framework for multi-view video generation in Sec. 6.3.2. Critical for coherent spatio-temporal reasoning, we construct an isotropic 4D position embedding based on the relative positions between camera rays in Sec. 6.3.3. This enables the development of a transformer-based geometry-free framework with minimal inductive biases for world modeling that takes diverse control signals in Sec. 6.3.4. Finally, to alleviate the distribution drift in the autoregressive generation of long videos, we develop a recurrent training paradigm that aligns the distributions in the training and inference stages (Sec. 6.3.5).

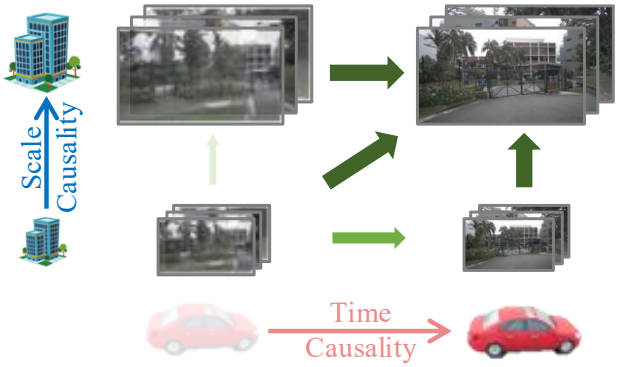


Figure 6.3: Dual-Causality for Multi-View Video Generation. Green arrows represent the causal dependency, while the darkness indicates the topological order of autoregression (from light to dark).

6.3.1 Preliminary: Next-Scale Prediction

Unlike vanilla autoregressive models [104, 256] that flatten the 2D grids of images into 1D tokens, some recent work [196, 61] shifts from "next-token prediction" to "next-scale prediction" strategy. Each image is quantized by the tokenizer into K multiscale token maps $X_{1:K} = (X_1, X_2, \ldots, X_K)$ with increasingly higher resolutions $h_k \times w_k, k = 1, 2, \ldots, K$. The autoregressive likelihood is formulated as follows.

$$p(X_1, X_2, \ldots, X_K) = \prod_{k=1}^{K} p(X_k | X_1, X_2, \ldots, X_{k-1}) \tag{6.1}$$

where X_k is the token map at scale k containing $h_k \times w_k$ visual tokens. The sequence $(X_1, \ldots, X_{k-1})$ is the prefix context for the prediction of X_k. During the k-th autoregressive step, all $h_k \times w_k$ tokens are generated in parallel.

6.3.2 Formulation of Dual-Causal Autoregression

We formulate our proposed RayNova for world modeling via multi-view videos beyond image-level "next-scale prediction". In favor of versatility, each image $I^{v,t}$ is tokenized independently as a multiscale token map $X^{v,t}_{1:K}$ for each frame $t = 1, \ldots, T$ and each camera view $v = 1, \ldots, V$. To model the joint distribution of multi-view and multi-frame images in a unified space, our autoregressive model considers two folds of causality: *scale* and *time* (Fig. 6.3).

For the scale causality, we start from a single-frame case. The distribution of multi-view images in the same frame is modeled in a joint manner, since multi-view images describe a unified 3D space at a specific time step as a whole. For clarity, we omit the timestep

superscripts.

$$p(X_1^{1:V}, \ldots, X_K^{1:V}) = \prod_{k=1}^{K} p(X_k^{1:V} | X_1^{1:V}, \ldots, X_{k-1}^{1:V}). \tag{6.2}$$

For time causality, the generation of each frame is conditioned on historical information. Different from some prior works [226, 171], we do not assume strong inter-dependency among multi-frame images from the same camera since this inductive bias cannot be maintained with complex ego-motions (e.g. turning). In contrast, we condition the multi-view image generation for the current timestep on all the views from past frames.

$$p(X_{1:K}^{1:V,1:T}) = \prod_{t=1}^{T} p(X_{1:K}^{1:V,t} | X_{1:K}^{1:V,1:t-1}). \tag{6.3}$$

Combining the scale and time causality from Eq. 6.2 and Eq. 6.3, we formulate the dual-causal block in RayNova as

$$p\left(X_{1:K}^{1:V,1:T}\right) = \prod_{t=1}^{T} \prod_{k=1}^{K} p\left(X_k^{1:V,t} | X_{1:k-1}^{1:V,1:t}\right). \tag{6.4}$$

It models the joint distribution of multi-view and multi-frame images with both spatial and temporal consistency.

6.3.3 Isotropic Spatio-Temporal Representation

Inductive biases, such as multi-view adjacency or 3D latent features, in previous work limit their generalization in open world and adaptivity to heterogeneous data. Alternatively, we propose an isotropic 4D position embedding by extending the relative rotary position embedding [188] to the camera ray space shared by all the multi-view and multi-frame visual tokens to enhance spatial and temporal consistency.

To this end, we start from a token-wise Plücker ray [155] for the multi-scale feature maps. For convenience, we define a global 3D coordinate space whose origin is the ego-vehicle position at the first timestep. The extrinsic parameters for each camera view v at each timestep t are denoted as $\mathbf{P}^{\mathbf{v},\mathbf{t}}$, while the intrinsic parameter is denoted as $\mathbf{K}^{v,t}$.[1] For each visual token $\mathbf{x}_k^{v,t}$ at scale k, it is straightforward to compute the corresponding Plücker ray $\mathbf{p}_k^{v,t}$ based on the camera parameters and its location in the image space. We further abuse this notation by appending an extra timestep t, so the extended Plücker ray is written as

$$\mathbf{p}_k^{v,t} = (\mathbf{m}_k^{v,t} \in \mathbb{R}^3, \mathbf{d}_k^{v,t} \in \mathbb{R}^3, t) \tag{6.5}$$

where $\mathbf{m}_k^{v,t} = \mathbf{o}^{v,t} \times \mathbf{d}_k^{v,t}$, camera center $\mathbf{o}^{v,t}$ is in the global coordinate system, and unit vector $\mathbf{d}_k^{v,t}$ is the ray direction.

[1] Although the intrinsic parameters are commonly fixed along the temporal dimension, we keep the superscript t to tolerate special varying cases.

Despite an intuitive approach that attaches the encoded Plücker ray to the corresponding visual token [96, 27], this absolute position embedding has limited extrapolation ability beyond the scene range in the training stage, thereby hindering the model's generalization. To this end, we introduce *relative embedding* that conditions networks on the relative position between camera rays using the modified self-attention blocks, where the attention score logit between two visual tokens is calculated as:

$$a_{i,j} = \left(\mathbf{R}_{k_i}^{v_i,t_i}\mathbf{q}_{k_i}^{v_i,t_i}\right)^T \left(\mathbf{R}_{k_j}^{v_j,t_j}\mathbf{k}_{k_j}^{v_j,t_j}\right) = \mathbf{q}_{k_i}^{v_i,t_i\,T}\mathbf{R}_{\Delta}^{i,j}\mathbf{k}_{k_j}^{v_j,t_j} \tag{6.6}$$

where $\mathbf{R}_{\Delta}^{i,j} = \mathbf{R}_{k_i}^{v_i,t_i\,T}\mathbf{R}_{k_j}^{v_j,t_j}$ reflects the relative position between visual token i and j in the camera ray space. We derive $\mathbf{R}_k^{v,t}$ by extending RoPE to 7D space based on the Plücker ray (Eq. 6.5). For clarity, we omit the superscripts and subscripts for scale k, camera view v, and timestep t. The rotary position encoding is written as:

$$\mathbf{R} = \begin{bmatrix} \mathbf{R_m} & 0 & 0 \\ 0 & \mathbf{R_d} & 0 \\ 0 & 0 & \mathrm{RoPE}_{\frac{d}{4}}(t) \end{bmatrix} \tag{6.7}$$

where $\mathrm{RoPE}_{\frac{d}{4}}(t) \in \mathbb{R}^{\frac{d}{4}\times\frac{d}{4}}$ is the 1D rotary position embedding [188] for time step and d is head dimension in the multi-head self-attention module [204]. $\mathbf{R_m}$ and $\mathbf{R_d}$ refer to the rotary position encoding for $\mathbf{m}$ and $\mathbf{d}$ terms in the Plücker ray. Given their small scales, we multiply a scalar factor to each of them separately.

$$\begin{aligned} \mathbf{R_m} &= \begin{bmatrix} \mathrm{RoPE}_{\frac{d}{8}}(\mathbf{m}_1) & 0 & 0 \\ 0 & \mathrm{RoPE}_{\frac{d}{8}}(\mathbf{m}_2) & 0 \\ 0 & 0 & \mathrm{RoPE}_{\frac{d}{8}}(\mathbf{m}_3) \end{bmatrix} \\ \mathbf{R_d} &= \begin{bmatrix} \mathrm{RoPE}_{\frac{d}{8}}(\mathbf{d}_1) & 0 & 0 \\ 0 & \mathrm{RoPE}_{\frac{d}{8}}(\mathbf{d}_2) & 0 \\ 0 & 0 & \mathrm{RoPE}_{\frac{d}{8}}(\mathbf{d}_3) \end{bmatrix} \end{aligned} \tag{6.8}$$

This relative position embedding considers visual tokens from all scales, views, and frames in a unified 4D space with minimal inductive biases, enabling generalization over long ranges and horizons. The model can adaptively learn interactions between visual tokens across spatial and temporal domains, and accommodate heterogeneous camera parameters and frame rates in training and inference.

6.3.4 Dual-Causal Block Architecture Design

We build a transformer-based framework for high-quality multi-view video generation upon a pretrained "next-scale prediction" image generation model [61] with minimal 3D inductive bias. In each transformer block, visual tokens are sequentially passed through an *image-wise*

self-attention module for image realism, a *global self-attention module* for spatio-temporal consistency, and an *image-wise cross-attention module* for conditional fidelity, as shown in Fig. 6.2.

Image-wise self-attention for image realism. The image-wise attention follows the design in Infinity [61]. Each image is processed independently, conditioned on its prefix scales, regardless of frames and camera views. Axial 2D RoPE [69] on the image space is adopted, which is normalized to a specific resolution for multi-scale features.

Global self-attention for spatio-temporal consistency. Previous methods [224, 226, 172] often apply separate spatial and temporal attention to reduce computational complexity. However, this decoupled design includes some prior assumptions about the camera motion and scene dynamics. In contrast, we propose a unified global attention based on the formulation in Eq. 6.4 where each visual token attends to all its prefix in both scale-wise and temporal topological orders from all the camera views (Fig. 6.3). The relative position embedding explained in Sec. 6.3.3 is also employed in this module to serve as a unified spatio-temporal representation. This module is implemented with masked self-attention during training, while the mask can be eliminated through proper KV-cache during inference.

Image-wise condition alignment. We apply an image-wise cross-attention module for input conditions, including text descriptions, 3D object bounding boxes, and HD maps. This module works on each image independently. The text descriptions are processed through a T5 encoder [163]. For each object bounding box per frame, we project its eight corners to camera image space. It is possible that each box may be projected to one or more camera views. The projected coordinates of eight corners in the image space are encoded with MLP and then concatenated with the object category embedding from T5 encoder [163] to obtain a high-dimensional latent feature. For the HD map, we sample several 3D points for each map element. In contrast to directly encoding the structural map element (e.g. polylines or masks), this sampling approach is more flexible when dealing with unstructured or sketched maps. The points are projected to camera image space similar to object bounding box corners. The project coordinates of points belonging to the same map element are encoded with a lightweight PointNet-style network [161], and are then concatenated with the map element category embedding from the T5 encoder [163] to obtain a high-dimensional feature. The latent features for text, object boxes, and HD maps are combined in the cross-attention module, allowing the visual tokens to attend to them simultaneously. For both objects and map conditions, we employ an Axial 2D RoPE [188, 69] between the features and tokens based on their projected centers in the image space. Experiments show that this relative position embedding can notably improve the locality of conditions.

Given the multi-view order-invariant nature of the above modules, we attach a frame-wise local absolute Plücker ray embedding in the beginning of our model to distinguish different views. The image-wise self-attention and cross-attention modules inherit the pretrained

Algorithm 1 Recurrent Training for Long-Horizon Video Generation

Require: Number of video frames T
Require: Maximal cache length M
Require: Multi-view video generation model returning latent feature cache G_θ^L
Require: KV projection Proj_θ^{KV} (included in G_θ)
while training **do**
  Initialize KV latent feature cache $\mathbf{L}_{KV} \leftarrow [\,]$
  for $t = 1$ to T **do**
    $\tilde{\mathbf{x}}^t \leftarrow \text{Random_Bitwise_Error}(\mathbf{x}^t)$
    // Add random noise in scale-wise features to simulate prediction errors
    $\mathbf{KV} \leftarrow \text{Proj}_\theta^{KV}(\mathbf{L}_{KV})$
    $\hat{\mathbf{x}}^t, \mathbf{l}_{KV}^t \leftarrow G_\theta^L(\tilde{\mathbf{x}}^t; \mathbf{KV})$
    if $\text{length}(\mathbf{L}_{KV}) > M$ **then**
      $\mathbf{L}_{KV}.\text{pop}(0)$
    end if
    $\mathbf{L}_{KV}.\text{append}(\text{stop_grad}(\mathbf{l}_{KV}^t))$
    // Cache latents instead of KV to enable gradients for Proj in past frames
    $loss^t \leftarrow \text{Cross_Entropy}(\hat{\mathbf{x}}^t, \mathbf{x}^t)$
    Accumulate gradient of $loss^t$
    // Per-frame backpropagation to save VRAM
  end for
  Update parameter θ
  Clear accumulated gradient
end while

weights from Infinity [61], while the global self-attention module is randomly initialized and combined with zero-initialization [261]. This alternating local-global module can help balance the spatio-temporal consistency and per-image realism.

6.3.5 Recurrent Training for Long-Horizon Videos

Given the GPU memory constraints, our autoregressive model training is limited to a few frames. Although it is extendable during inference through sliding windows, the generation of long-horizon videos suffers from distribution drift due to the gap between training and inference stages.

To bridge this gap, we propose a novel training paradigm to align the training and inference distributions (Alg. 1). It recurrently conducts forward and backward propagation frame by frame, and the gradients are accumulated until the end of each long sequence for model optimization. Obviously, the later frames are conditioned on the earlier frames. We cache the latent features in the global self-attention module since it is the only module operating across frames (Sec. 6.3.4). While previous works often cache the keys and values

Table 6.1: Multi-view Video Generation Performance.

Method	Resolution	Metrics		
		FID↓	FVD↓	Throughput↑
MagicDrive [51]	224×400	16.2	-	1.76
X-Drive [235]	224×400	16.0	-	0.83
DriveDreamer [218]	256×448	14.9	341	0.37
BEVWorld [263]	-	19.0	154	-
Panacea [226]	256×512	17.0	139	0.67
DrivingDiffusion [108]	512×512	15.8	332	-
S4-Driver	384×672	**10.5**	**91**	**1.96**

separately [39, 182] during inference, we directly store the latent features before KV projection layer instead. In this case, we can 1) save half of the GPU memory and 2) ensure the KV projection layer in the computational graph, which is critical for temporal information extraction.

However, there is an error accumulation issue during inference since the “next-scale prediction" framework takes ground-truth visual tokens as inputs during training. To this end, we draw inspiration from Infinity [61] to randomly incorporate some mistakes in the visual token inputs during training to simulate the prediction errors during inference.

6.4 Experiments

6.4.1 Experimental Setups

Datasets. S4-Driver can scale up to heterogeneous training data with varying camera configurations, resolutions, or frame rates. We combine two public datasets (nuScenes [18] and nuPlan [17]), which are converted to a unified ScenarioNet [107] format. NuScenes [18] dataset contains ∼5 hours of driving data collected by 6 cameras with bounding box and map annotations, and scene descriptions from OmniDrive [214]. For nuPlan [17] dataset, we filter the entire dataset based on complete sensor coverage and extract ∼55 hours of data from 8 cameras with auto-labeled bounding boxes and maps. We apply GPT-4o mini [195] for scene descriptions. S4-Driver can be further scaled to larger-scale data, with details in the supplementary materials.

Evaluation Metrics. To evaluate synthetic image quality, we adopt Frechet Video Distance (FVD) [202] and Frechet Inception Distance (FID) [70]. We also evaluate fidelity to conditions through perception models using NDS for objects and mIoU for map. Unless otherwise specified, we conduct the evaluation on the nuScenes validation set [18].

Implementation Details. We build S4-Driver on two variants of the pretrained Infinity [61] model with 130M and 2.8B parameters respectively. Our training pipeline includes three stages: 1) We start from training the model on low-resolution (192×336) multi-view short clips 2) The model is further finetuned on short clips of high-resolution (384×672) multi-view images. 3) Finally, we adopt the recurrent training in Sec. 6.3.5 on long sequence of driving scenarios to learn the long-term temporal coherence. All the training is conducted on 32 NVIDIA A100 GPUs.

6.4.2 Video Generation Performance

Results and Comparisons. In Tab. 6.1, we compare the multi-view generation performance of S4-Driver with other existing world models. We do not include in-house data in our model training, which makes our training data scale less than most of other approaches. Results show that we attain FID and FVD much better than all the baselines, demonstrating better image quality and temporal coherence. Notably, S4-Driver delivers a throughput of **1.96 images/s**, much faster than diffusion baselines, demonstrating the efficiency of our autoregressive architecture. It is also worth mentioning that our model applies an image-based VAE encoder-decoder to minimize the inductive bias relevant to camera setups and ego-motions, although this design choice may hurt the FID and FVD metric. We also provide some qualitative visualizations in Fig. 6.1. It can be observed that our model can generate under different conditions. Using dual-causal autoregression, S4-Driver not only matches or exceeds the performance of models trained on orders of magnitude more private data, but also operates at higher frame rates.

Object & Map Conditions. Our model takes optional object and map inputs as extra local geometric conditions to control the multi-view video generation. In addition to qualitative visualization in Fig. 6.1, we provide some additional quantitative results of object and map condition fidelity. We apply some state-of-the-art perception models [212, 236, 128] to our synthetic images in nuScenes validation set. Both vision-based and multi-modal (combined with ground-truth point clouds) perception models are considered in the evaluation to comprehensively evaluate both the semantic and geometric realism. It is worth mentioning that StreamPETR takes long sequences as input, requiring not only object realism in each single frame but also temporal coherence over long sequences. In Tab. 6.2 and 6.3, our synthetic images show great fidelity to both objects and maps notably better than baselines. For map condition, the projection of coordinates to camera space is not perfect due to the lack of height information in the map annotation, but S4-Driver still shows robust performance. For object condition, our synthetic images can reach a performance similar to real images.

Motion Cues in Synthetic Videos. In addition to visual quality, we further evaluate whether our synthetic videos support plausible driving decisions by applying an end-to-end planner (VAD [93]) pretrained on real scenes to our generated videos. Tab. 6.4 shows that the pretrained planner derives actions from synthetic videos consistent with real scenes. This reflects not only visual realism but also physical and motion plausibility in our generated videos.

