Paper Review

MLCF

Published as a conference paper at ICLR 2025

Navigation-Guided Sparse Scene Representation for End-to-End Autonomous Driving

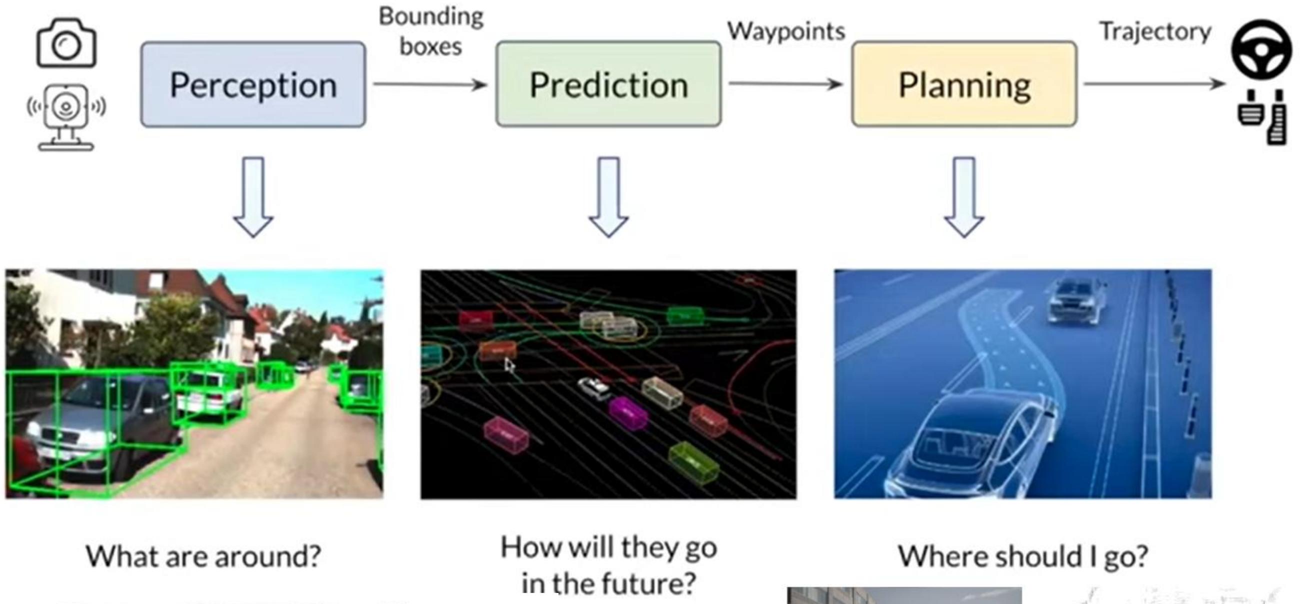
Peidong Li, Dixiao Cui
Zhijia Technology, Suzhou, China
{lipeidong,cuidixiao}@smartxtruck.com

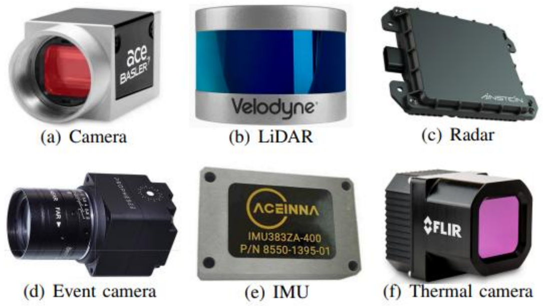
Susang Kim

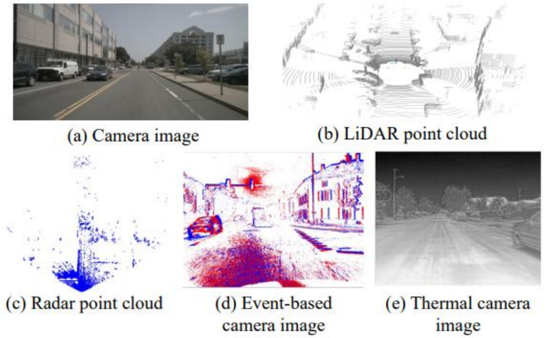
Contents

1.Introduction - Autonomous Driving

Challenge | Various weathers, illuminations, and scenarios

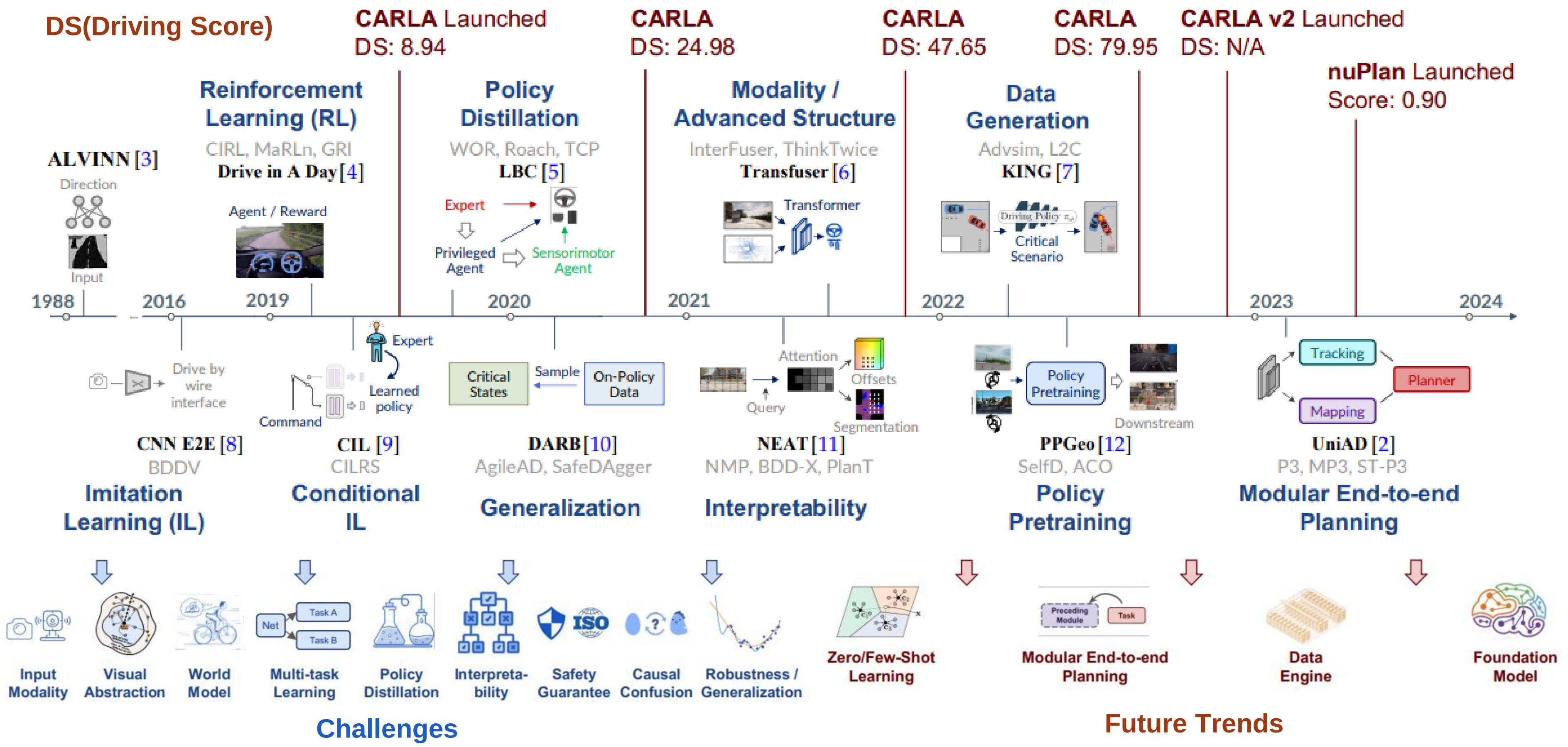






End-to-End Autonomy: A New Era of Self-Driving CVPR 2024 Tutorial https://wayve.ai/cvpr-e2ead-tutorial/
Liu, Mingyu, et al. "A survey on autonomous driving datasets: Statistics, annotation quality, and a future outlook." Transactions on Intelligent Vehicles 2024.

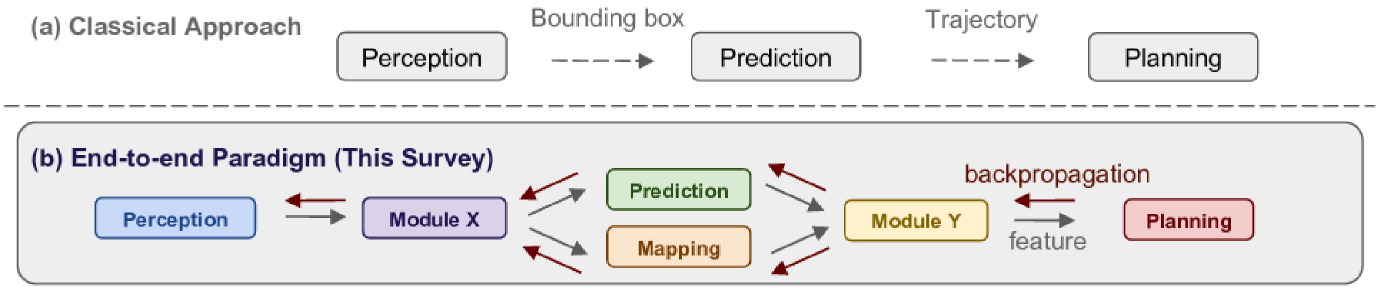
1.Introduction - Roadmap of End-to-end Autonomous Driving

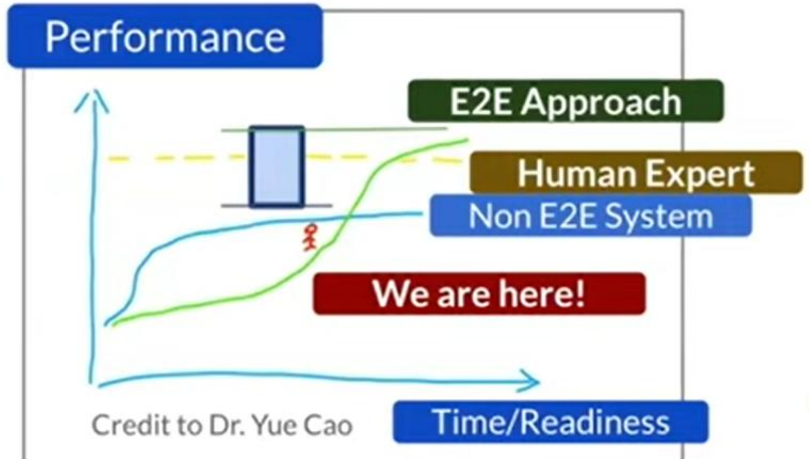


Chen, Li, et al. "End-to-end autonomous driving: Challenges and frontiers." TPAMI 2024.