Table 6.2: Fidelity to Object Condition

(a) Camera (StreamPETR[212])

Method	**NDS↑**
Oracle	46.9
Panacea [226]	32.1 (68%)
S4-Driver	**41.9** (89%)

(b) Multisensor (SparseFusion[236])

Method	**NDS↑**
Oracle	72.8
X-Drive [235]	69.6 (95%)
S4-Driver	**72.0** (99%)

Table 6.3: Fidelity to Map Condition

(a) Camera (BEVFusion-C[128])

Method	**mIoU↑**
Oracle	57.1
MagicDrive [51]	28.9 (51%)
S4-Driver	**34.9** (61%)

(b) Multisensor (BEVFusion[128])

Method	**mIoU↑**
Oracle	63.0
MagicDrive [51]	47.0 (75%)
S4-Driver	**49.9** (79%)

Table 6.4: Motion Cues in Generated Videos

Method	**Motion Planning** ($L_2 \downarrow$)		
	$1s$	$2s$	$3s$
Oracle	0.41	0.70	1.05
S4-Driver	**0.42** (98%)	**0.73** (96%)	**1.09** (96%)

Zero-shot to Unseen Camera Configurations. The ray-level relative position embedding allows S4-Driver to generalize to unseen camera configurations in a zero-shot manner. An example of sensor setups different from nuScenes and nuPlan is visualized in Fig. 6.1 and the appendix.

6.4.3 Novel View Synthesis

The relative position embedding in camera ray space allows our model to synthesize images for novel views. We follow [211] to evaluate the performance of S4-Driver for novel view synthesis task on nuScenes validation set. Different shifts $\{1m, 2m, 4m\}$ are applied to the camera viewpoints. We measure the FID and FVD metrics between the synthetic shifted multi-view videos and the original real videos. In Tab. 6.5, our FID and FVD metrics only

Table 6.5: Novel View Synthesis Performance with Camera Shifts.

Method	Shift $1m$		Shift $2m$		Shift $4m$	
	FID↓	FVD↓	FID↓	FVD↓	FID↓	FVD↓
StreetGaussian [246]	32.12	153.45	43.24	256.91	67.44	429.98
OmniRe [33]	31.48	152.01	43.31	254.52	67.36	428.20
FreeVS [211]	51.26	431.99	62.04	497.37	77.14	556.14
S4-Driver	**14.11**	**117.20**	**15.58**	**122.60**	**17.48**	**143.48**

Table 6.6: Ablation Study on Camera Ray.

S.-T. Module	Pos. Emb.	FID↓	FVD↓
✗	-	18.7	214
✓	Absolute	**17.2**	138
✓	Relative	**17.2**	**124**

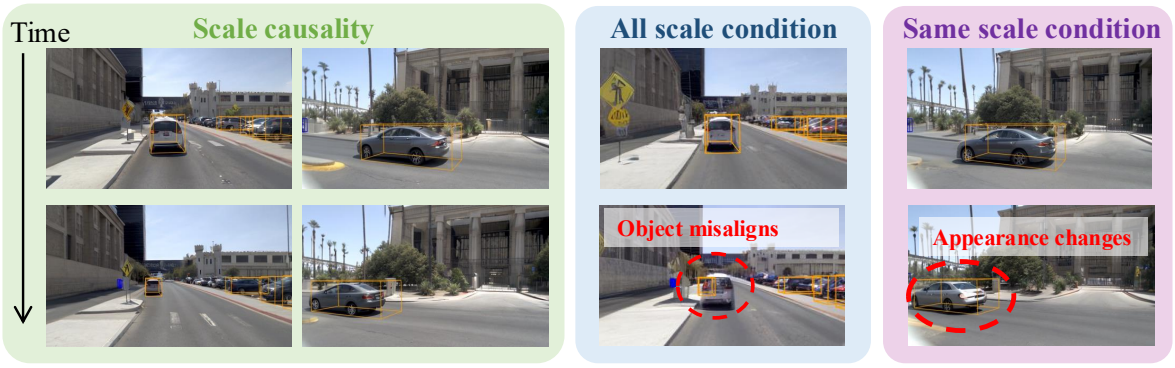


Figure 6.4: Ablation Study on Scale Causality. Conditioning on all scales in history hurts the modeling of dynamics, while conditioning only on same scale is insufficient for temporal coherence.

drop slightly after viewpoint shift. Compared to baselines, S4-Driver exhibits a significant performance gain even if it does not include any 3D inductive bias such as radiance field or point clouds. These results demonstrate the great capability of S4-Driver in camera controllability and novel view synthesis in a geometry-free data-driven manner.

Table 6.7: Ablation Study on Scale Causality and Model Size.

Past-frame Condition	Size	FID↓	FVD↓
same scale	130M	20.7	151
all scales	130M	60.4	454
prefix scales	130M	17.2	124
prefix scales	2.8B	**10.5**	**91**

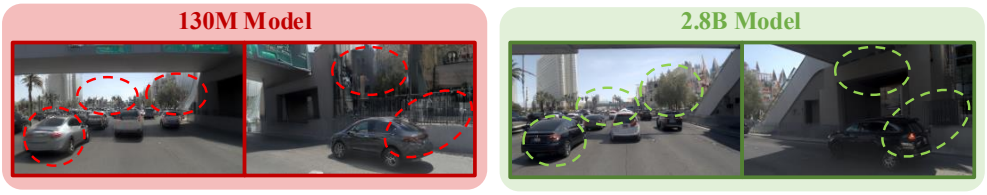


Figure 6.5: Ablation Study on Model Size. Large scale model can bring significantly better visual quality.

6.4.4 Ablation Study

Scale Causality. S4-Driver is built up on scale and time dual-causality. For scale causality (Eq. 6.4), different from low-scale condition in previous frames, we consider two alternatives conditions: 1) all scales or 2) same scale features in previous frames. Results are shown in Tab. 6.7 and Fig. 6.4. Higher scales features of previous time steps cause future frames simply copying the details without simulating the dynamics, and same scale condition is far from enough for stable spatio-temporal modeling.

Relative Ray Embedding. One of our key contributions lies in the relative position encoding in the camera ray space, which is critical for the spatio-temporal consistency. To justify our design, we consider several alternatives in Tab. 6.6. We find that this relative position embedding can perform better than absolute Plücker ray embedding, let alone none spatio-temporal reliance.

Recurrent Training. The recurrent training stage is greatly helpful for temporal coherence in video generation. It improves FVD from 100 to **91** within hundreds of iterations.

Model Size. Our minimal reliance on inductive bias provides a data-driven framework that benefits larger models. We compare two model variants with 130M and 2.8B parameters. In Fig. 6.5 and Tab. 6.7, with the same training data, larger model enables much better image quality.

6.5 Conclusion

This paper introduces RayNova, a versatile autoregressive 4D world foundation model. Built from dual-causal blocks, RayNova performs unified 4D spatio-temporal reasoning by autoregressing along both the *scale* and *time* topological orders. To enhance spatio-temporal consistency, we design a novel isotropic relative position embedding in the continuous 7D Plücker ray space, which represents multi-view, multi-frame, and multi-scale visual features in a unified manner. To mitigate distribution drift in long-horizon video generation, we further adopt a recurrent training paradigm that better aligns the distributions of training and inference. Experiments demonstrate that RayNova can generate realistic multi-view videos with high visual fidelity, strong spatio-temporal coherence, and low latency under diverse conditioning signals, while generalizing to novel views and unseen camera configurations.

Chapter 7

Final Words

7.1 Summary of Contributions

This dissertation studies autonomous driving through the lens of scalable world representations. The central goal is to understand how the 3D driving environment should be represented so that a learning system can perceive the current scene, make decisions from visual observations, and model the future evolution of the world. Rather than proposing a single unified architecture, the dissertation approaches this question through a sequence of methods that together form a coherent progression across perception, planning, and world modeling.

Chapter 2 addresses multi-sensor perception through SparseFusion, showing that efficient scene understanding benefits from representations aligned with the intrinsic sparsity of driving environments. Chapter 3 then studies unsupervised representation learning and demonstrates that temporal coherence is a key property of transferable visual representations for driving. Chapter 4 moves to end-to-end decision-making and shows that planning from raw visual inputs requires structured spatio-temporal world representations. Chapter 5 extends the thesis to generative modeling, introducing a framework for synthesizing aligned LiDAR point clouds and multi-view images with cross-modality consistency. Finally, Chapter 6 studies full-scene world modeling through geometry-agnostic multi-view video generation, bringing the dissertation to the setting of future-scene imagination.

Taken together, these chapters support a common thesis: progress in autonomous driving depends on better ways of representing the world. Across the dissertation, the proposed methods show that spatially structured and temporally coherent world representations can improve both the effectiveness and the scalability of autonomous driving systems.

7.2 Future Work

Several promising directions naturally follow from this dissertation. First, an important next step is to connect the representations developed here within a more unified framework. The methods in this thesis study perception, representation learning, planning, and world mod-

eling separately, but a stronger long-term direction is to build a shared architecture in which these capabilities become different interfaces to the same underlying world representation.

Second, future work can further explore the closed loop between perception, planning, and world modeling. World models can potentially serve not only as generative tools, but also as simulation engines for evaluating and improving perception and planning systems under realistic and controllable scenarios. Integrating these modules into a closed-loop learning system is therefore a promising direction.

Third, it is worth exploring how to better exploit out-of-domain data and foundation models. As large-scale multimodal data and general-purpose models continue to improve, autonomous driving may benefit substantially from more effective transfer of external knowledge into domain-specific scene understanding and decision-making.

Finally, improving efficiency remains essential for real-world deployment. Many current methods for perception, planning, and world modeling rely on large neural architectures with substantial computational and memory cost. Future work should therefore study compact representations, efficient architectures, and deployment-oriented optimization strategies so that these models can better satisfy the practical constraints of onboard inference, energy consumption, and real-time operation in autonomous vehicles.

7.3 Concluding Remarks

At its core, this dissertation argues that autonomous driving should be approached not only as a collection of separate tasks, but also as a problem of world representation. Whether the goal is detecting objects, planning an action, or modeling future scenes, success depends on how effectively the structure of the environment is captured and preserved.

The methods developed in this dissertation explore this idea from multiple angles, but they share a common direction towards more scalable, and more reusable representations of the driving world. It is our hope that these contributions help clarify the importance of representation design in autonomous driving and provide useful building blocks for future systems that are more robust, more data-efficient, and more capable of understanding and modeling the world they operate in.

Bibliography

[1] Josh Achiam et al. "Gpt-4 technical report". In: *arXiv preprint arXiv:2303.08774* (2023).

[2] Niket Agarwal et al. "Cosmos world foundation model platform for physical ai". In: *arXiv preprint arXiv:2501.03575* (2025).

[3] Jean-Baptiste Alayrac et al. "Flamingo: a visual language model for few-shot learning". In: *Advances in neural information processing systems* 35 (2022), pp. 23716–23736.

[4] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. "Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond". In: *arXiv preprint arXiv:2308.12966* 1.2 (2023), p. 3.

[5] Xuyang Bai, Zeyu Hu, Xinge Zhu, Qingqiu Huang, Yilun Chen, Hongbo Fu, and Chiew-Lan Tai. "Transfusion: Robust lidar-camera fusion for 3d object detection with transformers". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 1090–1099.

[6] Yutong Bai, Xinlei Chen, Alexander Kirillov, Alan Yuille, and Alexander C Berg. "Point-level region contrast for object detection pre-training". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 16061–16070.

[7] Yutong Bai, Xinyang Geng, Karttikeya Mangalam, Amir Bar, Alan L Yuille, Trevor Darrell, Jitendra Malik, and Alexei A Efros. "Sequential modeling enables scalable learning for large vision models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 22861–22872.

[8] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. "Simple online and realtime tracking". In: *2016 IEEE international conference on image processing (ICIP)*. IEEE. 2016, pp. 3464–3468.

[9] Lucas Beyer et al. "Paligemma: A versatile 3b vlm for transfer". In: *arXiv preprint arXiv:2407.07726* (2024).

[10] Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. "Align your latents: High-resolution video synthesis with latent diffusion models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 22563–22575.

[11] Andreas Blattmann et al. "Stable Video Diffusion: Scaling Latent Video Diffusion Models to Large Datasets". In: *arXiv preprint arXiv:2311.15127* (2023).

[12] Garrick Brazil, Abhinav Kumar, Julian Straub, Nikhila Ravi, Justin Johnson, and Georgia Gkioxari. "Omni3D: A Large Benchmark and Model for 3D Object Detection in the Wild". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 13154–13164.

[13] Tim Brooks, Aleksander Holynski, and Alexei A Efros. "Instructpix2pix: Learning to follow image editing instructions". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 18392–18402.

[14] Tim Brooks et al. "Video generation models as world simulators". In: *OpenAI Blog* 1.8 (2024), p. 1.

[15] Jake Bruce et al. "Genie: Generative interactive environments". In: *Forty-first International Conference on Machine Learning*. 2024.

[16] Lucas Caccia, Herke Van Hoof, Aaron Courville, and Joelle Pineau. "Deep generative modeling of lidar data". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2019, pp. 5034–5040.

[17] Holger Caesar, Juraj Kabzan, Kok Seang Tan, Whye Kit Fong, Eric Wolff, Alex Lang, Luke Fletcher, Oscar Beijbom, and Sammy Omari. *NuPlan: A closed-loop ML-based planning benchmark for autonomous vehicles*. 2022. arXiv: 2106.11810 [cs.CV]. URL: https://arxiv.org/abs/2106.11810.

[18] Holger Caesar et al. *nuScenes: A multimodal dataset for autonomous driving*. 2020. arXiv: 1903.11027 [cs.LG].

[19] Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. "Density-based clustering based on hierarchical density estimates". In: *Pacific-Asia conference on knowledge discovery and data mining*. Springer. 2013, pp. 160–172.

[20] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. "End-to-end object detection with transformers". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*. Springer. 2020, pp. 213–229.

[21] Sergio Casas, Abbas Sadat, and Raquel Urtasun. "Mp3: A unified model to map, perceive, predict and plan". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 14403–14412.

[22] Moitreya Chatterjee and Anoop Cherian. "Sound2sight: Generating visual dynamics from sound and context". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVII 16*. Springer. 2020, pp. 701–719.

[23] Anthony Chen, Wenzhao Zheng, Yida Wang, Xueyang Zhang, Kun Zhan, Peng Jia, Kurt Keutzer, and Shanghang Zhang. "Geodrive: 3d geometry-informed driving world model with precise action control". In: *arXiv preprint arXiv:2505.22421* (2025).

[24] Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. "Spatialvlm: Endowing vision-language models with spatial reasoning capabilities". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 14455–14465.

[25] Lele Chen, Sudhanshu Srivastava, Zhiyao Duan, and Chenliang Xu. "Deep cross-modal audio-visual generation". In: *Proceedings of the on Thematic Workshops of ACM Multimedia 2017*. 2017, pp. 349–357.

[26] Long Chen, Oleg Sinavski, Jan Hünermann, Alice Karnsund, Andrew James Willmott, Danny Birch, Daniel Maund, and Jamie Shotton. "Driving with llms: Fusing object-level vector modality for explainable autonomous driving". In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2024, pp. 14093–14100.

[27] Rui Chen, Zehuan Wu, Yichen Liu, Yuxin Guo, Jingcheng Ni, Haifeng Xia, and Siyu Xia. "Unimlvg: Unified framework for multi-view long video generation with comprehensive control capabilities for autonomous driving". In: *arXiv preprint arXiv:2412.04842* (2024).

[28] Xi Chen et al. "Pali-3 vision language models: Smaller, faster, stronger". In: *arXiv preprint arXiv:2310.09199* (2023).

[29] Xi Chen et al. "PaLI: A Jointly-Scaled Multilingual Language-Image Model". In: *The Eleventh International Conference on Learning Representations*.

[30] Xuanyao Chen, Tianyuan Zhang, Yue Wang, Yilun Wang, and Hang Zhao. "Futr3d: A unified sensor fusion framework for 3d detection". In: *arXiv preprint arXiv:2203.10642* (2022).

[31] Zehui Chen, Zhenyu Li, Shiquan Zhang, Liangji Fang, Qinhong Jiang, and Feng Zhao. "Autoalignv2: Deformable feature aggregation for dynamic multi-modal 3d object detection". In: *arXiv preprint arXiv:2207.10316* (2022).

[32] Ziyu Chen et al. *OmniRe: Omni Urban Scene Reconstruction*. 2025. arXiv: 2408.16760 [cs.CV]. URL: https://arxiv.org/abs/2408.16760.

[33] Ziyu Chen et al. "Omnire: Omni urban scene reconstruction". In: *arXiv preprint arXiv:2408.16760* (2024).

[34] An-Chieh Cheng, Hongxu Yin, Yang Fu, Qiushan Guo, Ruihan Yang, Jan Kautz, Xiaolong Wang, and Sifei Liu. "SpatialRGPT: Grounded Spatial Reasoning in Vision Language Model". In: *arXiv preprint arXiv:2406.01584* (2024).

[35] Hyung Won Chung et al. "Scaling instruction-finetuned language models". In: *Journal of Machine Learning Research* 25.70 (2024), pp. 1–53.

[36] Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. "End-to-end driving via conditional imitation learning". In: *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2018, pp. 4693–4700.

[37] MMDetection3D Contributors. *MMDetection3D: OpenMMLab next-generation platform for general 3D object detection*. https://github.com/open-mmlab/mmdetection3d. 2020.

[38] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. *InstructBLIP: Towards General-purpose Vision-Language Models with Instruction Tuning*. 2023. arXiv: 2305.06500 [cs.CV].

[39] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. "Transformer-xl: Attentive language models beyond a fixed-length context". In: *arXiv preprint arXiv:1901.02860* (2019).

[40] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang, and Houqiang Li. "Voxel r-cnn: Towards high performance voxel-based 3d object detection". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 2. 2021, pp. 1201–1209.

[41] Tianchen Deng et al. "GaussianDWM: 3D Gaussian Driving World Model for Unified Scene Understanding and Multi-Modal Generation". In: *arXiv preprint arXiv:2512.23180* (2025).

[42] Tianchen Deng et al. "What is the best 3d scene representation for robotics? from geometric to foundation models". In: *arXiv preprint arXiv:2512.03422* (2025).

[43] Prafulla Dhariwal and Alexander Nichol. "Diffusion models beat gans on image synthesis". In: *Advances in neural information processing systems* 34 (2021), pp. 8780–8794.

[44] Xinpeng Ding, Jianhua Han, Hang Xu, Xiaodan Liang, Wei Zhang, and Xiaomeng Li. "Holistic Autonomous Driving Understanding by Bird's-Eye-View Injected Multi-Modal Large Models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 13668–13677.

[45] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. "CARLA: An open urban driving simulator". In: *Conference on robot learning*. PMLR. 2017, pp. 1–16.

[46] Alexey Dosovitskiy et al. "An image is worth 16x16 words: Transformers for image recognition at scale". In: *arXiv preprint arXiv:2010.11929* (2020).

[47] Scott Ettinger et al. "Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 9710–9719.

[48] Lue Fan, Xuan Xiong, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. "Rangedet: In defense of range view for lidar-based 3d object detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 2918–2927.

[49] Yuxin Fang, Wen Wang, Binhui Xie, Quan Sun, Ledell Wu, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. "Eva: Exploring the limits of masked visual representation learning at scale". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 19358–19369.

[50] Ruiyuan Gao, Kai Chen, Zhihao Li, Lanqing Hong, Zhenguo Li, and Qiang Xu. "Magicdrive3d: Controllable 3d generation for any-view rendering in street scenes". In: *arXiv preprint arXiv:2405.14475* (2024).

[51] Ruiyuan Gao, Kai Chen, Enze Xie, Lanqing Hong, Zhenguo Li, Dit-Yan Yeung, and Qiang Xu. "Magicdrive: Street view generation with diverse 3d geometry control". In: *arXiv preprint arXiv:2310.02601* (2023).

[52] Songwei Ge, Thomas Hayes, Harry Yang, Xi Yin, Guan Pang, David Jacobs, Jia-Bin Huang, and Devi Parikh. "Long Video Generation with Time-Agnostic VQGAN and Time-Sensitive Transformer". In: *ECCV* (2022).

[53] Songwei Ge et al. *Preserve Your Own Correlation: A Noise Prior for Video Diffusion Models*. 2024. arXiv: 2305.10474 [cs.CV]. URL: https://arxiv.org/abs/2305.10474.

[54] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. "Digging into self-supervised monocular depth estimation". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 3828–3838.

[55] Jean-Bastien Grill et al. "Bootstrap your own latent-a new approach to self-supervised learning". In: *Advances in neural information processing systems* 33 (2020), pp. 21271–21284.

[56] Junru Gu, Chenxu Hu, Tianyuan Zhang, Xuanyao Chen, Yilun Wang, Yue Wang, and Hang Zhao. "ViP3D: End-to-end visual trajectory prediction via 3d agent queries". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 5496–5506.

[57] Hang Guo, Yawei Li, Taolin Zhang, Jiangshan Wang, Tao Dai, Shu-Tao Xia, and Luca Benini. "Fastvar: Linear visual autoregressive modeling via cached token pruning". In: *arXiv preprint arXiv:2503.23367* (2025).

[58] Jiazhe Guo et al. *DiST-4D: Disentangled Spatiotemporal Diffusion with Metric Depth for 4D Driving Scene Generation*. 2025. arXiv: 2503.15208 [cs.CV]. URL: https://arxiv.org/abs/2503.15208.

[59] Agrim Gupta, Jiajun Wu, Jia Deng, and Li Fei-Fei. "Siamese Masked Autoencoders". In: *arXiv preprint arXiv:2305.14344* (2023).

[60] David Ha and Jürgen Schmidhuber. "World models". In: *arXiv preprint arXiv:1803.10122* (2018).

[61] Jian Han, Jinlai Liu, Yi Jiang, Bin Yan, Yuqi Zhang, Zehuan Yuan, Bingyue Peng, and Xiaobing Liu. "Infinity: Scaling bitwise autoregressive modeling for high-resolution image synthesis". In: *arXiv preprint arXiv:2412.04431* (2024).

[62] Tengda Han, Weidi Xie, and Andrew Zisserman. "Self-supervised co-training for video representation learning". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 5679–5690.

[63] Wangli Hao, Zhaoxiang Zhang, and He Guan. "Cmcgan: A uniform framework for cross-modal visual-audio mutual generation". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 32. 2018.

[64] Adam W Harley, Zhaoyuan Fang, Jie Li, Rares Ambrus, and Katerina Fragkiadaki. "Simple-bev: What really matters for multi-sensor bev perception?" In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2023, pp. 2759–2765.

[65] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. "Masked autoencoders are scalable vision learners". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 16000–16009.

[66] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. "Momentum contrast for unsupervised visual representation learning". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 9729–9738.

[67] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. "Mask r-cnn". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2961–2969.

[68] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[69] Byeongho Heo, Song Park, Dongyoon Han, and Sangdoo Yun. "Rotary position embedding for vision transformer". In: *European Conference on Computer Vision*. Springer. 2024, pp. 289–305.

[70] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. "Gans trained by a two time-scale update rule converge to a local nash equilibrium". In: *Advances in neural information processing systems* 30 (2017).

[71] Michael Himmelsbach, Felix V Hundelshausen, and H-J Wuensche. "Fast segmentation of 3D point clouds for ground vehicles". In: *2010 IEEE Intelligent Vehicles Symposium*. IEEE. 2010, pp. 560–565.

[72] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models". In: *Advances in neural information processing systems* 33 (2020), pp. 6840–6851.

[73] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. "The curious case of neural text degeneration". In: *arXiv preprint arXiv:1904.09751* (2019).

[74] Yicong Hong et al. "Lrm: Large reconstruction model for single image to 3d". In: *arXiv preprint arXiv:2311.04400* (2023).

[75] Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. "3d-llm: Injecting the 3d world into large language models". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 20482–20494.

[76] Anthony Hu, Zak Murez, Nikhil Mohan, Sofía Dudas, Jeffrey Hawke, Vijay Badrinarayanan, Roberto Cipolla, and Alex Kendall. "Fiery: Future instance prediction in bird's-eye view from surround monocular cameras". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 15273–15282.

[77] Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. *GAIA-1: A Generative World Model for Autonomous Driving*. 2023. arXiv: 2309.17080 [cs.CV]. URL: https://arxiv.org/abs/2309.17080.

[78] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. "Lora: Low-rank adaptation of large language models". In: *arXiv preprint arXiv:2106.09685* (2021).

[79] Qianjiang Hu, Zhimin Zhang, and Wei Hu. "RangeLDM: Fast Realistic LiDAR Point Cloud Generation". In: *arXiv preprint arXiv:2403.10094* (2024).

[80] Shengchao Hu, Li Chen, Penghao Wu, Hongyang Li, Junchi Yan, and Dacheng Tao. "St-p3: End-to-end vision-based autonomous driving via spatial-temporal feature learning". In: *European Conference on Computer Vision*. Springer. 2022, pp. 533–549.

[81] Wenbo Hu, Xiangjun Gao, Xiaoyu Li, Sijie Zhao, Xiaodong Cun, Yong Zhang, Long Quan, and Ying Shan. "DepthCrafter: Generating Consistent Long Depth Sequences for Open-world Videos". In: *arXiv preprint arXiv:2409.02095* (2024).

[82] Yihan Hu et al. "Planning-oriented autonomous driving". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 17853–17862.

[83] Chenxi Huang, Tong He, Haidong Ren, Wenxiao Wang, Binbin Lin, and Deng Cai. "OBMO: One Bounding Box Multiple Objects for Monocular 3D Object Detection". In: *arXiv preprint arXiv:2212.10049* (2022).

[84] Junjie Huang and Guan Huang. "Bevdet4d: Exploit temporal cues in multi-camera 3d object detection". In: *arXiv preprint arXiv:2203.17054* (2022).

[85] Junjie Huang, Guan Huang, Zheng Zhu, and Dalong Du. "Bevdet: High-performance multi-camera 3d object detection in bird-eye-view". In: *arXiv preprint arXiv:2112.11790* (2021).

[86] Nan Huang, Xiaobao Wei, Wenzhao Zheng, Pengju An, Ming Lu, Wei Zhan, Masayoshi Tomizuka, Kurt Keutzer, and Shanghang Zhang. *S^3Gaussian: Self-Supervised Street Gaussians for Autonomous Driving*. 2024. arXiv: 2405.20323 [cs.CV]. URL: https://arxiv.org/abs/2405.20323.

[87] Siyuan Huang, Yichen Xie, Song-Chun Zhu, and Yixin Zhu. "Spatio-temporal self-supervised representation learning for 3d point clouds". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 6535–6545.

[88] Jyh-Jing Hwang et al. "Emma: End-to-end multimodal model for autonomous driving". In: *arXiv preprint arXiv:2410.23262* (2024).

[89] Maximilian Igl et al. "Symphony: Learning realistic and diverse agents for autonomous driving simulation". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 2445–2451.

[90] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. "Image-to-image translation with conditional adversarial networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1125–1134.

[91] Allan Jabri, Andrew Owens, and Alexei Efros. "Space-time correspondence as a contrastive random walk". In: *Advances in neural information processing systems* 33 (2020), pp. 19545–19560.

[92] Bo Jiang et al. "Perceive, interact, predict: Learning dynamic and static clues for end-to-end motion prediction". In: *arXiv preprint arXiv:2212.02181* (2022).

[93] Bo Jiang et al. "Vad: Vectorized scene representation for efficient autonomous driving". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 8340–8350.

[94] Hanwen Jiang et al. "RayZer: A Self-supervised Large View Synthesis Model". In: *arXiv preprint arXiv:2505.00702* (2025).

[95] Yang Jiao, Zequn Jie, Shaoxiang Chen, Jingjing Chen, Xiaolin Wei, Lin Ma, and Yu-Gang Jiang. "MSMDFusion: Fusing LiDAR and Camera at Multiple Scales with Multi-Depth Seeds for 3D Object Detection". In: *arXiv preprint arXiv:2209.03102* (2022).

[96] Haian Jin, Hanwen Jiang, Hao Tan, Kai Zhang, Sai Bi, Tianyuan Zhang, Fujun Luan, Noah Snavely, and Zexiang Xu. "Lvsm: A large view synthesis model with minimal 3d inductive bias". In: *arXiv preprint arXiv:2410.17242* (2024).

[97] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. "3d gaussian splatting for real-time radiance field rendering." In: *ACM Trans. Graph.* 42.4 (2023), pp. 139–1.

[98] Jinkyu Kim, Anna Rohrbach, Trevor Darrell, John Canny, and Zeynep Akata. "Textual explanations for self-driving vehicles". In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 563–578.

[99] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[100] Harold W Kuhn. "The Hungarian method for the assignment problem". In: *Naval research logistics quarterly* 2.1-2 (1955), pp. 83–97.

[101] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. "Pointpillars: Fast encoders for object detection from point clouds". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 12697–12705.

[102] Hei Law and Jia Deng. "Cornernet: Detecting objects as paired keypoints". In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 734–750.

[103] Yann LeCun. "A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27". In: *Open Review* 62.1 (2022), pp. 1–62.

[104] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. "Autoregressive image generation using residual quantization". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 11523–11532.

[105] Bo Li, Tianlei Zhang, and Tian Xia. "Vehicle detection from 3d lidar using fully convolutional network". In: *arXiv preprint arXiv:1608.07916* (2016).

[106] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. "Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models". In: *International conference on machine learning*. PMLR. 2023, pp. 19730–19742.

[107] Quanyi Li, Zhenghao Peng, Lan Feng, Zhizheng Liu, Chenda Duan, Wenjie Mo, and Bolei Zhou. "ScenarioNet: Open-Source Platform for Large-Scale Traffic Scenario Simulation and Modeling". In: *Advances in Neural Information Processing Systems* (2023).

[108] Xiaofan Li, Yifu Zhang, and Xiaoqing Ye. "DrivingDiffusion: Layout-Guided Multi-view Driving Scenarios Video Generation with Latent Diffusion Model". In: *European Conference on Computer Vision*. Springer. 2024, pp. 469–485.

[109] Yanwei Li, Yilun Chen, Xiaojuan Qi, Zeming Li, Jian Sun, and Jiaya Jia. "Unifying voxel-based representation with transformer for 3d object detection". In: *arXiv preprint arXiv:2206.00630* (2022).

[110] Yiheng Li, Seth Z Zhao, Chenfeng Xu, Chen Tang, Chenran Li, Mingyu Ding, Masayoshi Tomizuka, and Wei Zhan. "Pre-training on Synthetic Driving Data for Trajectory Prediction". In: *arXiv preprint arXiv:2309.10121* (2023).

[111] Yinhao Li, Zheng Ge, Guanyi Yu, Jinrong Yang, Zengran Wang, Yukang Shi, Jianjian Sun, and Zeming Li. "Bevdepth: Acquisition of reliable depth for multi-view 3d object detection". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. 2. 2023, pp. 1477–1485.

[112] Yuheng Li, Haotian Liu, Qingyang Wu, Fangzhou Mu, Jianwei Yang, Jianfeng Gao, Chunyuan Li, and Yong Jae Lee. "Gligen: Open-set grounded text-to-image generation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 22511–22521.

[113] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. "Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers". In: *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IX*. Springer. 2022, pp. 1–18.

[114] Zhiqi Li, Zhiding Yu, Shiyi Lan, Jiahan Li, Jan Kautz, Tong Lu, and Jose M Alvarez. "Is ego status all you need for open-loop end-to-end autonomous driving?" In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 14864–14873.

[115] Tingting Liang, Xiaojie Chu, Yudong Liu, Yongtao Wang, Zhi Tang, Wei Chu, Jingdong Chen, and Haibin Ling. "Cbnet: A composite backbone network architecture for object detection". In: *IEEE Transactions on Image Processing* 31 (2022), pp. 6893–6906.

[116] Tingting Liang, Hongwei Xie, Kaicheng Yu, Zhongyu Xia, Zhiwei Lin, Yongtao Wang, Tao Tang, Bing Wang, and Zhi Tang. "Bevfusion: A simple and robust lidar-camera fusion framework". In: *arXiv preprint arXiv:2205.13790* (2022).

[117] Zhidong Liang, Ming Zhang, Zehan Zhang, Xian Zhao, and Shiliang Pu. "Rangercnn: Towards fast and accurate 3d object detection with range image representation". In: *arXiv preprint arXiv:2009.00206* (2020).

[118] Bencheng Liao, Shaoyu Chen, Xinggang Wang, Tianheng Cheng, Qian Zhang, Wenyu Liu, and Chang Huang. "MapTR: Structured Modeling and Learning for Online Vectorized HD Map Construction". In: *The Eleventh International Conference on Learning Representations*. 2022.

[119] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. "Feature pyramid networks for object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2117–2125.

[120] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. "Focal loss for dense object detection". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2980–2988.

[121] Zhiwei Lin and Yongtao Wang. "BEV-MAE: Bird's Eye View Masked Autoencoders for Outdoor Point Cloud Pre-training". In: *arXiv preprint arXiv:2212.05758* (2022).

[122] Chang Liu, Rui Li, Kaidong Zhang, Yunwei Lan, and Dong Liu. *StableV2V: Stablizing Shape Consistency in Video-to-Video Editing*. 2024. arXiv: 2411.11045 [cs.CV]. URL: https://arxiv.org/abs/2411.11045.

[123] Hao Liu, Zhuoran Xu, Dan Wang, Baofeng Zhang, Guan Wang, Bo Dong, Xin Wen, and Xinyu Xu. "PAI3D: Painting Adaptive Instance-Prior for 3D Object Detection". In: *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part V*. Springer. 2023, pp. 459–475.

[124] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. "Visual instruction tuning". In: *Advances in neural information processing systems* 36 (2024).

[125] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. "Zero-1-to-3: Zero-shot one image to 3d object". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2023, pp. 9298–9309.

[126] Yingfei Liu, Tiancai Wang, Xiangyu Zhang, and Jian Sun. "Petr: Position embedding transformation for multi-view 3d object detection". In: *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVII*. Springer. 2022, pp. 531–548.

[127] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. "Swin transformer: Hierarchical vision transformer using shifted windows". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 10012–10022.

[128] Zhijian Liu, Haotian Tang, Alexander Amini, Xinyu Yang, Huizi Mao, Daniela L Rus, and Song Han. "Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation". In: *2023 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2023, pp. 2774–2781.

[129] Ilya Loshchilov and Frank Hutter. "Decoupled weight decay regularization". In: *arXiv preprint arXiv:1711.05101* (2017).

[130] Jiachen Lu, Ze Huang, Jiahui Zhang, Zeyu Yang, and Li Zhang. "Wovogen: World volume-aware diffusion for controllable multi-camera driving scene generation". In: *arXiv preprint arXiv:2312.02934* (2023).

[131] Xianzheng Ma et al. "When LLMs step into the 3D World: A Survey and Meta-Analysis of 3D Tasks via Multi-modal Large Language Models". In: *arXiv preprint arXiv:2405.10255* (2024).

[132] Xin Ma, Yaohui Wang, Xinyuan Chen, Gengyun Jia, Ziwei Liu, Yuan-Fang Li, Cunjian Chen, and Yu Qiao. *Latte: Latent Diffusion Transformer for Video Generation*. 2025. arXiv: 2401.03048 [cs.CV]. URL: https://arxiv.org/abs/2401.03048.

[133] Yingzi Ma, Yulong Cao, Jiachen Sun, Marco Pavone, and Chaowei Xiao. "Dolphins: Multimodal language model for driving". In: *arXiv preprint arXiv:2312.00438* (2023).

[134] Jiageng Mao, Yuxi Qian, Junjie Ye, Hang Zhao, and Yue Wang. "Gpt-driver: Learning to drive with gpt". In: *arXiv preprint arXiv:2310.01415* (2023).

[135] Jiageng Mao, Yujing Xue, Minzhe Niu, Haoyue Bai, Jiashi Feng, Xiaodan Liang, Hang Xu, and Chunjing Xu. "Voxel transformer for 3d object detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 3164–3173.

[136] Jiageng Mao, Junjie Ye, Yuxi Qian, Marco Pavone, and Yue Wang. *A Language Agent for Autonomous Driving*. 2023. arXiv: 2311.10813 [cs.CV].

[137] Brandon McKinzie et al. "Mm1: Methods, analysis & insights from multimodal llm pre-training". In: *arXiv preprint arXiv:2403.09611* (2024).

[138] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. "SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations". In: *International Conference on Learning Representations*. 2022.

[139] Vincent Micheli, Eloi Alonso, and François Fleuret. "Transformers are Sample-Efficient World Models". In: *ICLR* (2023).

[140] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. "Nerf: Representing scenes as neural radiance fields for view synthesis". In: *Communications of the ACM* 65.1 (2021), pp. 99–106.

[141] Chen Min, Xinli Xu, Dawei Zhao, Liang Xiao, Yiming Nie, and Bin Dai. "Occ-BEV: Multi-Camera Unified Pre-training via 3D Scene Reconstruction". In: *arXiv preprint arXiv:2305.18829* (2023).

[142] Ishan Misra, Rohit Girdhar, and Armand Joulin. "An end-to-end transformer model for 3d object detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 2906–2917.

[143] Nigamaa Nayakanti, Rami Al-Rfou, Aurick Zhou, Kratarth Goel, Khaled S Refaat, and Benjamin Sapp. "Wayformer: Motion forecasting via simple & efficient attention networks". In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2023, pp. 2980–2987.

[144] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. "Glide: Towards photorealistic image generation and editing with text-guided diffusion models". In: *arXiv preprint arXiv:2112.10741* (2021).

[145] Nvidia. *World foundation model*. https://www.nvidia.com/en-us/glossary/world-models/. 2024.

[146] OpenAI. *Gpt-4v(ision) system card*. 2023.