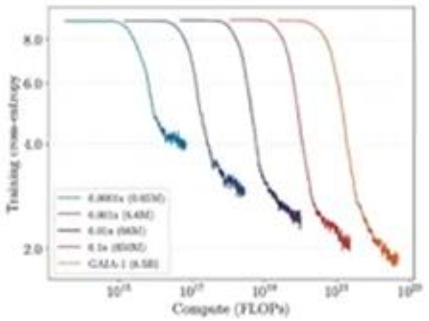
1.Introduction - End-to-End Autonomous Driving (E2EAD)

Pipeline
Section 1





+ **Scaling law:** massive amount of data + infra/compute —> **strong generalization**

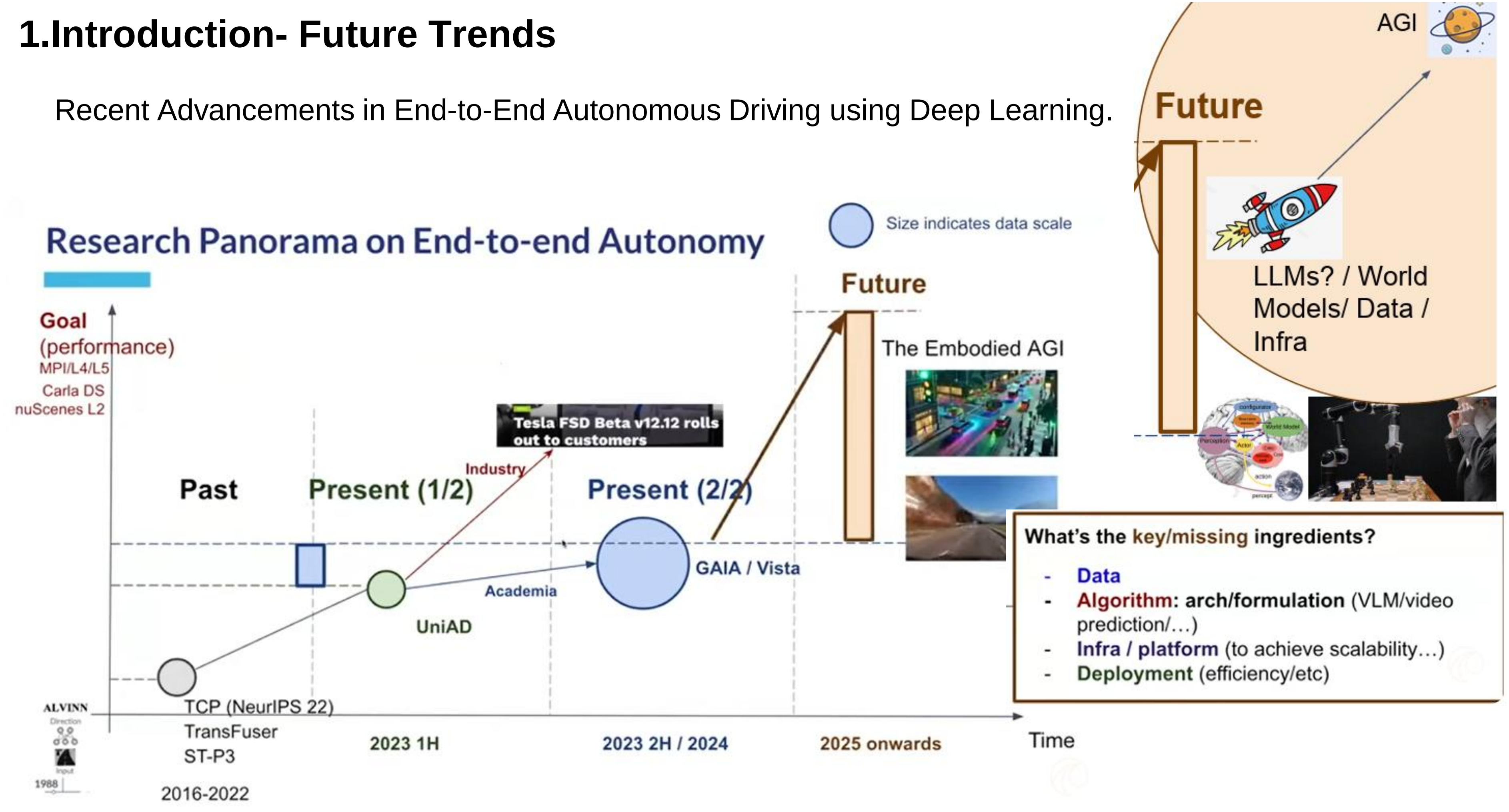


Hu et al. GAIA-1: A Generative World Model for Autonomous Driving.

Chen, Li, et al. "End-to-end autonomous driving: Challenges and frontiers." TPAMI 2024.
End-to-End Autonomy: A New Era of Self-Driving CVPR 2024 Tutorial https://wayve.ai/cvpr-e2ead-tutorial/

1.Introduction- Future Trends

Recent Advancements in End-to-End Autonomous Driving using Deep Learning.



What's the key/missing ingredients?

- **Data**
- **Algorithm: arch/formulation** (VLM/video prediction/…)
- **Infra / platform** (to achieve scalability…)
- **Deployment** (efficiency/etc)

1.Introduction - nuScenes (CVPR 2020)

Sensor	Details
6x Camera	RGB, 12Hz capture frequency, 1/1.8” CMOS sensor, 1600 × 900 resolution, auto exposure, JPEG compressed
1x Lidar	Spinning, 32 beams, 20Hz capture frequency, 360° horizontal FOV, $-30°$ to $10°$ vertical FOV, $\leq 70m$ range, ±2cm accuracy, up to $1.4M$ points per second.
5x Radar	$\leq 250m$ range, 77GHz, FMCW, 13Hz capture frequency, ±0.1km/h vel. accuracy
GPS & IMU	GPS, IMU, AHRS. 0.2° heading, 0.1° roll/pitch, 20mm RTK positioning, 1000Hz update rate

Table 2. Sensor data in nuScenes.

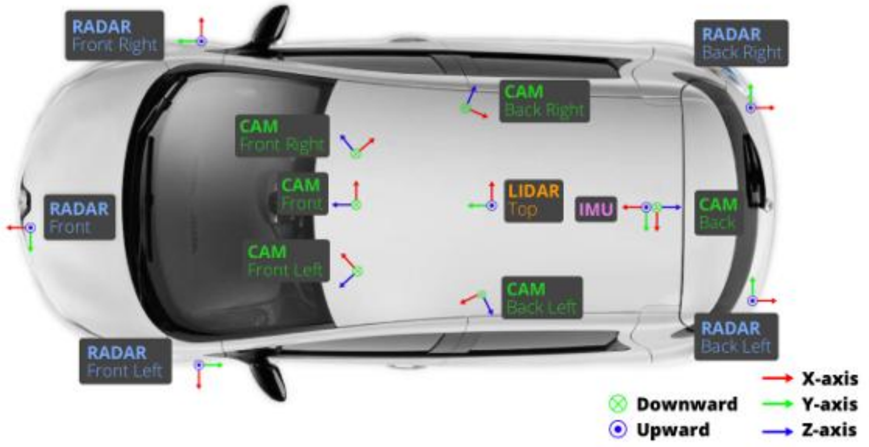


Figure 4. Sensor setup for our data collection platform.

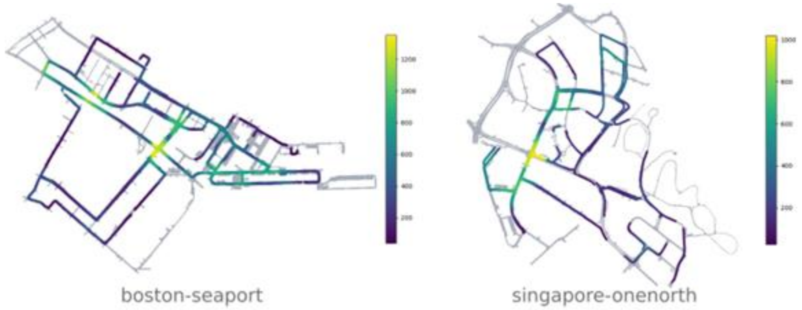
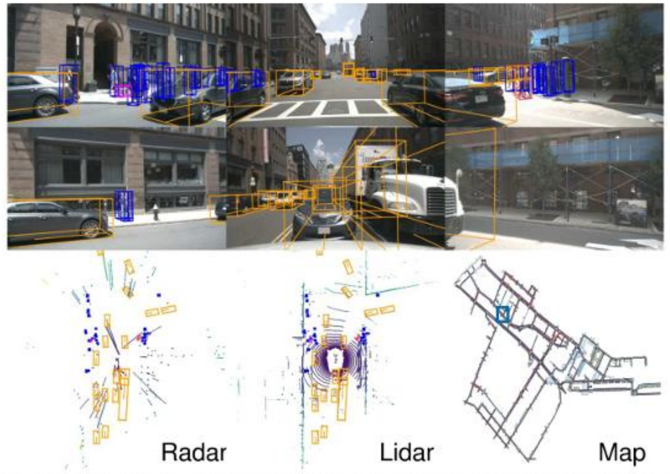


Figure 5. Spatial data coverage for two nuScenes locations. Colors indicate the number of keyframes with ego vehicle poses within a 100m radius across all scenes.

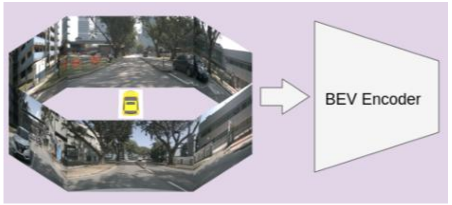


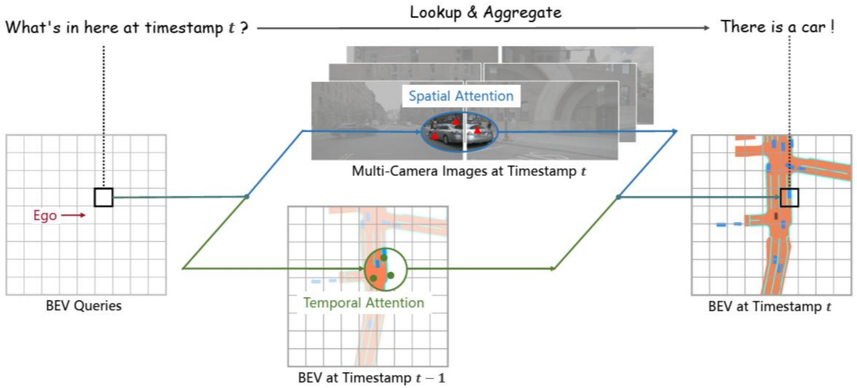
"Ped with pet, bicycle, car makes a u-turn, lane change, peds crossing crosswalk"

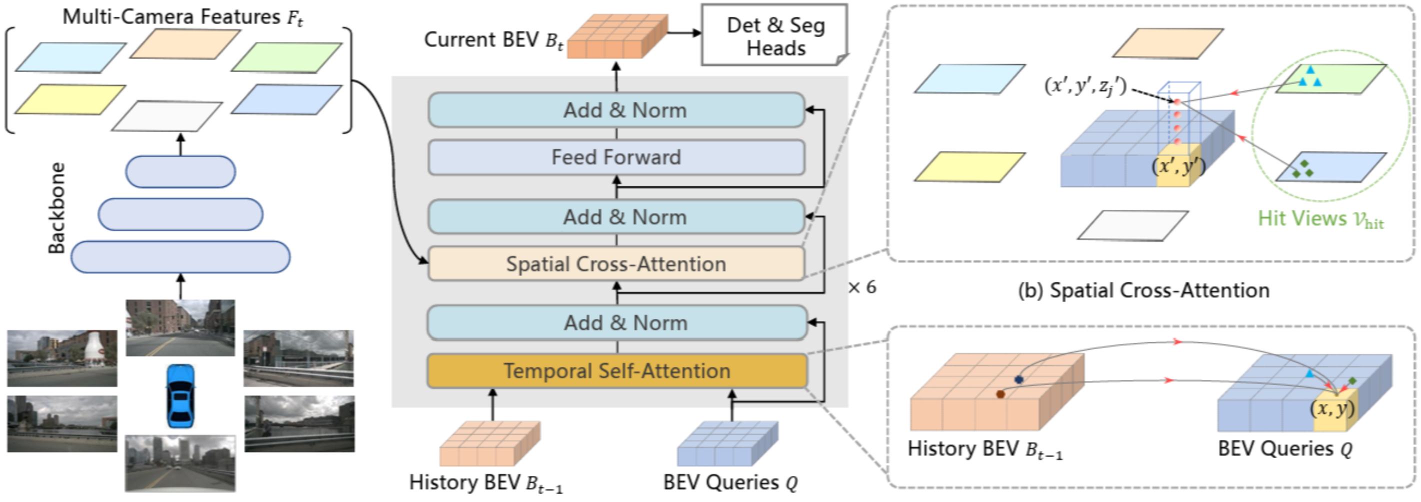
Caesar, Holger, et al. "nuscenes: A multimodal dataset for autonomous driving." CVPR. 2020.