[147] OpenAI. *Sora: Creating video from text*. 2024. URL: https://openai.com/sora.

[148] Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. "Masked autoencoders for point cloud self-supervised learning". In: *European conference on computer vision*. Springer. 2022, pp. 604–621.

[149] Dennis Park, Rares Ambrus, Vitor Guizilini, Jie Li, and Adrien Gaidon. "Is pseudo-lidar needed for monocular 3d object detection?" In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 3142–3152.

[150] Jinhyung Park, Chenfeng Xu, Shijia Yang, Kurt Keutzer, Kris Kitani, Masayoshi Tomizuka, and Wei Zhan. "Time will tell: New outlooks and a baseline for temporal multi-view 3d object detection". In: *arXiv preprint arXiv:2210.02443* (2022).

[151] William Peebles and Saining Xie. "Scalable diffusion models with transformers". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 4195–4205.

[152] Chensheng Peng, Chengwei Zhang, Yixiao Wang, Chenfeng Xu, Yichen Xie, Wenzhao Zheng, Kurt Keutzer, Masayoshi Tomizuka, and Wei Zhan. "Desire-gs: 4d street gaussians for static-dynamic decomposition and surface reconstruction for urban driving scenes". In: *arXiv preprint arXiv:2411.11921* (2024).

[153] Gabriel Peyré, Marco Cuturi, et al. "Computational optimal transport: With applications to data science". In: *Foundations and Trends® in Machine Learning* 11.5-6 (2019), pp. 355–607.

[154] Jonah Philion and Sanja Fidler. "Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*. Springer. 2020, pp. 194–210.

[155] Julius Plucker. "Xvii. on a new geometry of space". In: *Philosophical Transactions of the Royal Society of London* (1865), pp. 725–791.

[156] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. "Sdxl: Improving latent diffusion models for high-resolution image synthesis". In: *arXiv preprint arXiv:2307.01952* (2023).

[157] Adam Polyak et al. "Movie gen: A cast of media foundation models". In: *arXiv* (2024).

[158] Dean A Pomerleau. "Alvinn: An autonomous land vehicle in a neural network". In: *Advances in neural information processing systems* 1 (1988).

[159] C. Qi, Yin Zhou, Mahyar Najibi, Pei Sun, Khoa T. Vo, Boyang Deng, and Dragomir Anguelov. "Offboard 3D Object Detection from Point Cloud Sequences". In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021), pp. 6130–6140. URL: https://api.semanticscholar.org/CorpusID:232168876.

[160] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. "Frustum pointnets for 3d object detection from rgb-d data". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 918–927.

[161] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. "Pointnet: Deep learning on point sets for 3d classification and segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 652–660.

[162] Alec Radford et al. "Learning transferable visual models from natural language supervision". In: *International conference on machine learning*. PMLR. 2021, pp. 8748–8763.

[163] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. "Exploring the limits of transfer learning with a unified text-to-text transformer". In: *Journal of machine learning research* 21.140 (2020), pp. 1–67.

[164] Haoxi Ran, Vitor Guizilini, and Yue Wang. "Towards Realistic Scene Generation with LiDAR Diffusion Models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 14738–14748.

[165] Cody Reading, Ali Harakeh, Julia Chae, and Steven L Waslander. "Categorical depth distribution network for monocular 3d object detection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 8555–8564.

[166] Weiming Ren, Huan Yang, Ge Zhang, Cong Wei, Xinrun Du, Wenhao Huang, and Wenhu Chen. *ConsistI2V: Enhancing Visual Consistency for Image-to-Video Generation*. 2024. arXiv: 2402.04324 [cs.CV]. URL: https://arxiv.org/abs/2402.04324.

[167] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. "Stochastic backpropagation and approximate inference in deep generative models". In: *International conference on machine learning*. PMLR. 2014, pp. 1278–1286.

[168] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. "High-resolution image synthesis with latent diffusion models". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 10684–10695.

[169] Ludan Ruan, Yiyang Ma, Huan Yang, Huiguo He, Bei Liu, Jianlong Fu, Nicholas Jing Yuan, Qin Jin, and Baining Guo. "Mm-diffusion: Learning multi-modal diffusion models for joint audio and video generation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 10219–10228.

[170] Runway. *Introducing Gen-3 Alpha: A New Frontier for Video Generation*. https://runwayml.com/research/introducing-gen-3-alpha. 2024.

[171] Lloyd Russell, Anthony Hu, Lorenzo Bertoni, George Fedoseev, Jamie Shotton, Elahe Arani, and Gianluca Corrado. *GAIA-2: A Controllable Multi-View Generative World Model for Autonomous Driving*. 2025. arXiv: 2503.20523 [cs.CV]. URL: https://arxiv.org/abs/2503.20523.

[172] Lloyd Russell, Anthony Hu, Lorenzo Bertoni, George Fedoseev, Jamie Shotton, Elahe Arani, and Gianluca Corrado. "Gaia-2: A controllable multi-view generative world model for autonomous driving". In: *arXiv preprint arXiv:2503.20523* (2025).

[173] Avishkar Saha, Oscar Mendez, Chris Russell, and Richard Bowden. "Translating images into maps". In: *2022 International conference on robotics and automation (ICRA)*. IEEE. 2022, pp. 9200–9206.

[174] Mehdi SM Sajjadi, Daniel Duckworth, Aravindh Mahendran, Sjoerd Van Steenkiste, Filip Pavetic, Mario Lucic, Leonidas J Guibas, Klaus Greff, and Thomas Kipf. "Object scene representation transformer". In: *Advances in neural information processing systems* 35 (2022), pp. 9512–9524.

[175] Kyle Sargent et al. "ZeroNVS: Zero-Shot 360-Degree View Synthesis from a Single Image". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 9420–9429.

[176] Stefan Schubert, Peer Neubert, Johannes Pöschmann, and Peter Protzel. "Circular convolutional neural networks for panoramic images and laser data". In: *2019 IEEE intelligent vehicles symposium (IV)*. IEEE. 2019, pp. 653–660.

[177] Ari Seff, Brian Cera, Dian Chen, Mason Ng, Aurick Zhou, Nigamaa Nayakanti, Khaled S Refaat, Rami Al-Rfou, and Benjamin Sapp. "Motionlm: Multi-agent motion forecasting as language modeling". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 8579–8590.

[178] Hao Sha et al. "Languagempc: Large language models as decision makers for autonomous driving". In: *arXiv preprint arXiv:2310.03026* (2023).

[179] Hao Shao, Yuxuan Hu, Letian Wang, Guanglu Song, Steven L Waslander, Yu Liu, and Hongsheng Li. "Lmdrive: Closed-loop end-to-end driving with large language models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 15120–15130.

[180] Jiahao Shao, Yuanbo Yang, Hongyu Zhou, Youmin Zhang, Yujun Shen, Matteo Poggi, and Yiyi Liao. "Learning Temporally Consistent Video Depth from Video Diffusion Priors". In: *arXiv preprint arXiv:2406.01493* (2024).

[181] Hualian Sheng, Sijia Cai, Yuan Liu, Bing Deng, Jianqiang Huang, Xian-Sheng Hua, and Min-Jian Zhao. "Improving 3d object detection with channel-wise transformer". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 2743–2752.

[182] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. "Megatron-lm: Training multi-billion parameter language models using model parallelism". In: *arXiv preprint arXiv:1909.08053* (2019).

[183] Chonghao Sima et al. "DriveLM: Driving with Graph Visual Question Answering". In: *First Vision and Language for Autonomous Driving and Robotics Workshop*.

[184] Uriel Singer et al. "Make-A-Video: Text-to-Video Generation without Text-Video Data". In: *The Eleventh International Conference on Learning Representations*. 2022.

[185] Leslie N Smith. "Cyclical learning rates for training neural networks". In: *2017 IEEE winter conference on applications of computer vision (WACV)*. IEEE. 2017, pp. 464–472.

[186] Ibrahim Sobh, Loay Amin, Sherif Abdelkarim, Khaled Elmadawy, Mahmoud Saeed, Omar Abdeltawab, Mostafa Gamal, and Ahmad El Sallab. "End-to-end multi-modal sensors fusion system for urban automated driving". In: *32nd Conference on Neural Information Processing Systems* (2018).

[187] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. "Deep unsupervised learning using nonequilibrium thermodynamics". In: *International conference on machine learning*. PMLR. 2015, pp. 2256–2265.

[188] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. "Roformer: Enhanced transformer with rotary position embedding". In: *Neurocomputing* 568 (2024), p. 127063.

[189] Jiachen Sun, Haizhong Zheng, Qingzhao Zhang, Atul Prakash, Z Morley Mao, and Chaowei Xiao. "CALICO: Self-Supervised Camera-LiDAR Contrastive Pre-training for BEV Perception". In: *arXiv preprint arXiv:2306.00349* (2023).

[190] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. "LoFTR: Detector-free local feature matching with transformers". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 8922–8931.

[191] Pei Sun et al. "Scalability in perception for autonomous driving: Waymo open dataset". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 2446–2454.

[192] Alexander Swerdlow, Runsheng Xu, and Bolei Zhou. "Street-view image generation from a bird's-eye view layout". In: *IEEE Robotics and Automation Letters* (2024).

[193] Yi Tay et al. "Ul2: Unifying language learning paradigms". In: *arXiv preprint arXiv:2205.05131* (2022).

[194] Gemini Team et al. "Gemini: a family of highly capable multimodal models". In: *arXiv preprint arXiv:2312.11805* (2023).

[195] OpenAI Team. *GPT-4o mini: advancing costefficient intelligence*. 2024.

[196] Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. "Visual autoregressive modeling: Scalable image generation via next-scale prediction". In: *Advances in neural information processing systems* 37 (2024), pp. 84839–84865.

[197] Thomas Tian, Boyi Li, Xinshuo Weng, Yuxiao Chen, Edward Schmerling, Yue Wang, Boris Ivanovic, and Marco Pavone. "Tokenize the World into Object-level Knowledge to Address Long-tail Events in Autonomous Driving". In: *Workshop on Language and Robot Learning: Language as an Interface*.

[198] Xiaoyu Tian, Haoxi Ran, Yue Wang, and Hang Zhao. "GeoMAE: Masked Geometric Target Prediction for Self-supervised Point Cloud Pre-Training". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 13570–13580.

[199] Xiaoyu Tian et al. "Drivevlm: The convergence of autonomous driving and large vision-language models". In: *arXiv preprint arXiv:2402.12289* (2024).

[200] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. "Fcos: Fully convolutional one-stage object detection". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 9627–9636.

[201] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. "Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training". In: *Advances in neural information processing systems* 35 (2022), pp. 10078–10093.

[202] Thomas Unterthiner, Sjoerd Van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. "Towards accurate generative models of video: A new metric & challenges". In: *arXiv preprint arXiv:1812.01717* (2018).

[203] Balakrishnan Varadarajan et al. "Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 7814–7821.

[204] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need". In: *Advances in neural information processing systems* 30 (2017).

[205] Ruben Villegas, Mohammad Babaeizadeh, Pieter-Jan Kindermans, Hernan Moraldo, Han Zhang, Mohammad Taghi Saffar, Santiago Castro, Julius Kunze, and Dumitru Erhan. "Phenaki: Variable length video generation from open domain textual description". In: *ICLR*. 2023.

[206] Sourabh Vora, Alex H Lang, Bassam Helou, and Oscar Beijbom. "Pointpainting: Sequential fusion for 3d object detection". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 4604–4612.

[207] Anton Voronov, Denis Kuznedelev, Mikhail Khoroshikh, Valentin Khrulkov, and Dmitry Baranchuk. "Switti: Designing scale-wise transformers for text-to-image synthesis". In: *arXiv preprint arXiv:2412.01819* (2024).

[208] Chien-Yao Wang, Hong-Yuan Mark Liao, Yueh-Hua Wu, Ping-Yang Chen, Jun-Wei Hsieh, and I-Hau Yeh. "CSPNet: A new backbone that can enhance learning capability of CNN". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*. 2020, pp. 390–391.

[209] Chunwei Wang, Chao Ma, Ming Zhu, and Xiaokang Yang. "Pointaugmenting: Cross-modal augmentation for 3d object detection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 11794–11803.

[210] Limin Wang, Bingkun Huang, Zhiyu Zhao, Zhan Tong, Yinan He, Yi Wang, Yali Wang, and Yu Qiao. "Videomae v2: Scaling video masked autoencoders with dual masking". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 14549–14560.

[211] Qitai Wang, Lue Fan, Yuqi Wang, Yuntao Chen, and Zhaoxiang Zhang. "Freevs: Generative view synthesis on free driving trajectory". In: *arXiv preprint arXiv:2410.18079* (2024).

[212] Shihao Wang, Yingfei Liu, Tiancai Wang, Ying Li, and Xiangyu Zhang. "Exploring object-centric temporal modeling for efficient multi-view 3d object detection". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2023, pp. 3621–3631.

[213] Shihao Wang, Zhiding Yu, Xiaohui Jiang, Shiyi Lan, Min Shi, Nadine Chang, Jan Kautz, Ying Li, and Jose M Alvarez. "OmniDrive: A Holistic LLM-Agent Framework for Autonomous Driving with 3D Perception, Reasoning and Planning". In: *arXiv preprint arXiv:2405.01533* (2024).

[214] Shihao Wang, Zhiding Yu, Xiaohui Jiang, Shiyi Lan, Min Shi, Nadine Chang, Jan Kautz, Ying Li, and Jose M Alvarez. "Omnidrive: A holistic vision-language dataset for autonomous driving with counterfactual reasoning". In: *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2025, pp. 22442–22452.

[215] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. "Fcos3d: Fully convolutional one-stage monocular 3d object detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 913–922.

[216] Xiang Wang, Shiwei Zhang, Changxin Gao, Jiayu Wang, Xiaoqiang Zhou, Yingya Zhang, Luxin Yan, and Nong Sang. *UniAnimate: Taming Unified Video Diffusion Models for Consistent Human Image Animation*. 2024. arXiv: 2406.01188 [cs.CV]. URL: https://arxiv.org/abs/2406.01188.

[217] Xiao Wang, Haoqi Fan, Yuandong Tian, Daisuke Kihara, and Xinlei Chen. "On the importance of asymmetry for siamese representation learning". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 16570–16579.

[218] Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, and Jiwen Lu. "Drivedreamer: Towards real-world-driven world models for autonomous driving". In: *arXiv preprint arXiv:2309.09777* (2023).

[219] Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, Jiagang Zhu, and Jiwen Lu. *DriveDreamer: Towards Real-world-driven World Models for Autonomous Driving*. 2023. arXiv: 2309.09777 [cs.CV]. URL: https://arxiv.org/abs/2309.09777.

[220] Xiaolong Wang and Abhinav Gupta. "Unsupervised learning of visual representations using videos". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 2794–2802.

[221] Yue Wang, Vitor Campagnolo Guizilini, Tianyuan Zhang, Yilun Wang, Hang Zhao, and Justin Solomon. "Detr3d: 3d object detection from multi-view images via 3d-to-2d queries". In: *Conference on Robot Learning*. PMLR. 2022, pp. 180–191.

[222] Yue Wang and Justin M Solomon. "Deep closest point: Learning representations for point cloud registration". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 3523–3532.

[223] Yuqi Wang, Jiawei He, Lue Fan, Hongxin Li, Yuntao Chen, and Zhaoxiang Zhang. *Driving into the Future: Multiview Visual Forecasting and Planning with World Model for Autonomous Driving*. 2023. arXiv: 2311.17918 [cs.CV]. URL: https://arxiv.org/abs/2311.17918.

[224] Yuqi Wang, Jiawei He, Lue Fan, Hongxin Li, Yuntao Chen, and Zhaoxiang Zhang. "Driving into the future: Multiview visual forecasting and planning with world model for autonomous driving". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 14749–14759.

[225] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. "Chain-of-thought prompting elicits reasoning in large language models". In: *Advances in neural information processing systems* 35 (2022), pp. 24824–24837.

[226] Yuqing Wen et al. "Panacea: Panoramic and controllable video generation for autonomous driving". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 6902–6912.

[227] Xinshuo Weng, Boris Ivanovic, Yan Wang, Yue Wang, and Marco Pavone. "PARA-Drive: Parallelized Architecture for Real-time Autonomous Driving". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 15449–15458.

[228] Penghao Wu, Xiaosong Jia, Li Chen, Junchi Yan, Hongyang Li, and Yu Qiao. "Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 6119–6132.

[229] Yanhao Wu, Tong Zhang, Wei Ke, Sabine Süsstrunk, and Mathieu Salzmann. "Spatiotemporal Self-supervised Learning for Point Clouds in the Wild". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 5251–5260.

[230] xAI. *Grok-1.5 vision preview*. 2024.

[231] Enze Xie et al. "SANA: Efficient High-Resolution Image Synthesis with Linear Diffusion Transformers". In: *arXiv preprint arXiv:2410.10629* (2024).

[232] Saining Xie, Jiatao Gu, Demi Guo, Charles R Qi, Leonidas Guibas, and Or Litany. "Pointcontrast: Unsupervised pre-training for 3d point cloud understanding". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*. Springer. 2020, pp. 574–591.

[233] Yichen Xie, Hongge Chen, Gregory P Meyer, Yong Jae Lee, Eric M Wolff, Masayoshi Tomizuka, Wei Zhan, Yuning Chai, and Xin Huang. "Cohere3D: Exploiting Temporal Coherence for Unsupervised Representation Learning of Vision-based Autonomous Driving". In: *arXiv preprint arXiv:2402.15583* (2024).

[234] Yichen Xie, Chensheng Peng, Mazen Abdelfattah, Yihan Hu, Jiezhi Yang, Eric Higgins, Ryan Brigden, Masayoshi Tomizuka, and Wei Zhan. "RAYNOVA: Scale-Temporal Autoregressive World Modeling in Ray Space". In: *arXiv preprint arXiv:2602.20685* (2026).

[235] Yichen Xie, Chenfeng Xu, Chensheng Peng, Shuqi Zhao, Nhat Ho, Alexander T Pham, Mingyu Ding, Masayoshi Tomizuka, and Wei Zhan. "X-Drive: Cross-modality consistent multi-sensor data synthesis for driving scenarios". In: *arXiv preprint arXiv:2411.01123* (2024).

[236] Yichen Xie, Chenfeng Xu, Marie-Julie Rakotosaona, Patrick Rim, Federico Tombari, Kurt Keutzer, Masayoshi Tomizuka, and Wei Zhan. "Sparsefusion: Fusing multi-modal sparse representations for multi-sensor 3d object detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 17591–17602.

[237] Yichen Xie et al. "S4-driver: Scalable self-supervised driving multimodal large language model with spatio-temporal visual representation". In: *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2025, pp. 1622–1632.

[238] Zhenda Xie, Yutong Lin, Zheng Zhang, Yue Cao, Stephen Lin, and Han Hu. "Propagate yourself: Exploring pixel-level consistency for unsupervised visual representation learning". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 16684–16693.

[239] Chenfeng Xu, Tian Li, Chen Tang, Lingfeng Sun, Kurt Keutzer, Masayoshi Tomizuka, Alireza Fathi, and Wei Zhan. "PreTraM: Self-supervised Pre-training via Connecting Trajectory and Map". In: *Computer Vision – ECCV 2022*. Ed. by Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner. Cham: Springer Nature Switzerland, 2022, pp. 34–50. ISBN: 978-3-031-19842-7.

[240] Chenfeng Xu et al. "Image2Point: 3D Point-Cloud Understanding with 2D Image Pretrained Models". In: *European Conference on Computer Vision*. Springer. 2022, pp. 638–656.

[241] Zhenhua Xu, Yujia Zhang, Enze Xie, Zhen Zhao, Yong Guo, Kwan-Yee. K. Wong, Zhenguo Li, and Hengshuang Zhao. *DriveGPT4: Interpretable End-to-end Autonomous Driving via Large Language Model*. 2024. arXiv: 2310.01412 [cs.CV].

[242] Junjie Yan, Yingfei Liu, Jianjian Sun, Fan Jia, Shuailin Li, Tiancai Wang, and Xiangyu Zhang. "Cross Modal Transformer via Coordinates Encoding for 3D Object Dectection". In: *arXiv preprint arXiv:2301.01283* (2023).

[243] Wilson Yan, Yunzhi Zhang, Pieter Abbeel, and Aravind Srinivas. "VideoGPT: Video Generation using VQ-VAE and Transformers". In: *arXiv*. 2021.

[244] Yan Yan, Yuxing Mao, and Bo Li. "Second: Sparsely embedded convolutional detection". In: *Sensors* 18.10 (2018), p. 3337.

[245] Yunzhi Yan, Haotong Lin, Chenxu Zhou, Weijie Wang, Haiyang Sun, Kun Zhan, Xianpeng Lang, Xiaowei Zhou, and Sida Peng. *Street Gaussians: Modeling Dynamic Urban Scenes with Gaussian Splatting*. 2024. arXiv: 2401.01339 [cs.CV]. URL: https://arxiv.org/abs/2401.01339.

[246] Yunzhi Yan, Haotong Lin, Chenxu Zhou, Weijie Wang, Haiyang Sun, Kun Zhan, Xianpeng Lang, Xiaowei Zhou, and Sida Peng. "Street gaussians: Modeling dynamic urban scenes with gaussian splatting". In: *European Conference on Computer Vision*. Springer. 2024, pp. 156–173.

[247] Honghui Yang et al. "Unipad: A universal pre-training paradigm for autonomous driving". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 15238–15250.

[248] Kairui Yang, Enhui Ma, Jibin Peng, Qing Guo, Di Lin, and Kaicheng Yu. "Bevcontrol: Accurately controlling street-view elements with multi-perspective consistency via bev sketch layout". In: *arXiv preprint arXiv:2308.01661* (2023).

[249] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. "Depth anything: Unleashing the power of large-scale unlabeled data". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 10371–10381.

[250] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. "Depth Anything V2". In: *arXiv:2406.09414* (2024).

[251] Zeyu Yang, Jiaqi Chen, Zhenwei Miao, Wei Li, Xiatian Zhu, and Li Zhang. "Deepinteraction: 3d object detection via modality interaction". In: *arXiv preprint arXiv:2208.11112* (2022).

[252] Zhuoyi Yang et al. *CogVideoX: Text-to-Video Diffusion Models with An Expert Transformer*. 2025. arXiv: 2408.06072 [cs.CV]. URL: https://arxiv.org/abs/2408.06072.

[253] Junbo Yin, Dingfu Zhou, Liangjun Zhang, Jin Fang, Cheng-Zhong Xu, Jianbing Shen, and Wenguan Wang. "Proposalcontrast: Unsupervised pre-training for lidar-based 3d object detection". In: *European Conference on Computer Vision*. Springer. 2022, pp. 17–33.

[254] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. "Center-based 3d object detection and tracking". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 11784–11793.

[255] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. "Deep layer aggregation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 2403–2412.

[256] Jiahui Yu et al. "Vector-quantized image modeling with improved vqgan". In: *arXiv preprint arXiv:2110.04627* (2021).

[257] Lijun Yu et al. "MAGVIT: Masked Generative Video Transformer". In: *CVPR*. 2023.

[258] Sihyun Yu, Jihoon Tack, Sangwoo Mo, Hyunsu Kim, Junho Kim, Jung-Woo Ha, and Jinwoo Shin. "Generating Videos with Dynamics-aware Implicit Generative Adversarial Networks". In: *International Conference on Learning Representations*. 2022.

[259] Jiang-Tian Zhai, Ze Feng, Jihao Du, Yongqiang Mao, Jiang-Jiang Liu, Zichang Tan, Yifu Zhang, Xiaoqing Ye, and Jingdong Wang. "Rethinking the Open-Loop Evaluation of End-to-End Autonomous Driving in nuScenes". In: *arXiv preprint arXiv:2305.10430* (2023).

[260] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. "Sigmoid loss for language image pre-training". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 11975–11986.

[261] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. "Adding conditional control to text-to-image diffusion models". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 3836–3847.

[262] Renrui Zhang, Ziyu Guo, Peng Gao, Rongyao Fang, Bin Zhao, Dong Wang, Yu Qiao, and Hongsheng Li. "Point-m2ae: multi-scale masked autoencoders for hierarchical point cloud pre-training". In: *Advances in neural information processing systems* 35 (2022), pp. 27061–27074.

[263] Yumeng Zhang, Shi Gong, Kaixin Xiong, Xiaoqing Ye, Xiao Tan, Fan Wang, Jizhou Huang, Hua Wu, and Haifeng Wang. *BEVWorld: A Multimodal World Model for Autonomous Driving via Unified BEV Latent Space*. 2024. URL: https://openreview.net/forum?id=MFrqTfubEB.

[264] Guangcong Zheng, Xianpan Zhou, Xuewei Li, Zhongang Qi, Ying Shan, and Xi Li. "Layoutdiffusion: Controllable diffusion model for layout-to-image generation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 22490–22499.

[265] Lianmin Zheng et al. *Judging LLM-as-a-judge with MT-Bench and Chatbot Arena*. 2023. arXiv: 2306.05685 [cs.CL].

[266] Wenzhao Zheng, Weiliang Chen, Yuanhui Huang, Borui Zhang, Yueqi Duan, and Jiwen Lu. "Occworld: Learning a 3d occupancy world model for autonomous driving". In: *arXiv preprint arXiv:2311.16038* (2023).

[267] Wenzhao Zheng, Ruiqi Song, Xianda Guo, and Long Chen. "Genad: Generative end-to-end autonomous driving". In: *arXiv preprint arXiv:2402.11502* (2024).

[268] Xiaoyu Zhou, Zhiwei Lin, Xiaojun Shan, Yongtao Wang, Deqing Sun, and Ming-Hsuan Yang. *DrivingGaussian: Composite Gaussian Splatting for Surrounding Dynamic Autonomous Driving Scenes*. 2024. arXiv: 2312.07920 [cs.CV]. URL: https://arxiv.org/abs/2312.07920.

[269] Xiaoyu Zhou, Zhiwei Lin, Xiaojun Shan, Yongtao Wang, Deqing Sun, and Ming-Hsuan Yang. "Drivinggaussian: Composite gaussian splatting for surrounding dynamic autonomous driving scenes". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 21634–21643.

[270] Xin Zhou, Dingkang Liang, Sifan Tu, Xiwu Chen, Yikang Ding, Dingyuan Zhang, Feiyang Tan, Hengshuang Zhao, and Xiang Bai. "Hermes: A unified self-driving world model for simultaneous 3d scene understanding and generation". In: *arXiv preprint arXiv:2501.14729* (2025).

[271] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. "Objects as points". In: *arXiv preprint arXiv:1904.07850* (2019).

[272] Yin Zhou and Oncel Tuzel. "Voxelnet: End-to-end learning for point cloud based 3d object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 4490–4499.

[273] Zixiang Zhou, Xiangchen Zhao, Yu Wang, Panqu Wang, and Hassan Foroosh. "Centerformer: Center-based transformer for 3d object detection". In: *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXVIII*. Springer. 2022, pp. 496–513.

[274] Benjin Zhu, Zhengkai Jiang, Xiangxin Zhou, Zeming Li, and Gang Yu. "Class-balanced grouping and sampling for point cloud 3d object detection". In: *arXiv preprint arXiv:1908.09492* (2019).

[275] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. "Deformable detr: Deformable transformers for end-to-end object detection". In: *arXiv preprint arXiv:2010.04159* (2020).

[276] Ye Zhu, Yu Wu, Kyle Olszewski, Jian Ren, Sergey Tulyakov, and Yan Yan. "Discrete Contrastive Diffusion for Cross-Modal Music and Image Generation". In: *The Eleventh International Conference on Learning Representations*. 2022.

[277] Sicheng Zuo, Wenzhao Zheng, Yuanhui Huang, Jie Zhou, and Jiwen Lu. "Gaussianworld: Gaussian world model for streaming 3d occupancy prediction". In: *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2025, pp. 6772–6781.

[278] Vlas Zyrianov, Henry Che, Zhijian Liu, and Shenlong Wang. "LidarDM: Generative LiDAR Simulation in a Generated World". In: *arXiv preprint arXiv:2404.02903* (2024).

[279] Vlas Zyrianov, Xiyue Zhu, and Shenlong Wang. "Learning to generate realistic lidar point clouds". In: *European Conference on Computer Vision*. Springer. 2022, pp. 17–35.

Appendix A

Appendix for Chapter 2 "SparseFusion: Fusing Multi-Modal Sparse Representations for Multi-Sensor 3D Object Detection"

In Sec. A.1, we provide additional experimental results of SparseFusion and complementary details of the experiments presented in the main paper. Then, in Sec. A.2, we elaborate on the details of the architecture of SparseFusion.

A.1 Additional Experiments

A.1.1 Category-wise Results

In Tab. A.1, we report the performance of SparseFusion and our LiDAR-only baseline [5] for each object category in nuScenes *validation set*. SparseFusion achieves significant performance improvement for all of the object categories. In particular, the introduction of camera inputs helps to distinguish objects with similar shapes like motorcycles and bicycles.

Table A.1: Category-wise performance on nuScenes *validation set* including the overall NDS, mAP, and AP for each category.

Methods	Modality	NDS	mAP	car	truck	bus	trailer	const. vehicle	pedestrian	motorcycle	bicycle	traffic cone	barrier
TransFusion-L [5]	L	70.2	65.1	86.5	59.6	74.4	42.2	25.4	86.6	72.1	56.0	74.1	74.1
SparseFusion	L+C	**72.8**	**70.4**	**88.5**	**64.4**	**77.1**	**44.3**	**30.3**	**89.8**	**81.5**	**71.0**	**80.6**	**76.6**

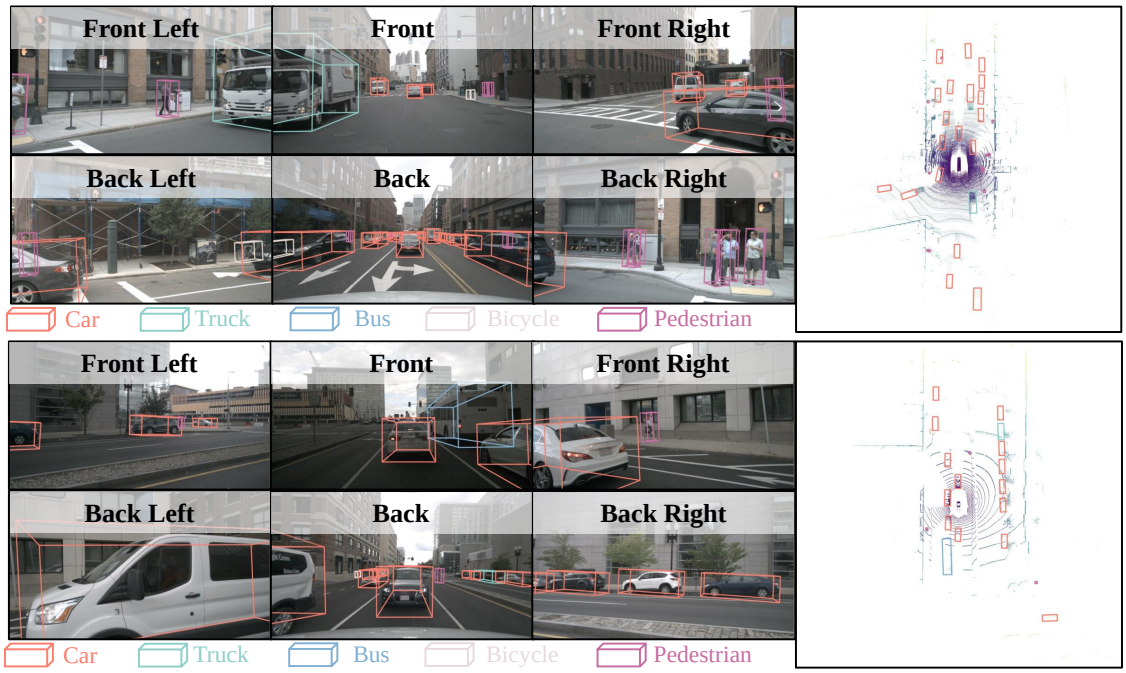


Figure A.1: Qualitative results of SparseFusion on nuScenes *validation set*.

A.1.2 Qualitative Results

We provide additional qualitative results in Fig. A.1, where SparseFusion effectively detects most objects in the scene with the correct classification.

A.1.3 Experiment Details

Cross-Modality Sparse Representation Interaction We provide more details about Fig. 6 in the main paper. The orange boxes refer to the high-confidence objects detected by the fusion branch. The blue and green dots denote all instances from the LiDAR and camera branches separately even if they only have very low confidence. The blue/green lines separately connect the orange boxes and blue/green dots. We only visualize the distribution of attention for high-confidence objects detected in the fusion branches (orange). The magnitude of relationships (*i.e.*, the attention value) is represented by the darkness and thickness of the lines. More examples are visualized in Fig. A.2.

Optimal Transport for Sparse Fusion (main paper Tab. 3a) We explain some details of the optimal transport strategy for sparse fusion in our ablation study. We model the distribution of LiDAR candidates as follows.

$$\mathbf{p}_L(\mathbf{q}_{L,i}) = \frac{s_{L,i}}{\sum_{i=1}^{N_L} s_{L,i}}, i = 1, 2, \ldots, N_L \tag{A.1}$$

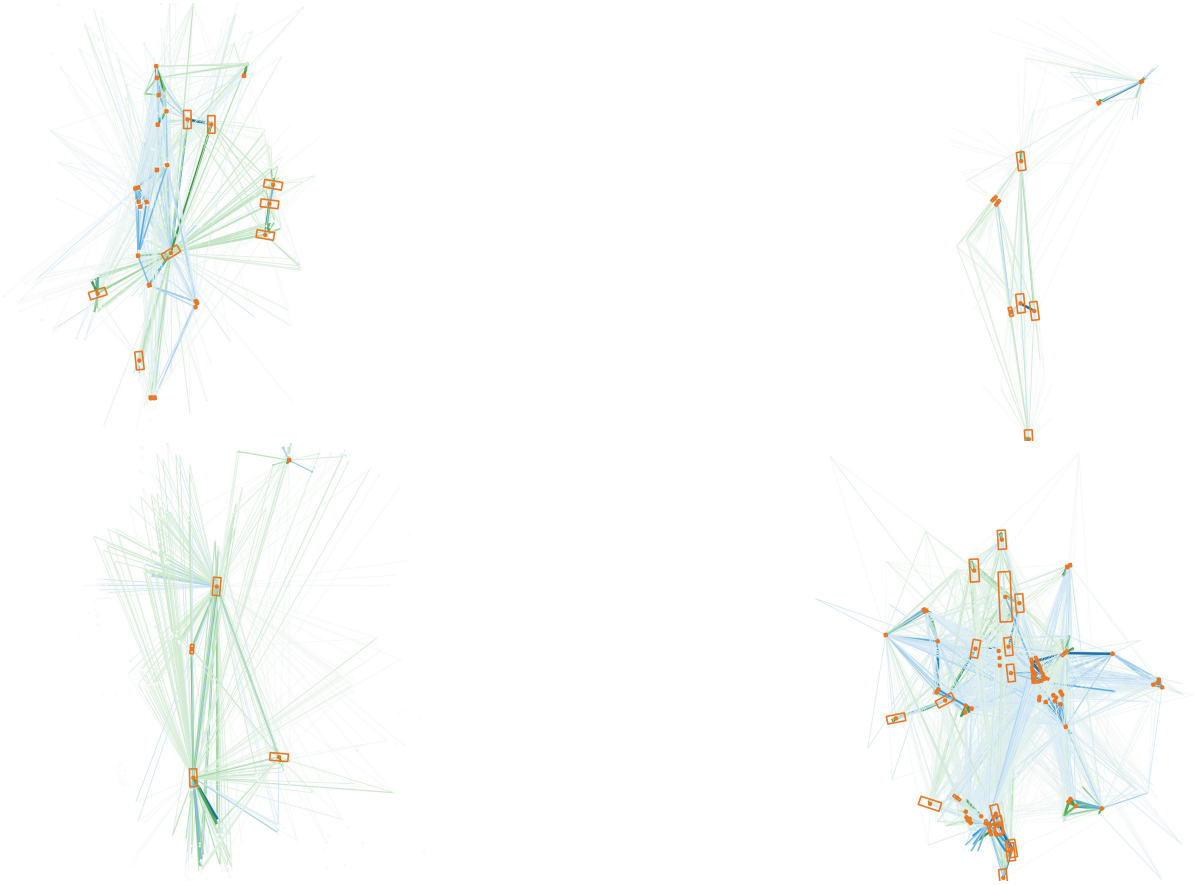


Figure A.2: Instance-level feature interaction in the fusion stage. Orange boxes are objects detected after the fusion stage with high confidence in the BEV space. Blue and green dots denote all instances from the LiDAR and camera branches separately. Orange boxes are connected with blue/green dots with blue/green lines. The strength of attention is represented by the darkness and thickness of the lines.

where $s_{L,i}$ is the classification confidence (highest category) of the i-th instance for the LiDAR detector. Similarly, the distribution of camera candidates is modeled as follows.

$$\mathbf{p}_C(\mathbf{q}_{C,j}) = \frac{s_{C,j}}{\sum_{i=1}^{N_C} s_{C,j}}, i = 1, 2, \ldots, N_C \tag{A.2}$$

where $s_{C,j}$ is the classification confidence (highest category) of the j-th instance for the camera detector (after view transformation). We construct a cost matrix $\mathbf{C} = [c_{ij}], i = 1, 2, \ldots, N_L, j = 1, 2, \ldots, N_C$, where c_{ij} is the euclidean distance between the centers of the i-th LiDAR instance and j-th camera instance on the BEV plane. We solve an optimal transport between $\mathbf{p}_L(\mathbf{q}_{L,i})$ and $p_C(\mathbf{q}_{C,i})$ using the IPOT algorithm [222] which outputs an

optimal transport plan $\mathbf{T}^*$, where

$$\mathbf{T}^* = arg \min_{\mathbf{T} \in \mathbf{R}_+^{N_L \times N_C}} < \mathbf{C}, \mathbf{T} > \tag{A.3}$$

$$s.t. \quad \mathbf{T}\mathbf{1}_{N_C} = \mathbf{p}_L, \mathbf{T}^T\mathbf{1}_{N_L} = \mathbf{p}_C \tag{A.4}$$

We normalize $\mathbf{T}$ for each row as $\hat{\mathbf{T}}_{ij} = \mathbf{T}_{ij} / \sum_{j=1}^{N_C} \mathbf{T}_{ij}$. Then, we concatenate LiDAR candidates $\mathbf{Q}_L$ with the weighted camera candidates $\hat{\mathbf{T}}\mathbf{Q}_C$ (matrix product) in a channel-wise manner. The output features are fed into a feed-forward network to get the N_L fused instance features, then the prediction head can get the object categories and bounding boxes based on the instance features.