2.Related Works - BEVFormer (ECCV 2022)

Converting **multi-camera image features to bird's-eye-view (BEV) features** can provide a unified surrounding environment representation for various autonomous driving perception tasks.







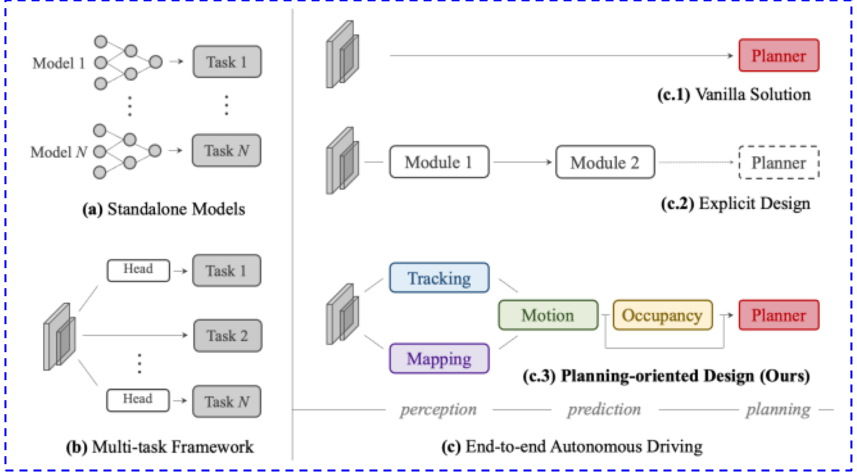
Multi-view Input at Time t (a) Overall Architecture (b) Spatial Cross-Attention (c) Temporal Self-Attention

Li, Z., et al. "BEVFormer: Learning Bird's-Eye-View Representation from Multi-Camera Images via Spatiotemporal Transformers.", ECCV 2022.

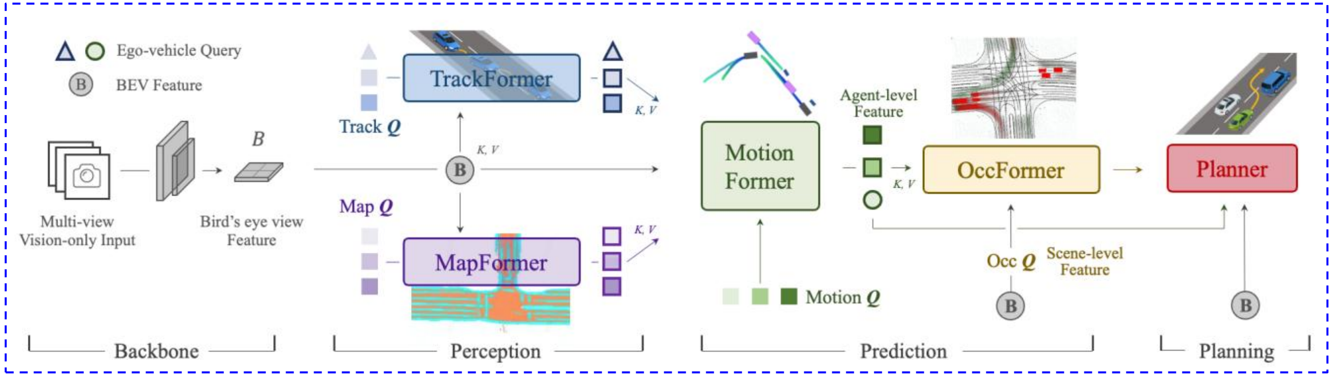
2.Related Works - Unified Autonomous Driving (UniAD) (CVPR 2023)

A comprehensive framework up-to-date that incorporates **full-stack driving tasks in one network**.

Design	Approach	Perception Det.	Perception Track	Perception Map	Prediction Motion	Prediction Occ.	Plan
(b)	NMP [101]	✓			✓		✓
	NEAT [19]			✓			✓
	BEVerse [105]	✓		✓		✓	
(c.1)	[14, 16, 78, 97]						✓
(c.2)	PnPNet† [57]	✓	✓		✓		
	ViP3D† [30]	✓	✓		✓		
	P3 [82]					✓	✓
	MP3 [11]			✓		✓	✓
	ST-P3 [38]			✓		✓	✓
	LAV [15]	✓		✓	✓		✓
(c.3)	**UniAD** (ours)	✓	✓	✓	✓	✓	✓



It is designed to integrate module advantages, **offering complementary feature abstractions for agent interaction from a global perspective**. Tasks communicate via **unified query interfaces to enhance collaborative planning**.

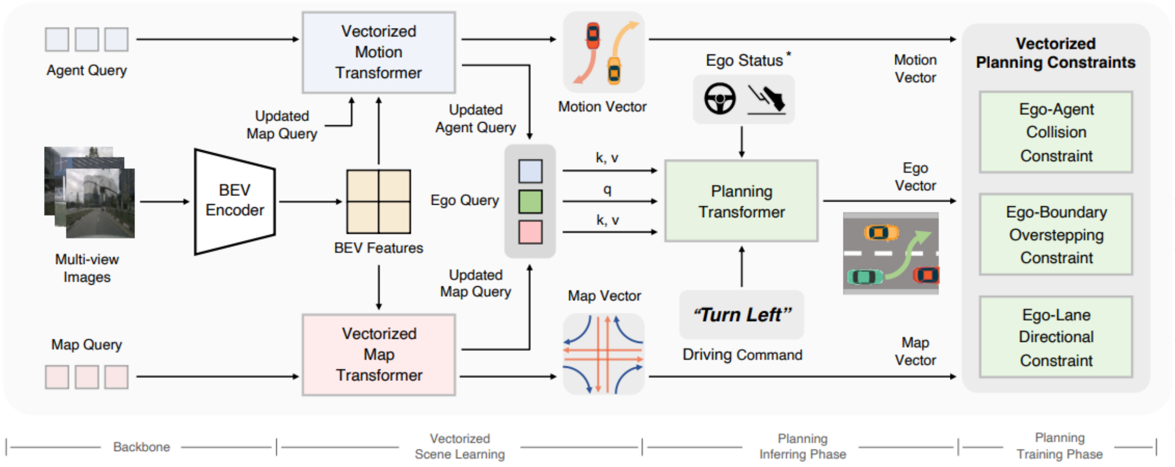


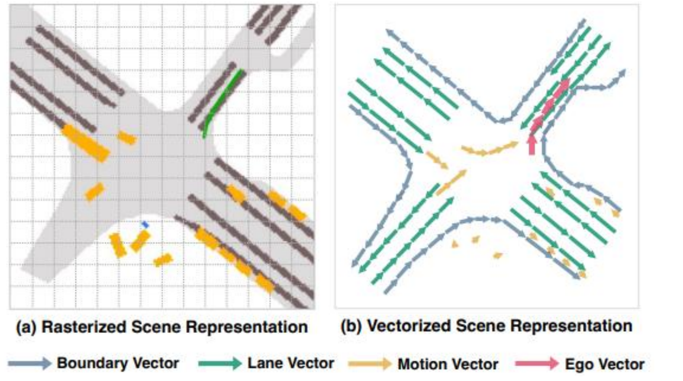
TrackFormer + MapFormer + MotionFormer + OccFormer

HU, Yihan, et al. Planning-oriented autonomous driving. CVPR 2023.

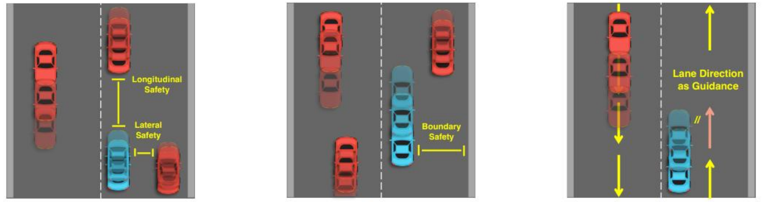
2.Related Works – Vectorized Autonomous Driving (VAD) (ICCV 2023)

VAD (Vectorized Autonomous Driving), an end-to-end vectorized paradigm for autonomous driving. VAD models **the scene in a fully vectorized way** (i.e., vectorized agent motion and map), getting rid of computationally intensive rasterized representation.




(a) Rasterized Scene Representation
(b) Vectorized Scene Representation
Boundary Vector
Lane Vector
Motion Vector
Ego Vector

Method	L2 (m) ↓ 1s	2s	3s	Avg.	Collision (%) ↓ 1s	2s	3s	Avg.	Latency (ms)	FPS
NMP† [49]	-	-	2.31	-	-	-	1.92	-	-	-
SA-NMP† [49]	-	-	2.05	-	-	-	1.59	-	-	-
FF† [18]	0.55	1.20	2.54	1.43	0.06	0.17	1.07	0.43	-	-
EO† [24]	0.67	1.36	2.78	1.60	0.04	0.09	0.88	0.33	-	-
ST-P3 [19]	1.33	2.11	2.90	2.11	0.23	0.62	1.27	0.71	628.3	1.6
UniAD [21]	0.48	0.96	1.65	1.03	**0.05**	**0.17**	0.71	0.31	555.6	1.8
VAD-Tiny	0.46	0.76	1.12	0.78	0.21	0.35	0.58	0.38	**59.5**	**16.8**
VAD-Base	**0.41**	**0.70**	**1.05**	**0.72**	0.07	**0.17**	**0.41**	**0.22**	224.3	4.5



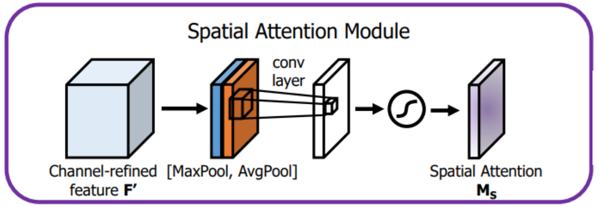
Ego-Agent Collision Constraint — Ego-Boundary Overstepping Constraint — Ego-Lane Directional Constraint

Vectorized Scene Learning – Encodes scene information into agent and map queries, representing the scene with motion and map vectors. **Inferring Phase (Planning)** – **Uses an ego query** to extract map and agent information through query interaction, producing the planning trajectory (ego vector).

Bo Jiang, et al. VAD: Vectorized Scene Representation for Efficient Autonomous Driving. ICCV 2023.

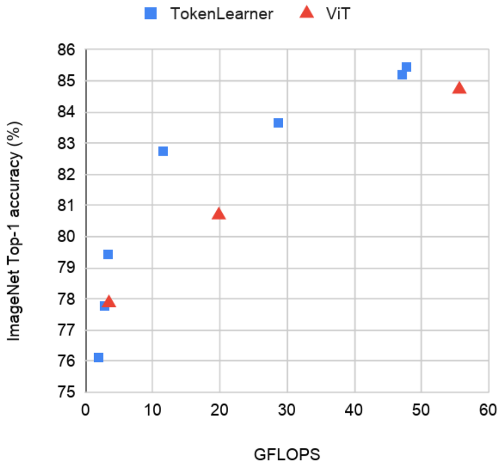
2.Related Works - TokenLearner (NeurIPS 2021)

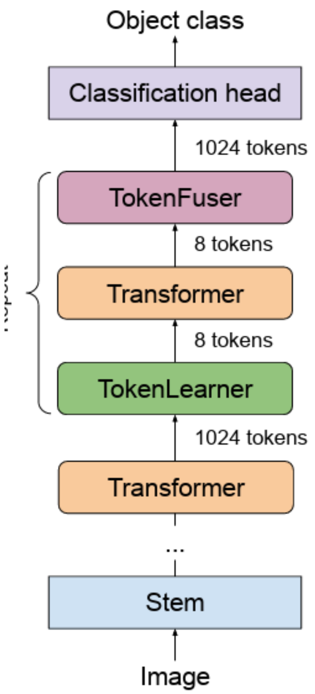
TokenLearner for visual representation learning, adaptively tokenizes the inputs. The goal is to learn to **extract important tokens** in images and video frames for the recognition tasks.