A.2 Architecture Details

In this section, we explain the detailed structure of each module in SparseFusion. In addition, we also illustrate the query initialization process for both LiDAR and camera detectors.

A.2.1 Network Architecture

LiDAR Detector We follow TransFusion-L [5] to adopt a transformer-based LiDAR detector. The initial LiDAR queries $\mathbf{Q}_L^0$ (Sec. A.2.3) are passed through a self-attention module, then cross-attention is conducted with the BEV features from the LiDAR backbone. The output queries are fed into a feed-forward network to get the LiDAR candidates $\mathbf{Q}_L$. In both the self-attention and cross-attention modules, we add a positional encoding to all of the queries, keys, and values. Instead of the fixed sine positional embedding [204], we apply the learned embeddings by inputting the 2D XY locations of the queries, keys, and values on the BEV plane to an MLP encoder. A LiDAR view prediction head (Sec. A.2.2) is attached to the LiDAR candidates $\mathbf{Q}_L$ to get the object category as well as the 3D bounding box in LiDAR coordinates.

Camera Detector We extend Deformable-DETR [275] to the 3D object detection task. The initial camera queries $\mathbf{Q}_C^0$ (Sec. A.2.3) go through a self-attention module, then deformable attention is conducted with the image features, where we aggregate multi-scale image features from FPN [119] through deformable attention. In deformable attention, each query only interacts with its corresponding single-view image features. The output queries are fed into a feed-forward network to get the perspective view camera candidates $\mathbf{Q}_C^P$. As we do with the LiDAR detector, we add positional embeddings to all of the queries, keys, and values, which indicate their 2D locations on the image of the corresponding view. A perspective view prediction head (Sec. A.2.2) is attached to the perspective view camera candidates $\mathbf{Q}_C^P$ to get the object category as well as the 3D bounding box in camera coordinates.

Algorithm 2 Geometric Transfer

Require: Multi-scale image feature map of view v: $\mathbf{F}_{C,v} = [\mathbf{F}^l_{C,v}]^L_{l=0}$, sparse depth map of view v: $\mathbf{D}_v$

Ensure: Multi-scale depth-aware image feature map of view v: $\hat{\mathbf{F}}_{C,v} = [\hat{\mathbf{F}}^l_{C,v}]^L_{l=0}$

$\mathbf{F}_D \leftarrow \text{Stem}(\mathbf{D}_v)$
$\hat{\mathbf{F}}_{C,v} \leftarrow [\,]$
for $l = 1, 2, \ldots, L$ **do**
  $\mathbf{F}_D \leftarrow \text{Residual-Block}_l(\mathbf{F}_D)$
  $\mathbf{F}_D \leftarrow \text{Concatenate}(\mathbf{F}_D, \mathbf{F}^l_{C,v})$ {channel-wise concatenation}
  $\mathbf{F}_D \leftarrow Conv^l_{3\times3}(\mathbf{F}_D)$
  Append $\mathbf{F}_D$ to $\hat{\mathbf{F}}_{C,v}$ as $\hat{\mathbf{F}}^l_{C,v}$
end for
return $\hat{\mathbf{F}}_{C,v}$

View Transformation Our view transformation module consists of two parts: feature projection and multi-view aggregation. The feature projection is already described in Eq. 1 of the main paper, which encodes the camera parameters and projected boxes with two MLPs and combines them with the original instance features. The multi-view aggregation is based on a self-attention module. The output instance features belonging to all the different views are put together as $\mathbf{Q}^L_C = \{\mathbf{q}^L_{C,i}\}^{N_C}_{i=1}$. They are fed into a self-attention module and feed-forward layer. For positional embeddings added to each instance feature, we take into account both the predicted box center on the image from the camera detector and the box center on the BEV plane after bounding box coordinate transformation. The 4-dimensional inputs are passed through an MLP to get the positional embedding for each instance feature. The updated queries serve as the camera candidates $\mathbf{Q}_C = \{\mathbf{q}_{C,i}\}^{N_C}_{i=1}$. We also attach a LiDAR view prediction head to the candidates to predict the object category and 3D bounding boxes in the LiDAR coordinates.

Fusion Branch We process the LiDAR candidates $\mathbf{Q}_L = \{\mathbf{q}_{L,i}\}^{N_L}_{i=1}$ and camera candidates $\mathbf{Q}_C = \{\mathbf{q}_{C,i}\}^{N_C}_{i=1}$ with two separate modules $f_L(\cdot), f_C(\cdot)$, each consisting of a fully-connected layer and layer normalization. Then, we concatenate the candidates as $\mathbf{Q}_{LC} = \{\mathbf{q}_{LC,i}\}^{N_L+N_C}_{i=1}$. Afterward, $\mathbf{Q}_{LC}$ is fed into a self-attention module and a feed-forward network to get the final fused instance features $\mathbf{Q}_F$. In the self-attention module, we also add a learned positional embedding to the instance features by encoding the XY box centers on the BEV with an MLP. Finally, we attach a LiDAR view prediction head to $\mathbf{Q}_F$ to predict the object category and 3D LiDAR view bounding boxes as the final results.

Geometric Transfer We project the LiDAR point clouds to multi-view images with camera parameters to get the sparse depth maps (200×112 for nuScenes) for each view. We combine the original multi-level image features from FPN [119] with the sparse depth map to

obtain the multi-level depth-aware image features as shown in Alg. 2, where: L is the scale level number ($L = 4$ in our experiments); Stem$(\cdot)$ is a stem block composed of a 3×3 convolution, batch normalization, and a ReLU activation; Residual-Block$(\cdot)$ is the basic residual block in ResNet-18 [68] with stride 2 for downsampling. Since we have multi-view images describing the surrounding scene, we run Alg. 2 separately for each view with the shared network parameters.

Semantic Transfer Given the dense BEV features $\mathbf{F}_L \in \mathbb{R}^{H \times W \times C}$, only a few positions are indeed covered by the LiDAR point clouds. For a position $(x_j, y_j), x_j \in \{1, 2, \ldots, W\}, y_j \in \{1, 2, \ldots, H\}$ on the BEV feature map occupied by point clouds, we denote the median height of the points in this pillar (x_j, y_j) as z_j. We project all these $\{(x_j, y_j, z_j)\}$ from LiDAR coordinates to the multi-view images. We fetch these image features at these positions (max-pooling to aggregate multi-scale image features), and we combine them with the original corresponding BEV features through element-wise addition. The added features serve as the queries to interact with the multi-scale image features through a deformable-attention module and a feed-forward network. We add the positional embeddings, which are the 2D locations on the images, to the queries, keys, and values. This process is run separately for images of each view. If (x_j, y_j, z_j) can be projected to multiple views, we perform max-pooling to aggregate the updated queries from multiple views. Each updated query replaces the original BEV features $\mathbf{F}_L$ at (x_j, y_j) to obtain the semantic-aware BEV features $\hat{\mathbf{F}}_L$, which will be used for the query initialization of the LiDAR detector (Sec. A.2.3).

A.2.2 Prediction Head

We use two different prediction heads for 3D objects in the perspective view and the LiDAR view.

Perspective View Head The perspective view prediction head is designed for the camera detector to detect objects in the camera coordinates. The head includes six independent MLPs as follows:

1. It predicts the category of each object. The output dimension is the number of object categories, denoting the confidence of each category.

2. For the image of each view, it regresses the offset of the projected center of each object in the image from the reference points indicated by the positional embedding. The output dimension is two, denoting the XY coordinate separately.

3. For the image of each view, it estimates the depths of each object. The output dimension is one.

4. It regresses the logarithms of the XYZ scale of the 3D bounding box. The output dimension is three.

5. It predicts the angle of each object around the vertical axis (Y-axis in the camera coordinate). The output dimension is two, denoting the sin and cos of this angle.

6. It predicts the velocity in the horizontal plane (XZ-plane in the camera coordinate space). The output dimension is two,denoting the velocities along the X-axis and the Z-axis.

LiDAR View Head The LiDAR view prediction head is designed to detect objects in the perspective view. The same head is used for the LiDAR detector, view transformation, and the fusion branch with different network weights. The head includes six independent MLPs as follows:

1. It predicts the category of each object. The output dimension is the number of object categories, denoting the confidence of each category.

2. It regresses the offset of the center of each object on the BEV plane from the reference points indicated by the positional embedding. The output dimension is two, denoting the XY coordinate separately.

3. It regresses the height of each object center. The output dimension is one.

4. It regresses the logarithms of the XYZ scale of the 3D bounding box. The output dimension is three.

5. It predicts the angle of each object around the vertical axis (Z-axis in the LiDAR coordinate space). The output dimension is two, denoting the sin and cos of this angle.

6. It predicts the velocity in the horizontal plane (XY-plane in the LiDAR coordinate). The output dimension is two, denoting the velocities along the X-axis and the Y-axis.

A.2.3 Query Initialization

We follow CenterFormer [273] and TransFusion [5] to initialize our queries using a heatmap, which helps to accelerate the convergence and reduce the number of queries.

Initialization for LiDAR Detector We splatter the bounding box centers on the BEV onto a category-aware heatmap $\mathbf{Y} \in [0,1]^{H\times W\times K}$ [102, 271], where K is the category number, with a Gaussian kernel $\mathbf{Y}_{x,y,k_i} = \exp\left[\frac{(x-c^L_{x,i})^2+(y-c^L_{y,i})^2}{2\sigma_i^2}\right]$, where k_i is the category of the i-th object, $(c_{x,i}, c_{y,i})$ is its center on the BEV, and σ_i is a standard deviation related to the object scale as done in [5]. The heatmap is calculated for each object separately, and we combine the multiple-object heatmaps by using the maximal value at each location. A dense head composed of 3×3 convolutions are attached to the BEV features $\hat{\mathbf{F}}_L$, which is augmented

by the semantic transfer. Positions $\mathbf{p}_{L,i}^0 \in \mathbb{R}^2, i = 1, \ldots, N_L$ on the BEV with the highest confidence scores in $\max_{k_i} \hat{\mathbf{Y}}_{x_i,y_i,k_i}$ are selected as the reference points on BEV plane, along with their categories $\{k_i\}_{i=0}^{N_L}$. The local BEV features at these positions from $\mathbf{F}_L$ are fetched. We add the local feature from $\mathbf{F}_L$ and a learnable category embedding $\{\mathbf{e}_{k_i}^L\}_{i=0}^{N_L}$ to get the initial LiDAR query features $\mathbf{Q}_L^0 = \{\mathbf{q}_{L,i}^0\}_{i=1}^{N_L}$.

Initialization for Camera Detector For the camera modality, 3D box centers are projected into the multi-view images. We follow FCOS [200] to divide the objects of different sizes after projection into certain levels of multi-scale image features. We set the size thresholds to $0, 48, 96, 192, +\infty$. For each bounding box projected on the image plane, if its $\max(length, width)$ falls between the i-th and $i+1$-th threshold, the object is assigned to the i-th scale level. As we do in the LiDAR modality, corresponding projected centers of each feature level are splattered onto a heatmap. We also get the reference points $\mathbf{p}_C^0 = \{\mathbf{p}_{C,i}^0\}_{i=1}^{N_C}$ with top confidence scores from the multi-view image features, as well as the corresponding categories $\{k_i\}_{i=0}^{N_C}$. The corresponding features from the depth-aware image features $\hat{\mathbf{F}}_C$ are added with the learnable category embedding $\{\mathbf{e}_{k_i}^C\}_{i=0}^{N_C}$ to get the initial camera query features $\mathbf{Q}_C^0 = \{\mathbf{q}_{L,i}^0\}_{i=1}^{N_L}$. It is worth mentioning that 3D objects are projected to multi-scale multi-view images, so initial queries come from the image features from different views and different scales.

Appendix B

Appendix for Chapter 4 "S4-Driver: Scalable Self-Supervised Driving Multimodal Large Language Model with Spatio-Temporal Visual Representation"

The supplementary material is organized as follows. Sec. B.1 provides implementation details of our proposed methods including the concrete input prompt and target output. Sec. B.2 provides the model performance for each ego-vehicle behavior separately. Sec. B.3 gives additional visualization of S4-Driver motion planning in diverse scenarios. Finally, Sec. B.4 conducts extra ablation studies on WOMD-Planning-ADE benchmark to justify the design of S4-Driver.

B.1 Implementation Details

WOMD-Planning-ADE benchmark. This benchmark contains 487k scenarios for model training and 44k for validation, which are divided from 103k sequences of $20s$ length. Each scenario contains $1s$ history and $8s$ future. We only consider the future $5s$ for the open-loop evaluation of motion planning since motion planning in real task is conducted in an iterative manner. At each time stamp, the dataset provides multi-view images captured by 8 cameras (namely *front left, front, front right, side left, side right, rear left, rear, rear right*). The dataset is also equipped with fine-grained labels for all the other agents and the roadgraph although they are not used in our self-supervised motion planning algorithm.

Ego-vehicle coordinate system. We conduct the motion planning task in the ego-vehicle coordinate system. At current timestamp, the ego-vehicle position is defined as the origin. The $x-$axis is oriented towards the current heading angle of the ego-vehicle. The $y-$axis is oriented towards the left of the ego-vehicle. The $z-$axis is oriented upwards. Although it is

a 3D coordinate system, we only consider the $xy-$plane for the description of the positions, velocities, and accelerations of the vehicles in the motion planning task.

Architecture and hyperparameters. Unless otherwise specified, we build up the S4-Driver framework based on multimodal large-language model PaLI3-5B [28]. The 3D scene representation (Sec. 4.4.3) in default covers the range of $(-30m, 80m) \times (-30m, 30m) \times (-2m, 8m)$ in the ego-vehicle coordinate along x, y, z axes separately. The position embedding in Eq. 4.4 and Eq. 4.8 are implemented with a two-layer MLP attached to the Fourier embedding of (x, y, z). To generate the gate values, we reduce the dimension of visual features from $C = 1536$ to $C' = 96$ in Eq. 4.6. The volume resolution is $1m \times 1m \times 2m$. For each scene, we select $M = 6000$ sparse volumes based on the gate values. For attention bias (Sec. 4.4.3), there are independent intra-modality biases for visual tokens and text tokens separately. We leave the details of bias function $b(\cdot)$ in the extra ablation study (Sec. B.4). For each attention head at each self-attention layer, we also include learnable scalar inter-modality biases for the attention values between text tokens and visual tokens. In Sec. 4.4.4, for temporal fusion, apart from the current frame, we also take camera images from $T = 1$ historical frame at the timestamp $-0.5s$ as model input. For multi-decoding aggregation (Sec. 4.4.5), the model outputs $K = 16$ trajectories in parallel through Top-p sampling (nucleus sampling) [73] ($p = 0.9$), which are aggregated as one unique planning result. For both datasets, the input images are resized to 448×448 as model inputs.

Model initialization. To ensure the alignment within the pretrained MLLM, we apply the following special rules to relieve the disturbance of injected modules to the pretrained weights in the early finetuning stage. The last FC-layer of MLP for the position embeddings in Eq. 4.4 and Eq. 4.8 are zero initialized. The learnable vacant feature $\mathbf{f}_{vac}$ for sparse volume representation is also initialized as zeros (Eq. 4.9). We also set the initial bias for each bin in the bias function $b(\cdot)$ (Eq. 4.10) as zeros (details in Sec. B.4). Finally, the temporal fusion (Sec. 4.4.4), we maintain the channel-wise semantics of visual features by initializing the weight matrix of the FC layer as $\mathbf{W} = \left[I \in \mathbb{R}^{C\times C}; \mathbf{0}^{C\times(T-1)\cdot C}\right]$ and the bias as zero. Based on the above rules, the inserted modules would not significantly change the channel-wise semantics of the visual features from original pretrained perspective view features, which allows for a stable and efficient finetuning process.

Input prompt and target output. The input prompt is composed of the high-level behavior command and ego-vehicle historical states. We represent all historical states in the language space. The position, velocity, and acceleration at each time stamp are represented with two floating numbers (two decimal places) separately for the $x-$component and $y-$component in the ego-vehicle coordinate. The target output is composed of meta-decision (Sec. 4.4.2 only in S4-Driver) and future waypoints. The future waypoints are also represented in the language space with two floating numbers for each time stamp. The historical and future states are sampled at a frequency of $5Hz$ on WOMD-Planning-ADE

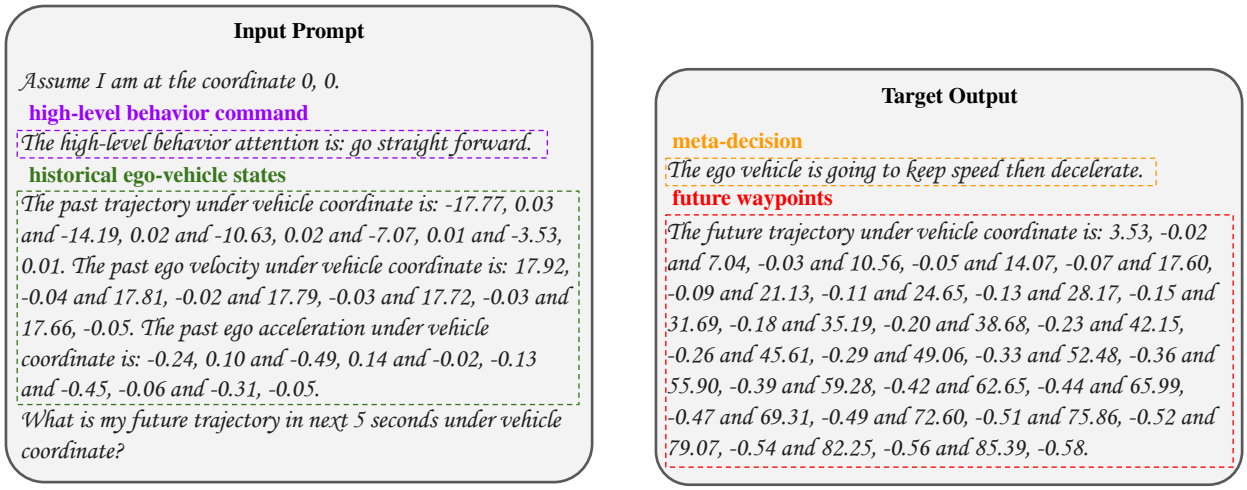


Figure B.1: Example input prompt and target output on WOMD-Planning-ADE.

where we consider the historical states of past $1s$ and future waypoints of future $5s$. On nuScenes dataset, we consider the historical states of past $1s$ and future waypoints of future $3s$ at a frequency of $2Hz$. An example of the input and output on WOMD-Planning-ADE is visualized in Fig. B.1.

Heuristics for high-level behaviors. We consider the seven behaviors for the ego-vehicle on WOMD-Planning-ADE benchmark, which are determined based on the ego-vehicle ground-truth future trajectories following heuristic rules.

1. **Stop**: The ego-vehicle movement is $< 5m$ and the maximal speed is $< 2m/s$.
2. **Do left turn**: The ego-vehicle does not stop. The final heading angle is $> 30°$. The final position is $\geq -5m$ along $x-$axis.
3. **Do left U-turn**: The ego-vehicle does not stop. The final heading angle is $> 30°$. The final displacement is $< -5m$ along $x-$axis.
4. **Do right turn**: The ego-vehicle does not stop. The final heading angle is $< -30°$.
5. **Go straight left**: The ego-vehicle does not stop. The final heading angle is in the range $[-30°, 30°]$. The final displacement is $> 5m$ along $y-$axis.
6. **Go straight right**: The ego-vehicle does not stop. The final heading angle is in the range $[-30°, 30°]$. The final displacement is $< -5m$ along $y-$axis.
7. **Go straight forward**: The ego-vehicle does not stop. The final heading angle is in the range $[-30°, 30°]$. The final displacement along $y-$axis is in the range $[-5m, 5m]$.

For the calculation of *bADE* metric in the model evaluation, we follow the above rules to determine the behavior based on the ground-truth future trajectories of $8s$ although our planning horizon is only $5s$. For the high-level behavior command in the model inputs, "stop" is excluded to avoid future information leakage. To determine the high-level behavior command input, we start from the ground-truth $8s$ future trajectories. If none of behaviors 2-7 is satisfied, we would prolong the future horizon by $2s$ until at least one of behaviors 2-7 is satisfied. If the end of the collected trajectory sequence is reached, we would just consider that scenario as "go straight forward".

Heuristics for meta-decisions. In Sec. 4.4.2, we design a meta-decision strategy as a preliminary prediction before the waypoints. The meta-decision of ego-vehicle includes four categories determined with following heuristic rules based on the ground-truth future information.

1. **Keep stationary**: The maximal speed is $< 2m/s$ and the final displacement is $< 1.5m$.
2. **Keep speed**: The ego-vehicle does not keep stationary. The average acceleration is in the range of $[-0.5m/s^2, 0.5m/s^2]$.
3. **Accelerate**: The ego-vehicle does not keep stationary. The average acceleration is $> 0.5m/s^2$.
4. **Decelerate**: The ego-vehicle does not keep stationary. The average acceleration is $< -0.5m/s^2$.

The model should firstly predict the meta-decision and then auto-regressively output the future waypoints. On WOMD-Planning-ADE with $5s$ future prediction horizon, we divide the $5s$ into two stages of $2.5s$, where meta-decisions are predicted independently for each stage. On nuScenes dataset with $3s$ prediction horizon, we only consider one-stage meta-decision.

Table B.1: Behavior-wise sliced metrics on WOMD-Planning-ADE benchmark. "*" denotes methods with internal data pretraining.

Methods	**ADE@5s for each behavior**							ADE@5s	bADE@5s
	stop	straight forward	straight left	straight right	left turn	right turn	left U-turn		
Vanilla PaLI	**0.048**	0.960	1.252	1.297	1.566	1.323	1.039	0.798	1.069
S4-Driver (ours)	0.063	0.843	1.077	1.177	1.252	1.158	0.925	0.693	0.928
S4-Driver* (ours)	0.065	**0.806**	**0.957**	**1.074**	**1.124**	**1.027**	**0.756**	**0.655**	**0.830**
Models that take high-quality objects, tracks, and roadgraphs as inputs instead of using raw camera images.									
MotionLM	0.048	0.832	1.293	1.275	1.239	1.172	0.990	0.697	0.978



Figure B.2: Additional qualitative results of motion planning. We show the *front left, front, front right* cameras for each case.

B.2 Behavior-wise Model Performance

In Tab. B.1, we report the *ADE@5s* metric of S4-Driver for each ego-vehicle behavior on WOMD-Planning-ADE benchmark separately. Results show the superiority of S4-Driver especially in complicated scenarios like turnings, where a significant performance gain is witnessed from vanilla PaLI baseline to S4-Driver. Motion planning for these difficult behaviors requires a great understanding of the roadgraph and other agents from raw camera images, which reflects the strong reasoning ability of our proposed spatio-temporal visual representation.

B.3 Additional Qualitative Results

In Fig. B.2, we visualize more planning results on WOMD-Planning-ADE. Examples cover different behaviors, speeds, lighting conditions, and weathers. Results show the robust performance of S4-Driver in all these diverse scenarios.

Table B.2: Ablation studies on camera configurations.

Camera configuration	ADE@5s	bADE@5s
front	0.765	1.036
front, front left/right	0.751	1.012
front, front left/right, side left/right	0.746	**0.981**
all eight surrounding cameras	**0.732**	0.985

B.4 Additional Ablation Studies

In this part, we conduct several additional ablation studies to further justify the design of our S4-Driver including the camera configuration, relative attention bias, multi-decoding aggregation, and motion tokens.

Camera configuration. We apply different configurations of camera sensors in Tab. B.2. Consistent with intuitions, the *front* camera is the most important, which can solely guarantee a reasonable planning performance. Adding other cameras continuously improves the model performance since they provide more and more complete information of the surrounding environment. The three cameras in the back can notably boost the planning model since they provide cues about other agents behind ego-vehicle. This information helps to determine the future ego-vehicle velocities and accelerations.

Relative attention bias. We consider two types of relative position bias function $b(\cdot)$ in Eq. 4.10.

1. **Linear bias:** The bias is linearly related to the relative distance between tokens. For the sparse volume visual tokens, $b(x, y, z) = b_x(\Delta x) + b_y(\Delta y) + b_z(\Delta z)$, we take the $x-$axis as an example,
$$b_x(\Delta x) = \tau_x \cdot |\Delta x|, \tag{B.1}$$
where τ_x is a learnable parameter separately for each attention head at each layer, and Δx is the relative position between sparse volume tokens along $x-$axis. Similarly, for 1D text tokens, the separate bias function is
$$b_p(\Delta p) = \tau_p \cdot |\Delta p| \tag{B.2}$$
where τ_p is a learnable parameter separately for each attention head in each layer, and Δp is the relative position between text tokens.

2. **Bin-wise bias (our choice):** The relative positions $\Delta x, \Delta y, \Delta z$ (visual tokens) and Δp (text tokens) for each axis are divided into 32 bins independently. For each axis,

16 bins cover the range $[-8(m), 8(m)]$ linearly with an interval $1(m)$[1]. The other 16 bins symmetrically cover the range $(-128(m), -8(m))$ and $(8(m), 128(m))$ in log scale, where the relative distances are truncated to at most $128(m)$. In this case, if we take $b_x(\cdot)$ as an example, the relative bias function is written as

$$b_x(\Delta x) = m_x(\text{bin}(\Delta x)) \tag{B.3}$$

where $m_x(\cdot)$ maps each bin to a learnable bias value independently for each attention head at each self-attention layer.

Tab. B.3 justifies the design of bin-wise bias, which brings greatly better performance. We think it is important to distinguish the two directions along each axis (*e.g.* front or back) in the motion planning task. Besides, local feature aggregation is more sensitive to close neighbors at each locality, so the bin-wise bias function has finer grains in close distance compared to the linear bias function.

Table B.3: Ablation studies on attention bias.

Attention bias	**ADE@5s**	**bADE@5s**
no bias	0.750	1.005
linear bias function	0.770	1.082
bin-wise bias function	**0.732**	**0.985**

Multi-decoding aggregation. We dig into the multi-decoding strategy which can bring notable performance gain. The motivation is that the MLLM is prone to assigning high confidence scores to simple future behaviors such as *stop*. To this end, we encourage the model to output multiple future trajectories. Their aggregation serves as the final planning result, which can counteract the model's high confidence in simple behaviors. In Tab. B.4, we find that 1) The aggregation of more decoded trajectories leads to better performance. 2) Nucleus sampling can outperform beam search since it can generate more diverse outputs. 3) Weighted average is inferior to mean average since the model is prone to assigning high confidence to simple degenerated behaviors.

Motion tokens. Several prior works [177, 183] benefit from specialized trajectory tokenization modules, which converts motion trajectories into extra discrete tokens added to the vocabulary of language models. However, in Tab. B.5, the trajectory tokenization strategy similar with MotionLM [177] hurts the performance of our S4-Driver in comparison with naïve floating number waypoints representation (Fig. B.1). We also witness a much slower

[1]The unit (m) is only for visual tokens $(\Delta x, \Delta y, \Delta z)$ throughout this paragraph.

Table B.4: Ablation studies on multi-decoding aggregation.

Sample number	**Sample strategy**	**Aggregation**	**ADE@5s**	**bADE@5s**
1	greedy sampling	-	0.728	0.986
4	beam search	average	0.739	0.997
4	nucleus sampling	average	0.709	0.941
4	nucleus sampling	weighted average	0.747	1.005
16	nucleus sampling	average	**0.693**	**0.928**

Table B.5: Ablation studies on motion trajectory tokenization.

Trajectory representation	**ADE@5s**	**bADE@5s**
motion tokens	0.779	1.061
floating numbers	**0.750**	**1.005**

convergence speed with this extra trajectory tokenization. Since the MLLM is already pretrained on large-scale data, additionally injected trajectory tokens may not align with the pretrained model. In contrast, the floating number representation can align the historical and future states in the language space to exploit the large-scale MLLM pretraining with lower requirement for finetuning.

Appendix C

Appendix for Chapter 5 "X-Drive: Cross-Modality Consistent Multi-Sensor Data Synthesis for Driving Scenarios"

In this appendix, we first give an extra ablation study about the classifier-free guidance in Appendix C.1. Then, we elaborate our implementation details in Appendix C.2. Afterwards, we discuss the limitation of our framework and future works in Appendix C.3. Finally, more qualitative results of multi-modality data generation are demonstrated in Appendix C.4.

C.1 Ablation Study of Classifier-Free Guidance Scales

We employ classifier-free guidance (CFG) during inference for box and text conditions. Illustrated in Fig. C.1, it improves the realism of synthetic data. However, higher CFG scale would hurt the image and point cloud quality by increasing the contrast and sharpness.

C.2 Implementation Details

In this part, we explain details in the implementation of X-Drive including network architecture, training schedule, and evaluation metrics. Our code will be made publicly available upon the acceptance of this paper.

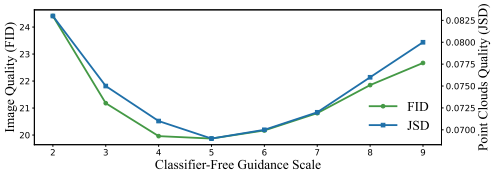


Figure C.1: Effect of CFG scales.

Network architecture. Given the small height dimension of LiDAR range images and limited data amount, we employ smaller sizes for the VAE and latent diffusion model (LDM) for LiDAR range images.

Specifically, the VAE includes three encoder blocks and three decoder blocks with channels $[32, 64, 128]$ separately. There are downsampling modules in the first two encoder blocks. The latent $\mathbf{z}^r$ for range image has a size of $8 \times 256 \times 8$. The range image LDM is composed of three downsampling blocks, one middle block, and three upsampling blocks with attention modules in all the blocks. The channel numbers for three downsampling and upsampling blocks are $[128, 256, 512]$ respectively. The cross-modality epipolar condition modules are inserted between each pair of corresponding blocks in point cloud and multi-view image diffusion model branches. For these seven cross-modality modules, we set the sampling number R and D along range and depth axes as $[24, 24, 12, 6, 12, 24, 24]$ separately with both maximal depth and range as $60m$.

Training schedule. The training of X-Drive on nuScenes dataset [18] includes three stages. Since there are no publicly available pretrained weights for LiDAR diffusion models, we train the VAE and LDM for LiDAR range image in the first two stages separately. Afterwards, we train the entire multi-modality generation framework together. Before training, we initialize our multi-view image diffusion model branch with pretrained Stable-Diffusion V1.5 weights, while all the other modules except the zero-initialization gates are initialized randomly. In all the stages, our model is trained using NVIDIA RTX A6000 GPUs.

In the first stage, VAE for LiDAR range image is trained using batch size 96 and learning rate 4e-4 for 200 epochs. The discriminator takes effect after 1000 iterations. In the second stage, we train the LiDAR LDM from scratch using batch size 96 and learning rate 1e-4 for 2000 epochs. The model includes the text prompt and 3D range-view bounding box condition modules with drop-rate 0.25 for either condition during training. In the last stage, we train the entire framework in an end-to-end manner with module trainable states illustrated in Fig. 5.2 using condition drop-rate 0.25 as well. The entire model is trained for 250 epochs with learning rate 8e-5 and batch size 24 in our main experiments. For ablation studies, we reduce the epoch number to 80 for efficiency.

Evaluation metrics. In our quantitative evaluation, we apply the FID metric from MagicDrive [51], VSC metric from BEVGen [192], MMD and JSD metrics from LiDARGen [279]. For our proposed DAS metric, we run pretrained DepthAnythingV2 model [250] with ViT-B backbone [46] on the synthetic images to estimate the disparity [249]. Then, we project the synthetic point clouds to multi-view images to get sparse disparity. We normalize the projected sparse disparity to the same scale as the estimated disparity. The mean absolute error is fetched as our DAS metric.

C.3 Limitation and Future Works

This paper concentrates on the joint generation of consistent multi-modality data. In this case, due to the limitation of accessible computational resources, we do not include some natural extensions of our X-Drive framework. Firstly, our method is limited to the generation

of single-frame point clouds and multi-view images. In fact, it is intuitive to combine existing temporal attention modules like [226] with our X-Drive. Secondly, we can also incorporate some extra conditions for data synthesis such as HD maps [51] into our framework for more controllable generation. Lastly, due to the distribution of nuScenes training set [18], we cannot synthesize some specific scenarios such as snowy weathers. We will try to extend our framework to deal with above limitations.

Our proposed X-Drive is orthogonal to other efforts in improving single-modality generative model performance such as transformer-based image generation framework [151], high image resolutions [231], or point clouds representation for LiDAR sensor data. We will combine the cross-modality condition module of X-Drive with these advanced techniques to further enhance its performance in multi-modality data generation.

C.4 Qualitative Results

In this section, we demonstrate additional examples of our synthetic multi-modality data in Fig. C.2. X-Drive can generate realistic point clouds and multi-view images with cross-modality consistency based on the input conditions.

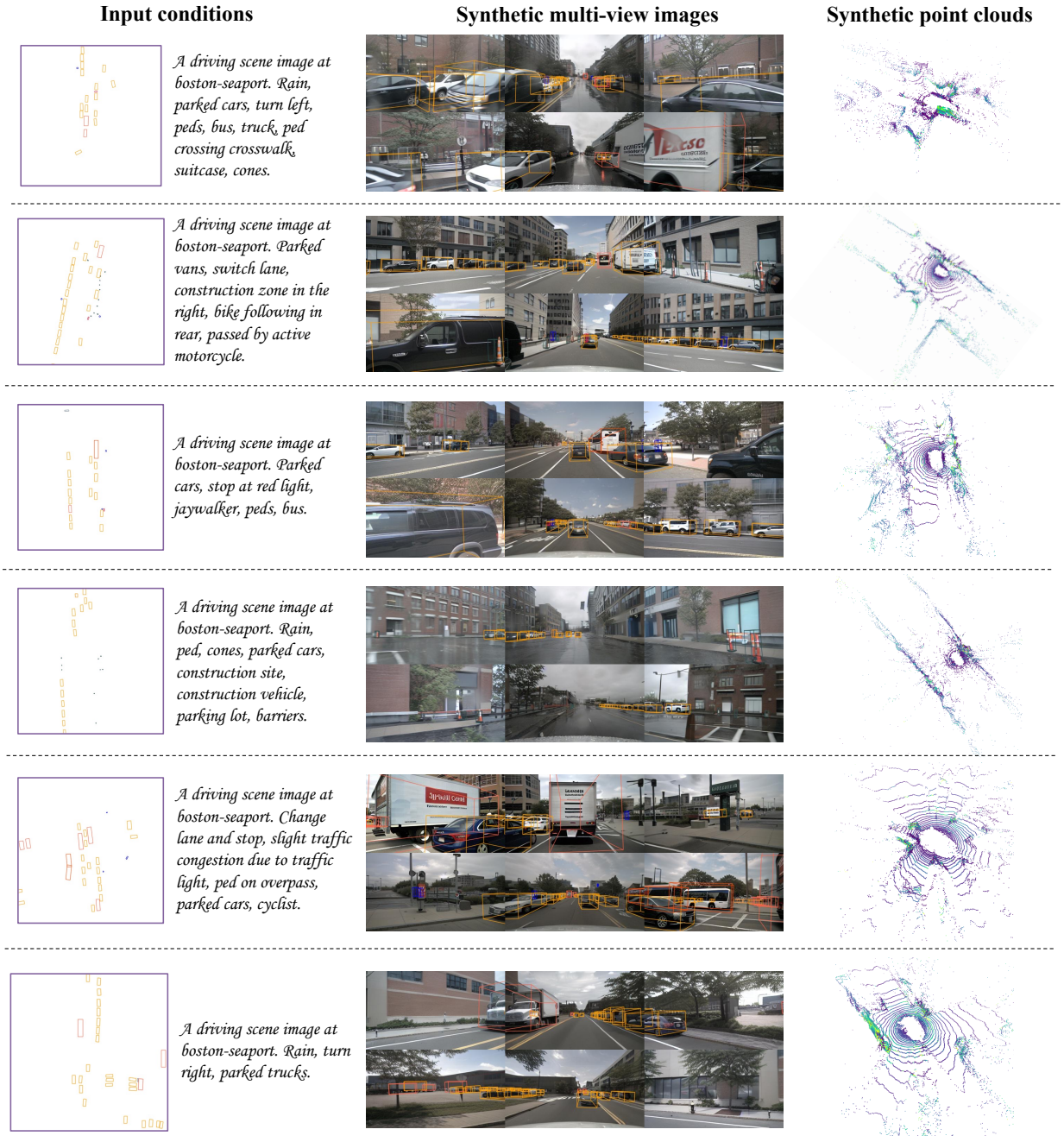


Figure C.2: Additional multi-modality data synthesis results using our proposed X-Drive.