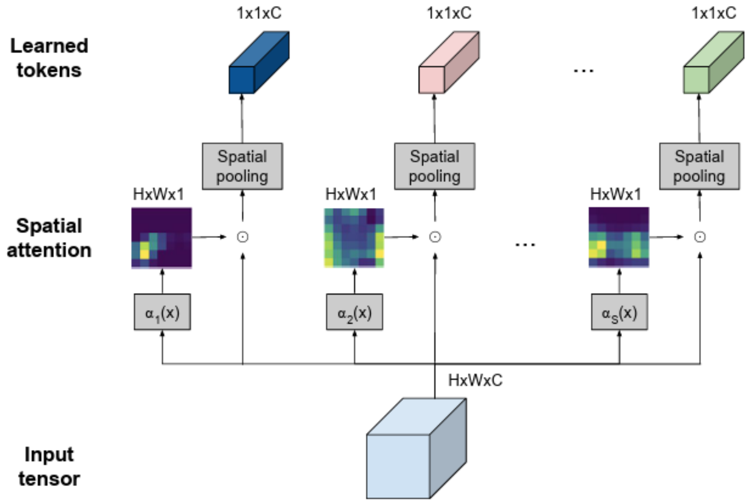


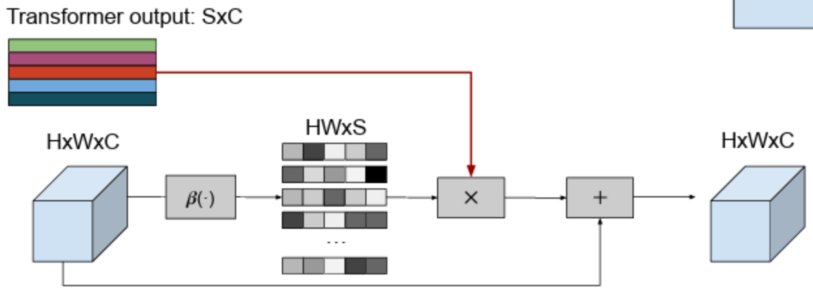
$$\mathbf{M}_s(F) = \sigma\left(f^{7x7}([\mathrm{AvgPool}(F); \mathrm{MaxPool}(F)])\right)$$

$$\mathbf{M}_s(F) = \sigma\left(f^{7x7}\left(\left[\mathbf{F}^s_{avg}; \mathbf{F}^s_{max}\right]\right)\right)$$





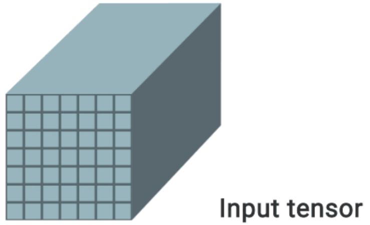




Ryoo, Michael S., et al. "Tokenlearner: What can 8 learned tokens do for images and videos?." NeurIPS 2021.
Spatial Attention : https://paperswithcode.com/method/spatial-attention-module

2.Related Works - TokenLearner (NeurIPS 2021)

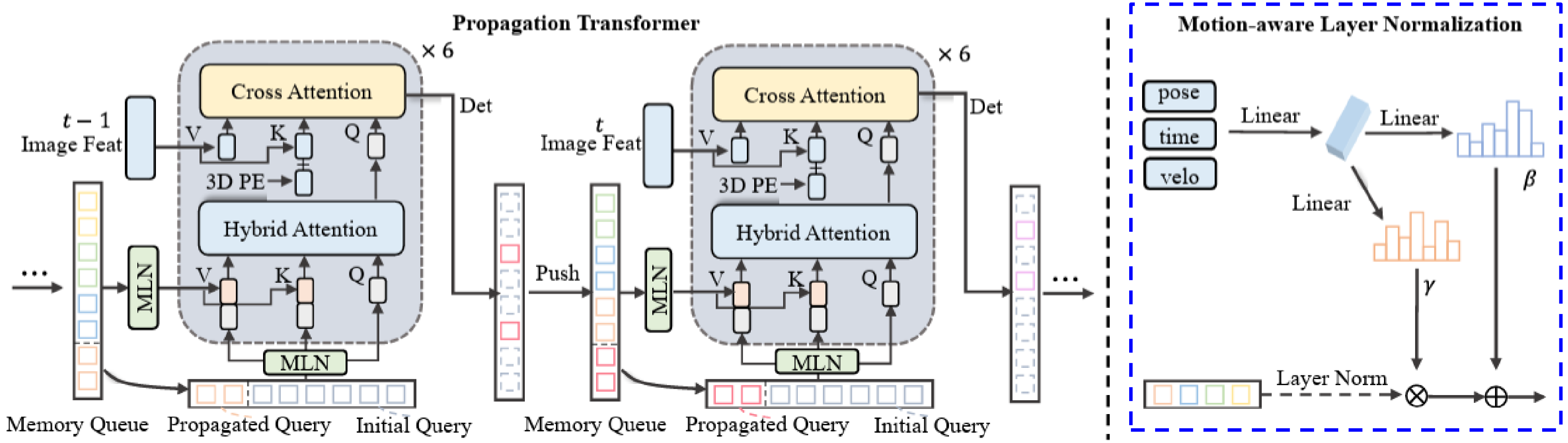
1) We enable the adaptive tokenization so that the **tokens can be dynamically selected** conditioned on the input.
2) This also **effectively reduces the total number of tokens for the transformer**, which is particularly beneficial considering that there are many tokens in videos and the computation is quadratic to the number of tokens.
3) Finally, we provide an ability for each subsequent layer to learn to rely on different space-time tokenizations, potentially **allowing different layers to capture different aspects** of the video.



Ryoo, Michael S., et al. "Tokenlearner: What can 8 learned tokens do for images and videos?." NeurIPS 2021.

2.Related Works - Motion aware Layer Normalization (MLN)

Motion-aware layer normalization (MLN) enables the modeling of object movement.



Flatten

$$\gamma = \xi_1(\underbrace{E_{t-1}^{t}, v, \Delta t}),$$
$$\beta = \xi_2(E_{t-1}^{t}, v, \Delta t)$$

two linear layers

$$E_{t-1}^{t} = E_t^{inv} \cdot E_{t-1}$$

the ego pose matrix $\tilde{Q}_p^t = E_{t-1}^t \cdot Q_p^{t-1}$

The velocity v and time interval Δt of the current frame are zero-initialized

motion-aware position encoding

MLP

$$\tilde{Q}_{pe}^t = \gamma \cdot LN(\psi(\tilde{Q}_p^t)) + \beta,$$
$$\tilde{Q}_c^t = \gamma \cdot LN(Q_c^t) + \beta$$

motion-aware context embedding

Wang, Shihao, et al. "Exploring object-centric temporal modeling for efficient multi-view 3d object detection." ICCV 2023.

3.Method - Comparison of Various End-to-End Paradigms

Fig. 2(a), Existing methods typically **extract all perception elements** by following previous BEV perception paradigms.

Fig. 2(b), SSR directly **extracts only the essential perception elements** in the guidance of navigation commands, thereby minimizing redundancy.

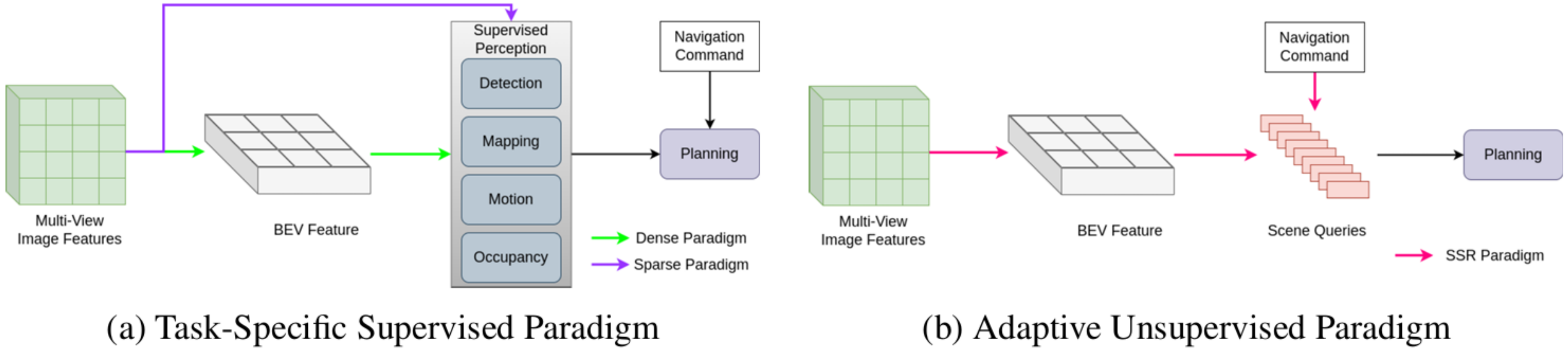


Figure 2: **Comparison of Various End-to-End Paradigms.** Compared to previous task-specific supervised paradigms, our adaptive unsupervised approach takes full advantage of end-to-end framework by utilizing navigation-guided perception, without the need to differentiate between sub-tasks.

3.Method - Sparse Scene Representation (SSR)

SSR delivers **SOTA on the nuScenes dataset, with minimal computational overhead**.
SSR decreases average L2 error by 0.28 meters and reduces the average collision rate by 51.6% relatively compared to UniAD, even **without any annotations**.
SSR achieves **superior performance on CARLA**'s Town05 Long benchmark.
SSR reduces **training time to 1/13th of that required by UniAD** and is **10.9× faster during inference**.
SSR has the potential to manage large-scale data in real-time applications.