Appendix D

Appendix for Chapter 6 "RAYNOVA: Geometry-Free Auto-Regressive 4D World Modeling with Unified Spatio-Temporal Representation"

In this supplementary material, we present additional implementation details and results that could not be included in the main paper due to space constraints. In Sec. D.1, we provide further details on the model architecture and training procedure. In Sec. D.2, we report additional ablation studies to support our design choices. In Sec. D.3, we include more qualitative results. Finally, we discuss the limitations of our work in Sec. D.4.

D.1 Implementation Details

As mentioned in Sec. 6.4.1, RAYNOVA model is initialized from the pretrained weights of Infinity [61]. Although video VAEs [147, 252] are widely recognized for improving temporal coherence and yielding better FVD scores, we instead directly adopt the multi-scale image tokenizer used in Infinity [61] to better accommodate drastic camera motions and varying frame rates.

RAYNOVA includes two model variants with approximately 130M and 2B parameters. Unless otherwise specified, all experiments use the 2B model. The 130M model is only used in several ablation studies, including Tab. 6.6, Fig. 6.5, and Tab. D.2.

The training of RAYNOVA includes three stages as follows. In the first stage, we first train the model on short multi-view video clips containing 4 frames at a resolution of 192×336. The frame intervals are independently sampled between 0.1s and 1s. We use a batch size of 64 and a learning rate of *5e-5*, and train for 4k iterations. In the second stage, we then train the model on the same short clips but at a higher resolution of 384×672. The batch size is reduced to 32, and the learning rate is decreased to *2.5e-5*. This stage also

Table D.1: Ablation Study on Spatio-Temporal Module.

S.-T. Module	FID↓	FVD↓
decoupled	15.6	140
global	**10.5**	**91**

Table D.2: Ablation Study on Random Errors in Training.

Random Errors	FID↓	FVD↓
✗	19.8	142
✓	**17.2**	**124**

runs for 4k iterations. In the last stage, we adopt the recurrent training scheme described in Alg. 1. The model is trained on long multi-view image sequences with 20 frames at 384×672 resolution. We use a batch size of 32 and a learning rate of *2.5e-6* for 300 iterations. All training is conducted on 32 NVIDIA A100 GPUs using the AdamW optimizer [129].

D.2 Additional Ablation Study

Global Causal Attention. In Sec. 6.3.4, we introduce a global self-attention module within the dual-causal block that jointly attends over all cameras and frames to ensure spatio-temporal consistency. Following prior work [51, 226, 224], we also experiment with an alternative decoupled design that applies two sequential self-attention modules: one for cross-view interactions per frame and one for cross-frame interactions per camera. As shown in Tab. D.1, the global attention module significantly outperforms the decoupled spatio–temporal design. The decoupled approach implicitly assumes strong temporal correlation across frames captured by the same camera, thereby injecting strong ego-motion priors. In contrast, the global attention module incorporates minimal inductive bias and therefore generalizes better.

Random Errors in Training. RAYNOVA inherits the teacher-forcing autoregressive training scheme used in VAR [196] and Infinity [61], which can lead to error propagation due to the mismatch between training and inference. To mitigate this issue, we follow the idea introduced in Infinity [61] and inject random errors into the bit-wise multi-scale tokens X_k across all three training stages by randomly flipping a portion of bits in each token. As shown in Tab. D.2, this simple strategy improves video generation quality by narrowing the gap between training and inference.

D.3 Additional Qualitative Results

Zero-Shot to Novel Camera Configurations. The ray-level relative position embedding enables RAYNOVA to generalize to unseen camera configurations. In Fig. D.1, we generate multi-view videos under the camera setup of the Waymo Open Dataset [191], which is never seen during training. Nevertheless, RAYNOVA is able to synthesize multi-view videos with reasonable visual quality and temporal coherence.

Camera Rotation & Shifting. In Fig. D.2, RAYNOVA is able to simulate camera rotations and translations. This highlights another advantage of our ray-level position embedding: it enables RAYNOVA to effectively function as a feed-forward novel view synthesis model.

Multi-View Video Generation. In Fig. D.3 and Fig. D.4, we present additional examples of multi-view video generation across diverse camera configurations and environments. We also provide several synthetic multi-view videos on our website with object and map input conditions. RAYNOVA demonstrates strong condition fidelity and high visual realism.

D.4 Limitation and Future Work

Our current training data are limited to driving scenarios, which constrains the generalization ability of RAYNOVA in non-driving environments. In future work, we plan to incorporate more diverse datasets (e.g., robotics or UAV) into the training pipeline. In addition, our evaluation primarily focuses on multi-view video generation. We aim to explore broader applications of RAYNOVA, such as closed-loop simulation for end-to-end driving models [88, 237]. Recent advances in visual autoregressive models [57, 207] can also be readily integrated into the RAYNOVA framework to further enhance its capability and efficiency.

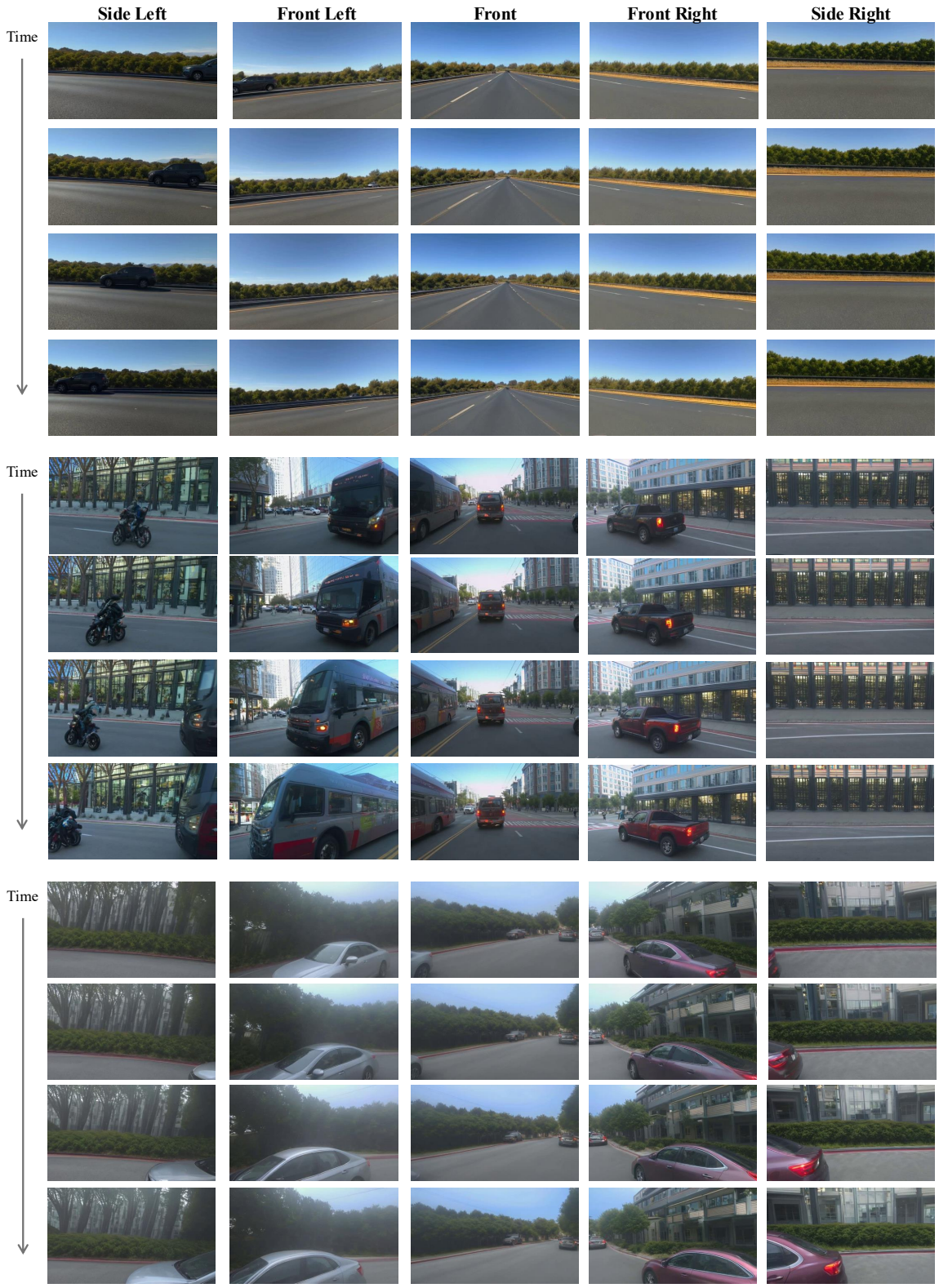
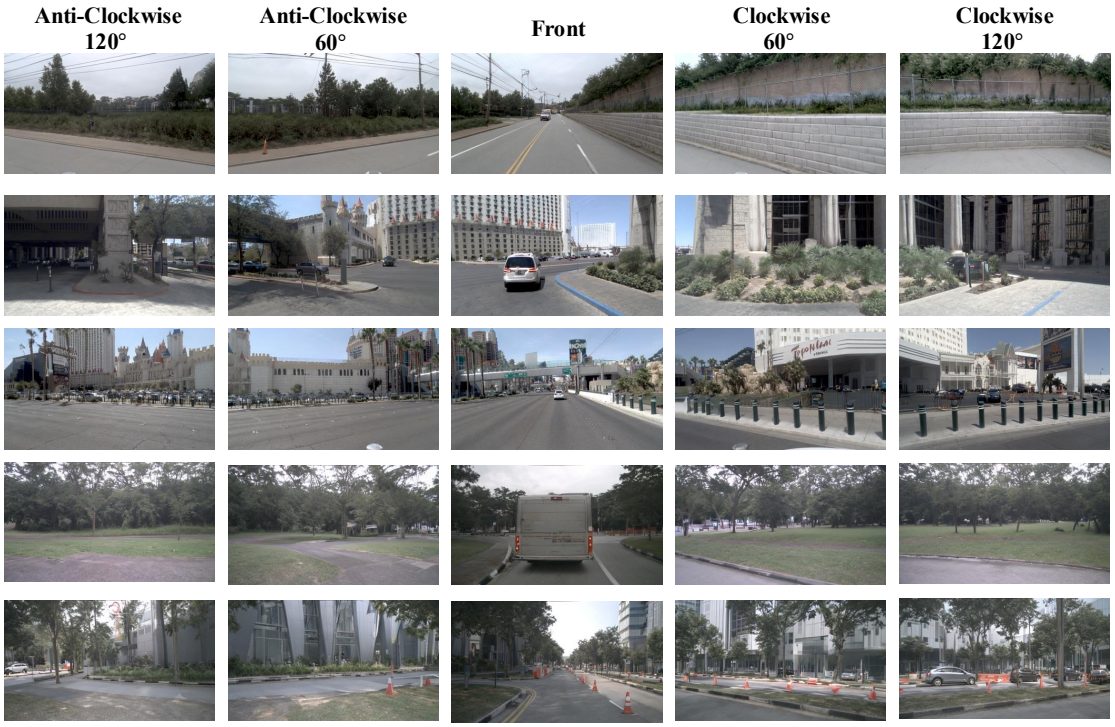


Figure D.1: Zero-Shot to Unseen Waymo Open Dataset Camera Configuration.



(a) Camera Rotation.



(b) Camera Shifting.

Figure D.2: Novel View Synthesis with Camera Rotation and Shifting.

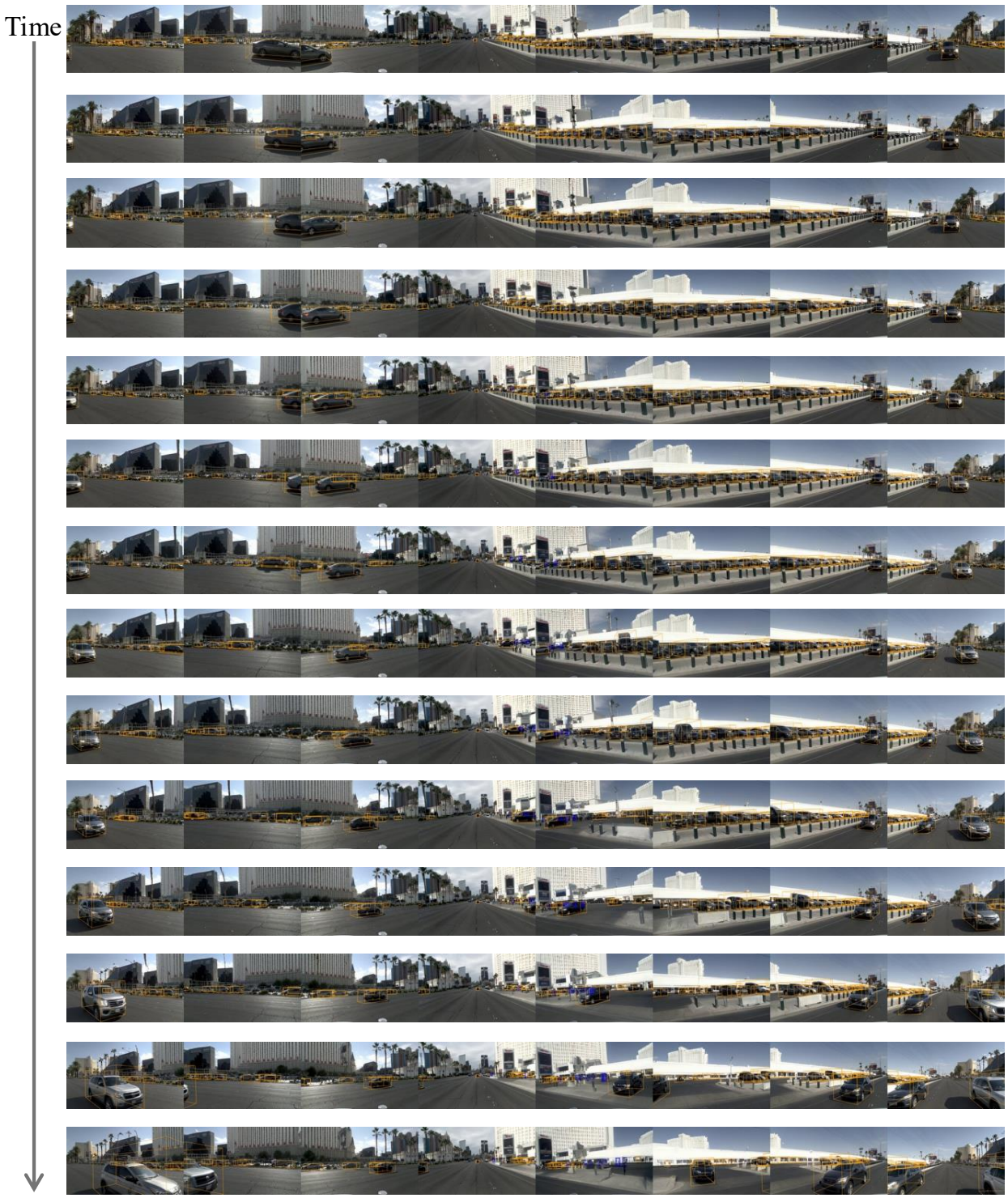


Figure D.3: Multi-View Video Generation with Eight Cameras.

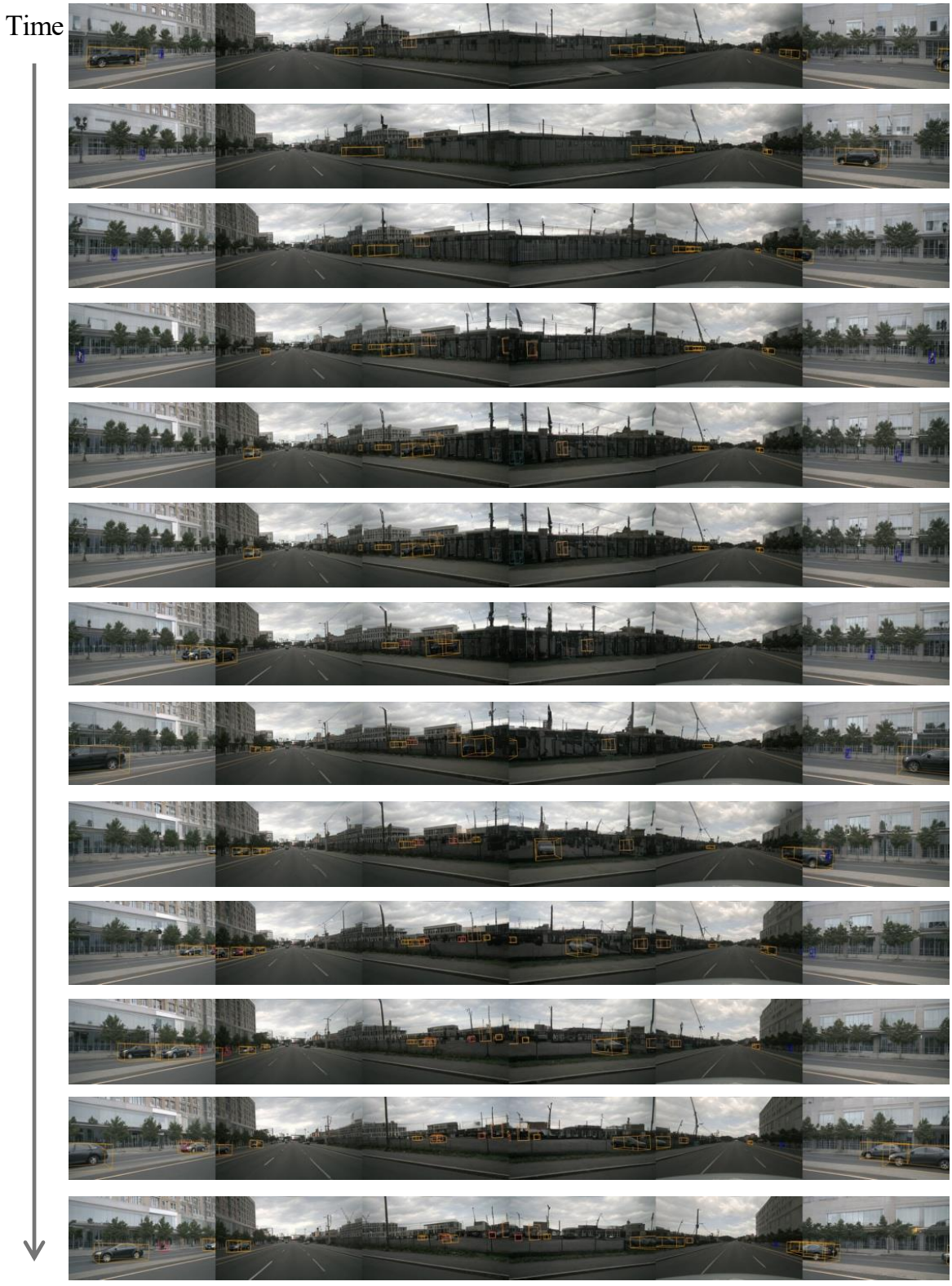


Figure D.4: Multi-View Video Generation with Six Cameras.