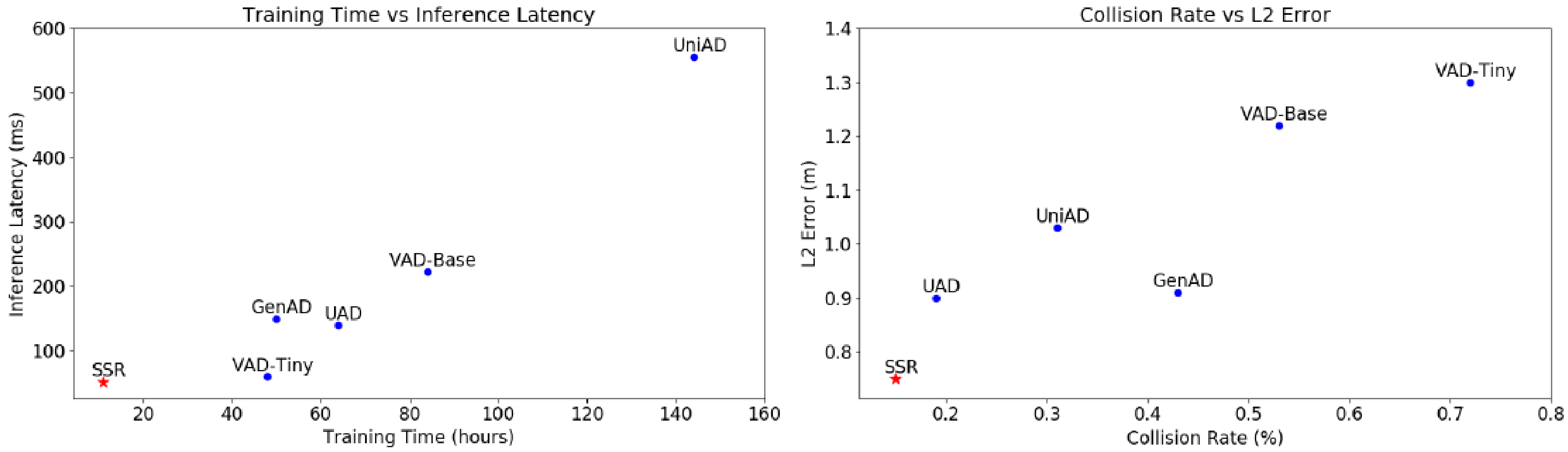
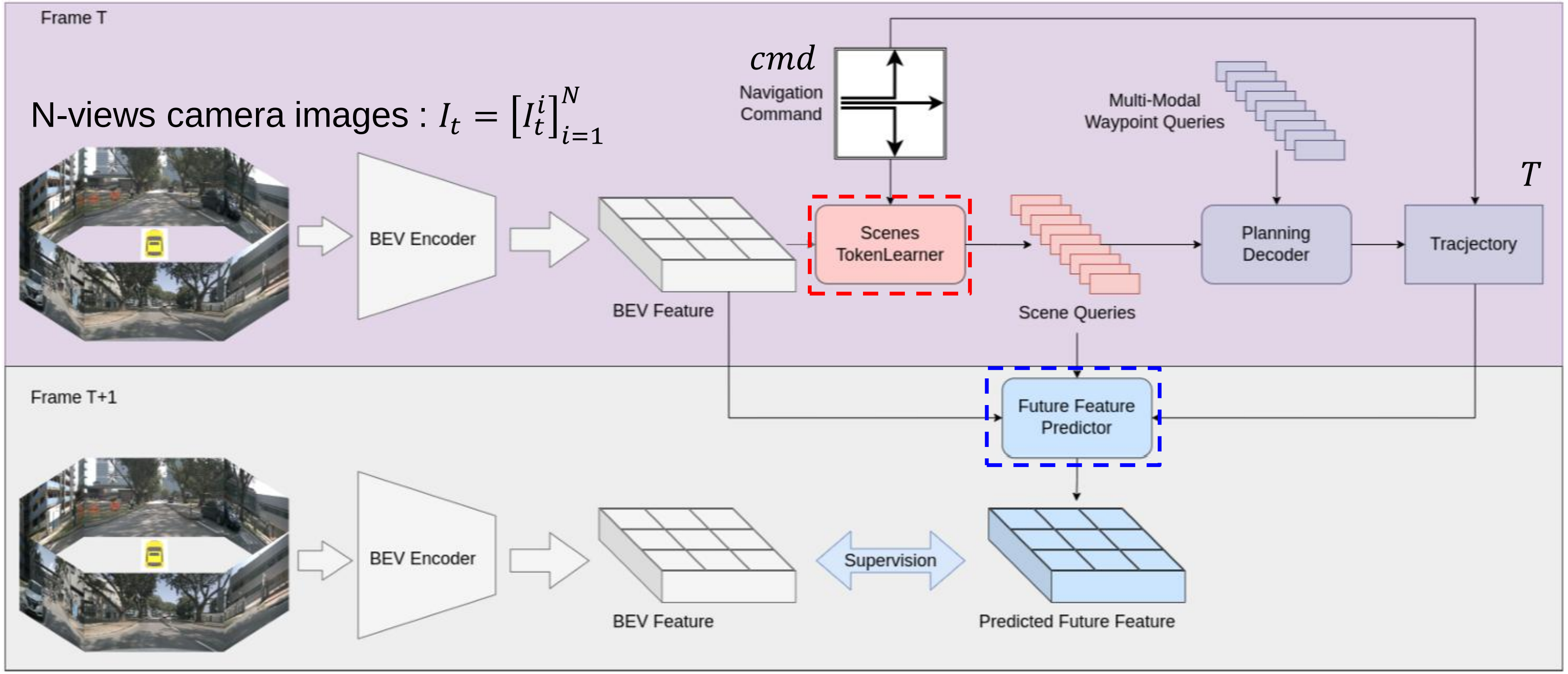


Figure 1: **Performance Comparison of Various Methods in Speed and Accuracy on nuScenes.**

3.Method – Overview of SSR

SSR has two components: the purple part, used in both training and inference.
the gray part, used only during training.

SSR's core is the Scenes **TokenLearner** module, which extracts crucial scene information using **16 tokens** instead of dense **BEV feature numerous queries(hundreds).** These sparse tokens generate the planning trajectory and are enhanced by a **self-supervised future feature predictor** for improved representation.

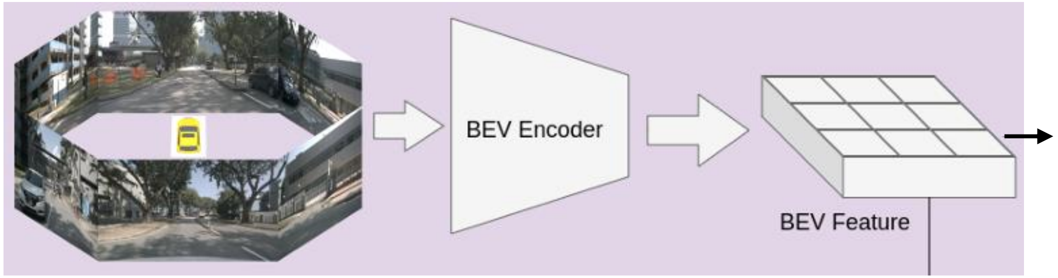


3.Method – BEV Feature Construction & Scene Queries

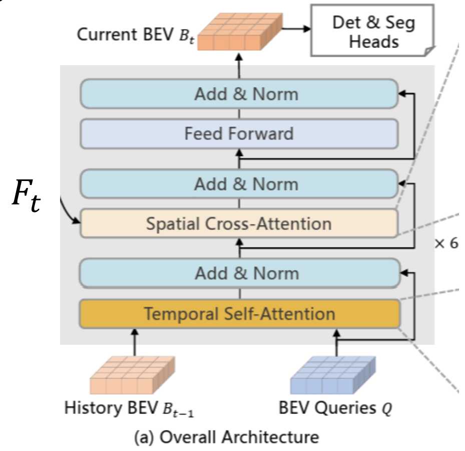
The vision-based E2EAD model predicts the planning trajectory T, a set of points in BEV space.

N-views camera images : $I_t = [I_t^i]_{i=1}^N$ ➔ ResNet-(34, 50, 101) ➔ N-views camera features : $F_t = [F_t^i]_{i=1}^N$

640 × 360

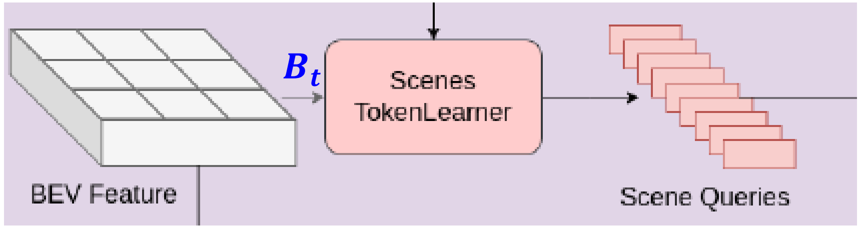


The current BEV feature $B_t \in \mathbb{R}^{H \times W \times C}$



A BEV query Q ∈ $\mathbb{R}^{H \times W \times C}$, B_{t-1} the previous frame's BEV feature

$$\mathbf{Q} = CrossAttention(\mathbf{Q}, \mathbf{B}_{t-1}, \mathbf{B}_{t-1}),$$
$$\mathbf{B}_t = CrossAttention(\mathbf{Q}, \mathbf{F}_t, \mathbf{F}_t).$$



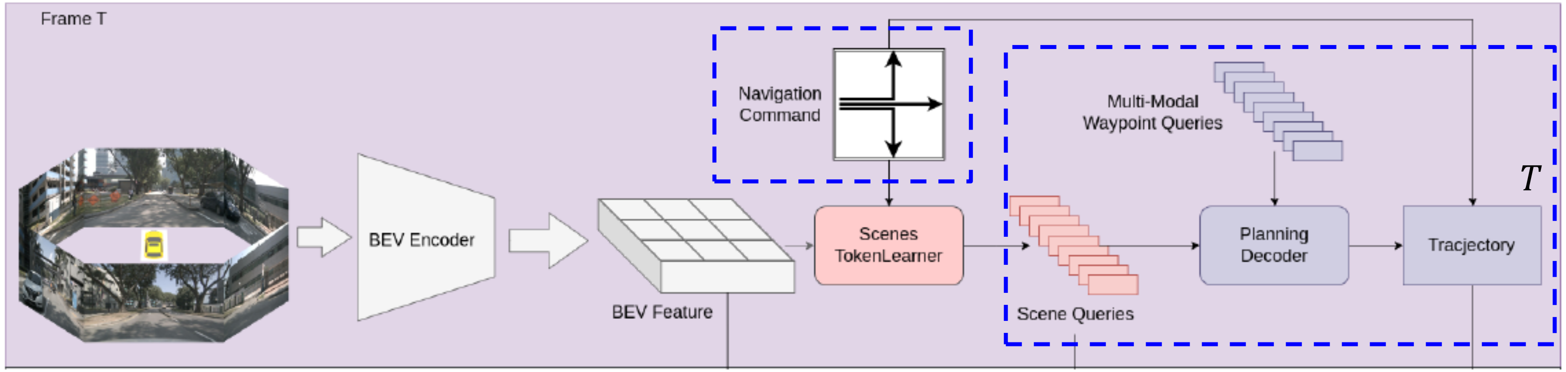
scene queries : $S_t = [s_t^i]_{i=1}^{N_s} \in \mathbb{R}^{N_s \times C}$

the number of scene queries : N_s

3.Method – Planning Based on Sparse Scene Representation

Navigation command : *cmd*

(Trun Right, Turn Left, Go Straight)



Scene Queries : S_t

$$\mathbf{W}_t = CrossAttention(\mathbf{W}_t, \mathbf{S}_t, \mathbf{S}_t).$$

$$\mathbf{T} = Select(MLP(\mathbf{W}_t), cmd).$$

$\mathrm{T} \in \mathbb{R}^{N_t \times 2}$

the number of future timestamps x 2(x,y)

A way point query represents a spatial location or trajectory point that the model uses to predict the agent's path.

a set of way point queries : $W_t \in \mathbb{R}^{N_m \times N_t \times C}$

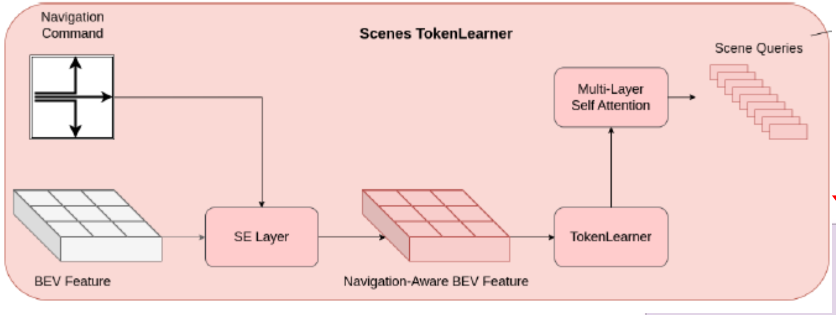
$$\mathcal{L}_{imi} = \|\mathbf{T}_{GT} - \mathbf{T}\|_1.$$

imitation loss (L1 loss)

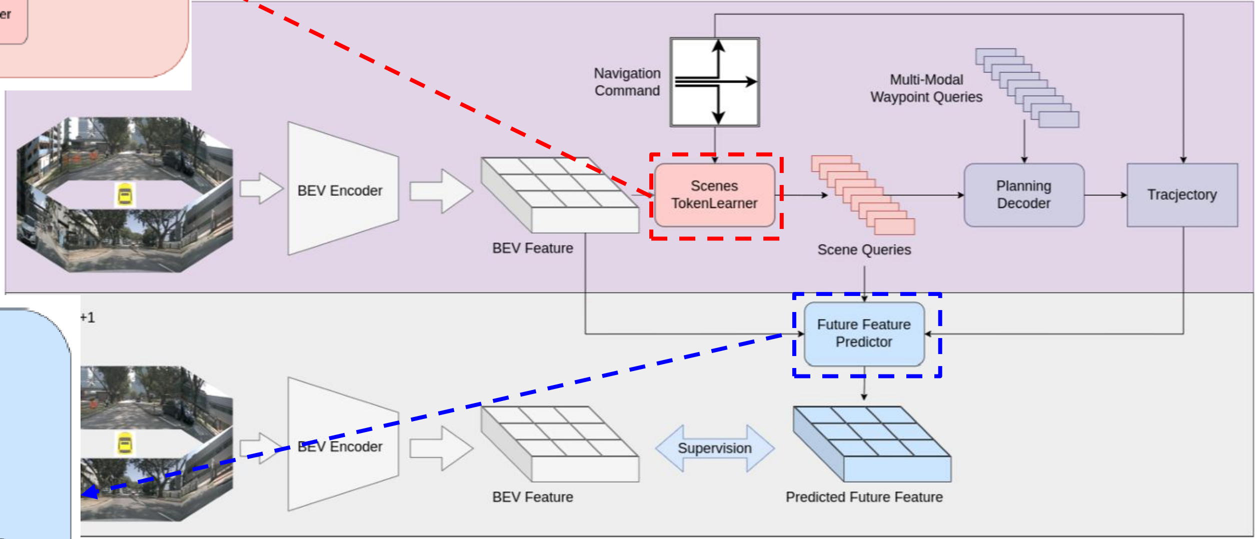
N_t : the number of future timestamps

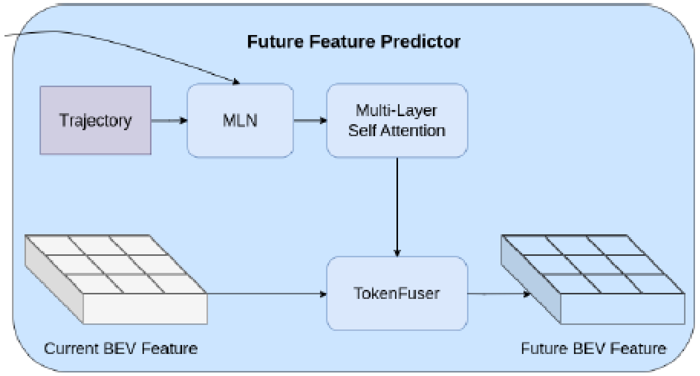
N_m : the number of driving commands

3.Method – Structure of Modules (Scene Token Learner & Future Feature Predictor)



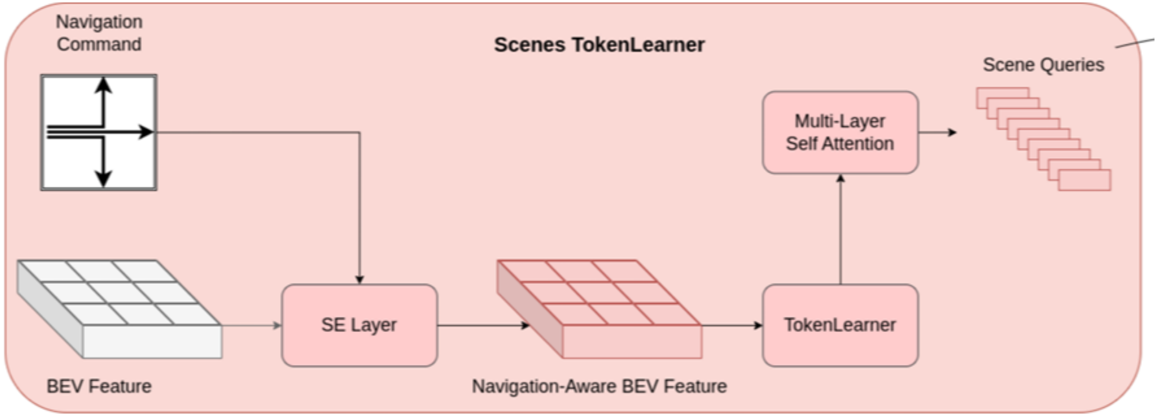
BEV features provide rich perception information but increase inference time. To overcome this, we introduce **a sparse scene representation with adaptive spatial attention**, reducing computational load while preserving scene understanding.





Enhance scene representation through **self-supervised temporal context**, ensuring predicted future scenes align with real ones.

3.Method – Scenes Token Learner (STL)



$$\mathbf{S}_t = TL_{BEV}(\mathbf{B}_t^{navi}).$$

The navigation-aware BEV feature is then passed into the BEV TokenLearner **to adaptively focus on** the most important information.

$$\mathbf{B}_t^{navi} = SE(\mathbf{B}_t, cmd).$$

Squeeze-and-Excitation layer to encode the navigation command cmd into the dense BEV feature, producing the **navigation-aware BEV feature**

$$\mathbf{s}_i = M_i(\mathbf{B}_t^{navi}) = \rho(\mathbf{B}_t^{navi} \odot \varpi_i(\mathbf{B}_t^{navi})),$$

➔ TokenLearner

For each scene query s_i, we adopt a tokenizer function M_i that maps B_t^{navi} into a token vector: $\mathbb{R}^{H \times W \times C} \to \mathbb{R}^C$. The tokenizer predicts spatial attention maps of shape $H \times W \times 1$, and the learned scene tokens are obtained through global average pooling

$$\mathbf{S}_t = SelfAttention(\mathbf{S}_t).$$

ϖ(·) is the spatial attention function and ρ(·) is the global average pooling function. The multi-layer self-attention is applied to further enhance the scene queries

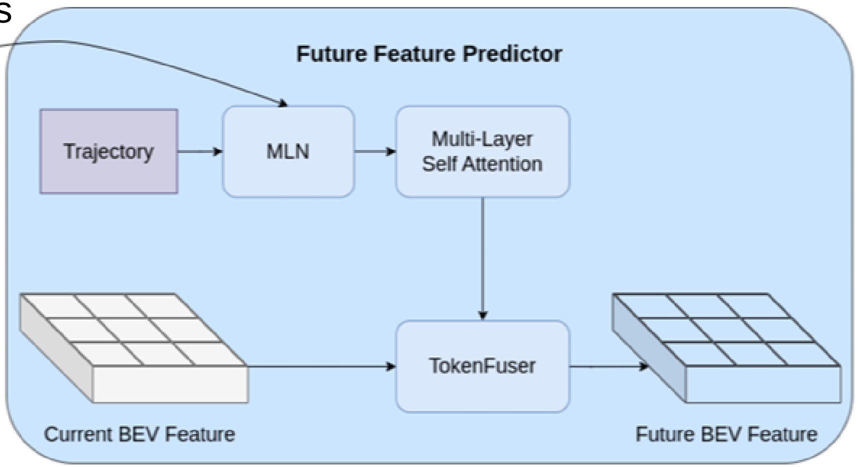
3.Method – Future Feature Predcitor (FFP)

Scene Queries

S_t

$T \in \mathbb{R}^{N_t \times 2}$

(x,y)



Prioritize temporal context to enhance scene representation through **self-supervision rather than perception sub-tasks**. If our predicted actions correspond to real actions**, the predicted future scenes should closely resemble the actual future scenes**. (Scene Queries, Trajectory, Current BEV)

$\mathbf{D}_t = MLN(\mathbf{S}_t, \mathbf{T}).$ Motion aware Layer Normalization (MLN) helps current scene queries **encode motion information(T)**

$\hat{\mathbf{S}}_{t+1} = SelfAttention(\mathbf{D}_t).$ to **predict the future scene queries** $\hat{S}_{t+1} \in \mathbb{R}^{N_s \times C}$

$\hat{\mathbf{B}}_{t+1} = TokenFuser(\hat{\mathbf{S}}_{t+1}, \mathbf{B}_t) = \boxed{\psi}(\mathbf{B}_t) \otimes \hat{\mathbf{S}}_{t+1},$ **recover the BEV feature** from the predicted scene queries for further **self-supervision**.

$\hat{B}_{t+1} \in \mathbb{R}^{H \times W \times C}$

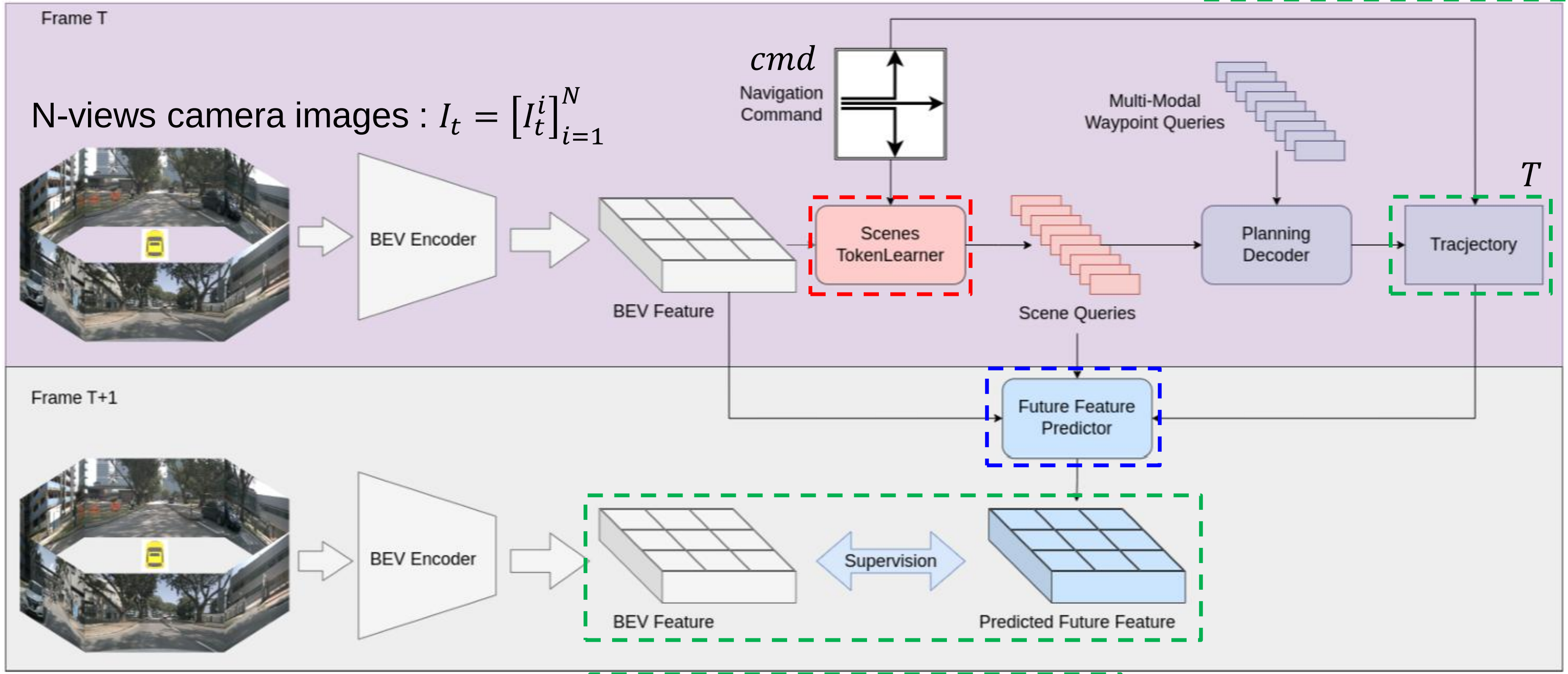
MLP with the sigmoid function to remap $\mathbb{R}^{H \times W \times C} \rightarrow \mathbb{R}^{H \times W \times N_s}$

$\mathcal{L}_{bev} = \|\hat{\mathbf{B}}_{t+1} - \mathbf{B}_{t+1}\|_2.$ L2 loss with the real future BEV feature

3.Method – Overall Architecture (recap)

$$\mathbf{T} = Select(MLP(\mathbf{W}_t), cmd).$$

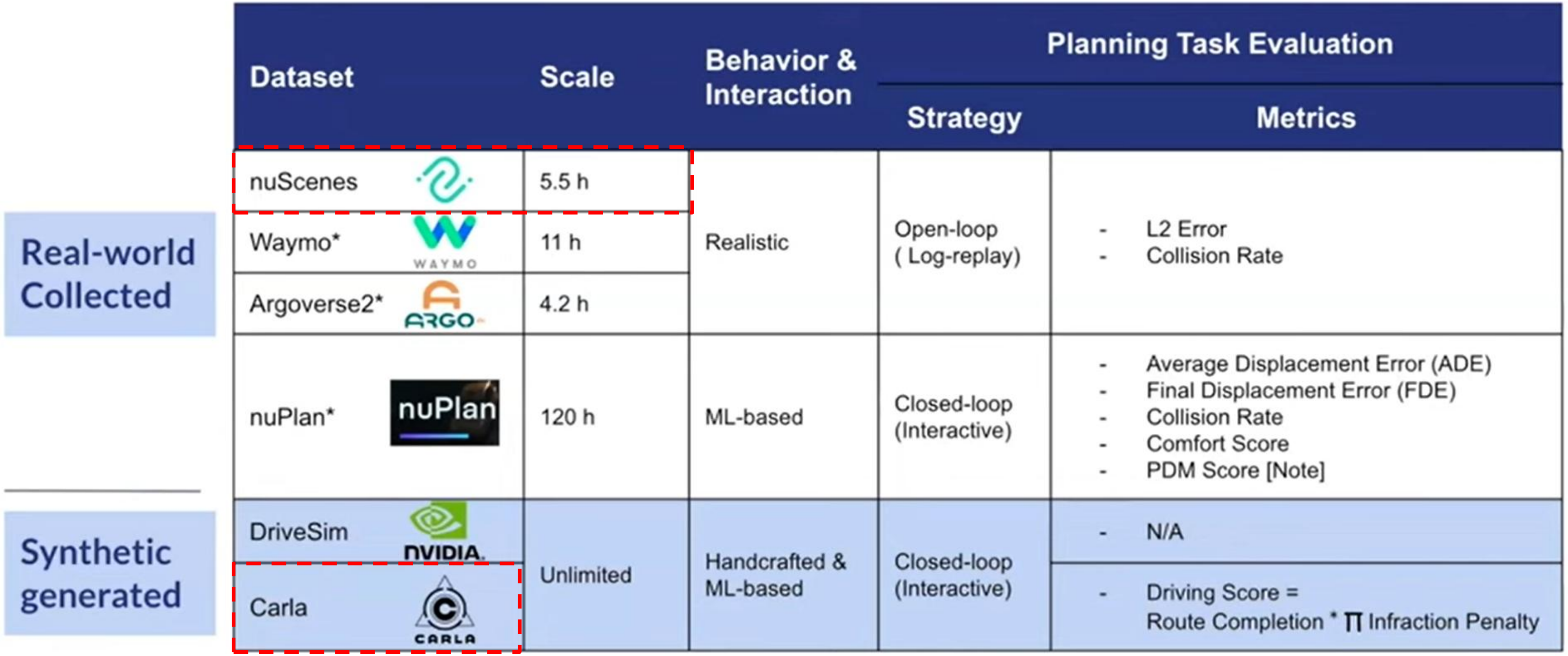
$$\mathcal{L}_{imi} = \|\mathbf{T}_{GT} - \mathbf{T}\|_1.$$



$$\mathcal{L}_{bev} = \|\hat{\mathbf{B}}_{t+1} - \mathbf{B}_{t+1}\|_2.$$

$\mathcal{L}_{total} = \mathcal{L}_{imi} + \mathcal{L}_{bev}.$ predicted trajectory + BEV reconstruction loss for the predicted BEV feature

4.Experiments – Dataset



	Dataset	Scale	Behavior & Interaction	Planning Task Evaluation	
				Strategy	Metrics
Real-world Collected	nuScenes	5.5 h	Realistic	Open-loop (Log-replay)	- L2 Error - Collision Rate
	Waymo*	11 h			
	Argoverse2*	4.2 h			
	nuPlan*	120 h	ML-based	Closed-loop (Interactive)	- Average Displacement Error (ADE) - Final Displacement Error (FDE) - Collision Rate - Comfort Score - PDM Score [Note]
Synthetic generated	DriveSim	Unlimited	Handcrafted & ML-based	Closed-loop (Interactive)	- N/A
	Carla				- Driving Score = Route Completion * $\prod$ Infraction Penalty

nuScene : **L2 (m) error** measures trajectory accuracy, while **collision rate (%)** quantifies collisions with objects. All metrics are calculated in 3s future horizon with a 0.5s interval and evaluated at 1s, 2s and 3s.
CARLA : **Route Completion (RC)** measures the percentage of the route completed, **Infraction Score (IS)** quantifies infractions(pedestrians, vehicles, road layouts, signals) **Driving Score (DS)** is the product of RC and IS.

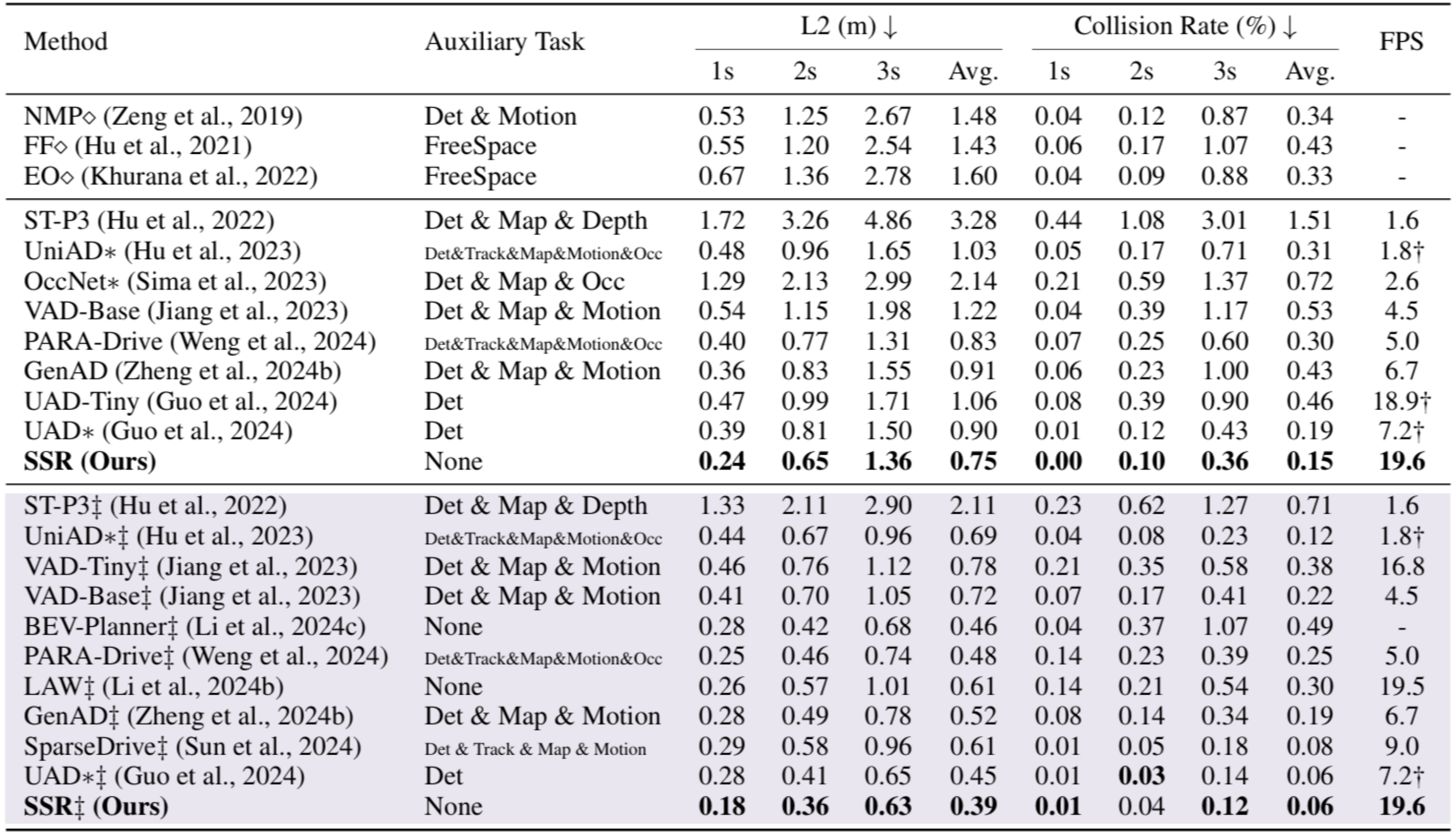
4.Experiments – Implementation Details

Settings We build up **SSR on VAD** and follow the setting of **VAD-Tiny**.
We adopt ResNet-50 as image backbone operating at an **image resolution of 640 × 360**.
The **BEV representation is generated at a 100 × 100 resolution** and then compressed into sparse scene tokens with shape **16 × 256**.
The number of **navigation commands remains 3** as prior works.
Other settings follow VAD-Tiny unless otherwise specified.
In closed-loop simulation, we utilize **ResNet-34** as the image backbone, resizing the input image size to **900 × 256**.
The target point is concatenated with **driving commands as the navigation information**.
The **Trajectory-Guided Control Prediction (TCP) head** is applied for planning module.

T

Training Parameters **Our open-loop model** is trained for **12 epochs on 8 NVIDIA RTX 3090** GPUs with a batch size of 1 per GPU.
The **training phase costs about 11 hours** which is 13× faster than UniAD.
We utilize the AdamW optimizer with a learning rate set to 5×10−5.
The weight of imitation loss and BEV loss is both 1.0.
The closed-loop model is trained for 60 epochs on **4 NVIDIA RTX 3090 GPUs** with a batch size of 32 per GPU.
The learning rate is set to 1×10−4 while being halved after 30 epochs.

4.Experiments – Comparison of SOTA on the nuScenes dataset (Open-Loop)



Method	Auxiliary Task	L2 (m) ↓				Collision Rate (%) ↓				FPS
		1s	2s	3s	Avg.	1s	2s	3s	Avg.	
NMP⋄ (Zeng et al., 2019)	Det & Motion	0.53	1.25	2.67	1.48	0.04	0.12	0.87	0.34	-
FF⋄ (Hu et al., 2021)	FreeSpace	0.55	1.20	2.54	1.43	0.06	0.17	1.07	0.43	-
EO⋄ (Khurana et al., 2022)	FreeSpace	0.67	1.36	2.78	1.60	0.04	0.09	0.88	0.33	-
ST-P3 (Hu et al., 2022)	Det & Map & Depth	1.72	3.26	4.86	3.28	0.44	1.08	3.01	1.51	1.6
UniAD∗ (Hu et al., 2023)	Det&Track&Map&Motion&Occ	0.48	0.96	1.65	1.03	0.05	0.17	0.71	0.31	1.8†
OccNet∗ (Sima et al., 2023)	Det & Map & Occ	1.29	2.13	2.99	2.14	0.21	0.59	1.37	0.72	2.6
VAD-Base (Jiang et al., 2023)	Det & Map & Motion	0.54	1.15	1.98	1.22	0.04	0.39	1.17	0.53	4.5
PARA-Drive (Weng et al., 2024)	Det&Track&Map&Motion&Occ	0.40	0.77	1.31	0.83	0.07	0.25	0.60	0.30	5.0
GenAD (Zheng et al., 2024b)	Det & Map & Motion	0.36	0.83	1.55	0.91	0.06	0.23	1.00	0.43	6.7
UAD-Tiny (Guo et al., 2024)	Det	0.47	0.99	1.71	1.06	0.08	0.39	0.90	0.46	18.9†
UAD∗ (Guo et al., 2024)	Det	0.39	0.81	1.50	0.90	0.01	0.12	0.43	0.19	7.2†
SSR (Ours)	None	**0.24**	**0.65**	**1.36**	**0.75**	**0.00**	**0.10**	**0.36**	**0.15**	**19.6**
ST-P3‡ (Hu et al., 2022)	Det & Map & Depth	1.33	2.11	2.90	2.11	0.23	0.62	1.27	0.71	1.6
UniAD∗‡ (Hu et al., 2023)	Det&Track&Map&Motion&Occ	0.44	0.67	0.96	0.69	0.04	0.08	0.23	0.12	1.8†
VAD-Tiny‡ (Jiang et al., 2023)	Det & Map & Motion	0.46	0.76	1.12	0.78	0.21	0.35	0.58	0.38	16.8
VAD-Base‡ (Jiang et al., 2023)	Det & Map & Motion	0.41	0.70	1.05	0.72	0.07	0.17	0.41	0.22	4.5
BEV-Planner‡ (Li et al., 2024c)	None	0.28	0.42	0.68	0.46	0.04	0.37	1.07	0.49	-
PARA-Drive‡ (Weng et al., 2024)	Det&Track&Map&Motion&Occ	0.25	0.46	0.74	0.48	0.14	0.23	0.39	0.25	5.0
LAW‡ (Li et al., 2024b)	None	0.26	0.57	1.01	0.61	0.14	0.21	0.54	0.30	19.5
GenAD‡ (Zheng et al., 2024b)	Det & Map & Motion	0.28	0.49	0.78	0.52	0.08	0.14	0.34	0.19	6.7
SparseDrive‡ (Sun et al., 2024)	Det & Track & Map & Motion	0.29	0.58	0.96	0.61	0.01	0.05	0.18	0.08	9.0
UAD∗‡ (Guo et al., 2024)	Det	0.28	0.41	0.65	0.45	0.01	**0.03**	0.14	0.06	7.2†
SSR‡ (Ours)	None	**0.18**	**0.36**	**0.63**	**0.39**	**0.01**	0.04	**0.12**	**0.06**	**19.6**

The ego status was not utilized in the planning module.

⋄:Lidar-based methods.

∗:Backbone with ResNet-101,while others use ResNet-50 or similar.

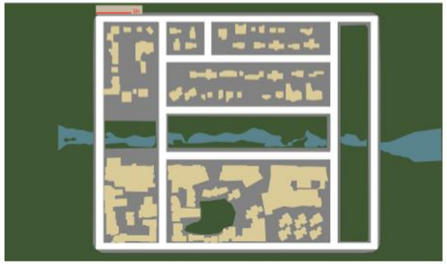
†:FPS measure d on an NVIDIA A100 GPU, while others were tested on an NVIDIA RTX3090.

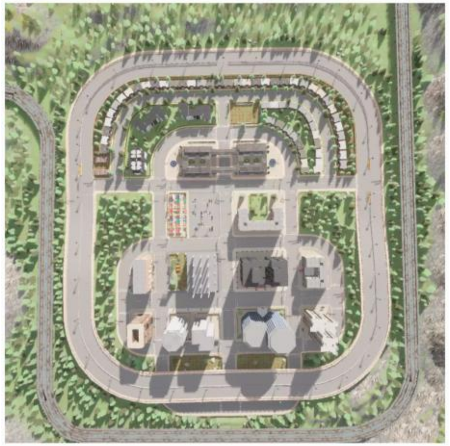
‡:AVG metric protocol as same as VAD.

4.Experiments – Comparison of SOTA on the CARLA dataset (Closed-Loop)

Table 2: **Performance on Town05 Long benchmark.**

Method	Modality	**DS↑**	RC↑	IS↑
CILRS (Codevilla et al., 2019)	C	7.8	10.3	0.75
LBC (Chen et al., 2020)	C	12.3	31.9	0.66
Transfuser (Prakash et al., 2021)	C+L	31.0	47.5	0.77
Roach (Zhang et al., 2021)	C	41.6	**96.4**	0.43
ST-P3 (Hu et al., 2022)	C	11.5	83.2	-
TCP (Wu et al., 2022)	C	57.2	80.4	0.73
VAD-Base (Jiang et al., 2023)	C	30.3	75.2	-
ThinkTwice (Jia et al., 2023b)	C+L	65.0	95.5	0.69
DriveAdapter(Jia et al., 2023a)	C+L	65.9	94.4	0.72
SSR (Ours)	C	**78.9**	95.5	**0.83**


Town 1 Map


Town05

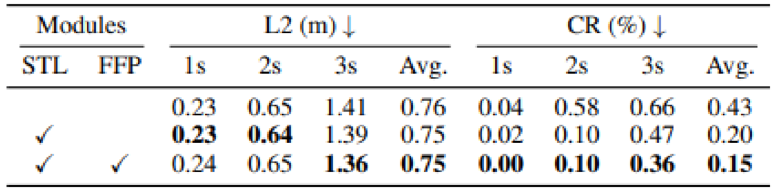
The training data has **no overlap with Town05 Long** benchmark.

CARLA has been built for flexibility and realism in the rendering and physics simulation. It is implemented as an open-source layer over **Unreal Engine 4 (UE4)**, enabling future extensions by the community. Town 1 with a total of 29 km of drivable roads, used for training, and Town 2 with 14 km of drivable roads, used for testing.

Dosovitskiy, Alexey, et al. "CARLA: An open urban driving simulator." Conference on robot learning. PMLR, 2017.

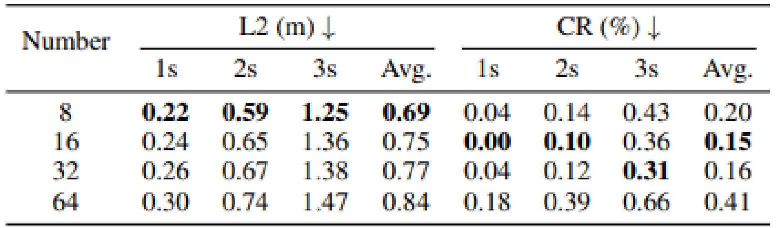
4.Experiments – Comparison of SOTA on the nuScenes dataset (Closed-Loop)

Table 3: **Component-wise Ablation.**



Modules		L2 (m) ↓				CR (%) ↓			
STL	FFP	1s	2s	3s	Avg.	1s	2s	3s	Avg.
		0.23	0.65	1.41	0.76	0.04	0.58	0.66	0.43
✓		**0.23**	**0.64**	1.39	0.75	0.02	0.10	0.47	0.20
✓	✓	0.24	0.65	**1.36**	**0.75**	**0.00**	**0.10**	**0.36**	**0.15**

Table 4: **Number of Scene Queries.**



Number	L2 (m) ↓				CR (%) ↓			
	1s	2s	3s	Avg.	1s	2s	3s	Avg.
8	**0.22**	**0.59**	**1.25**	**0.69**	0.04	0.14	0.43	0.20
16	0.24	0.65	1.36	0.75	**0.00**	**0.10**	0.36	**0.15**
32	0.26	0.67	1.38	0.77	0.04	0.12	**0.31**	0.16
64	0.30	0.74	1.47	0.84	0.18	0.39	0.66	0.41

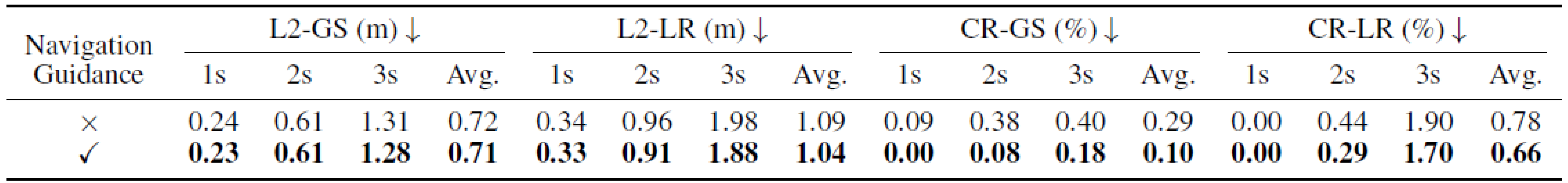
STL's ability to effectively distill critical information from the dense scene data, thereby minimizing the impact of irrelevant features and reducing computational redundancy.

Future Feature Predictor's role in enhancing SSR's comprehension of scene dynamics, contributing to more safe trajectory planning and overall performance gains.

Select 16 queries as the default setting in SSR to balance minimizing L2 error and reducing the collision rate.

Tab. 5 shows that **navigation guidance improves planning results** across all cases.

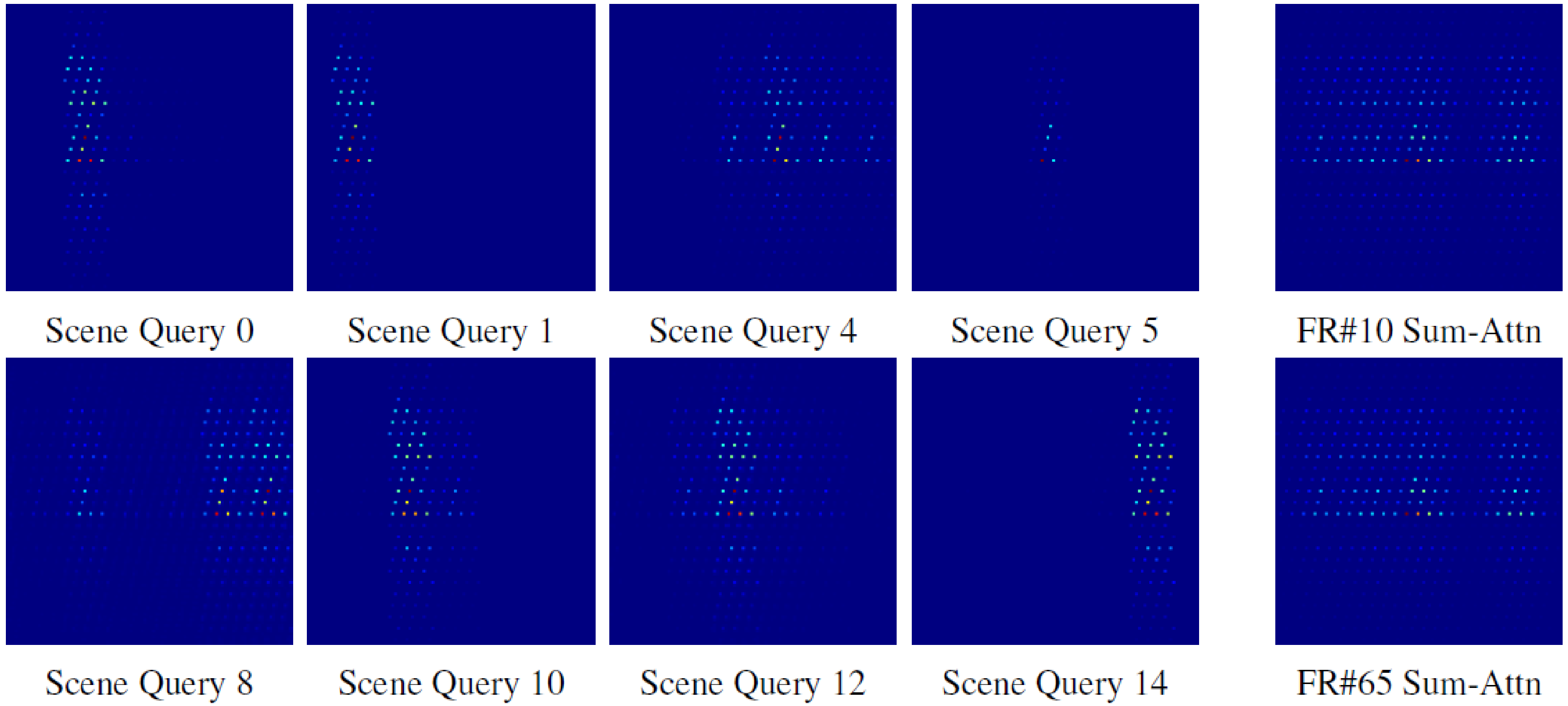
Table 5: **Ablation of navigation guidance.** GS means *go straight* and LR denotes *turn left / right.*



Navigation Guidance	L2-GS (m) ↓				L2-LR (m) ↓				CR-GS (%) ↓				CR-LR (%) ↓			
	1s	2s	3s	Avg.	1s	2s	3s	Avg.	1s	2s	3s	Avg.	1s	2s	3s	Avg.
×	0.24	0.61	1.31	0.72	0.34	0.96	1.98	1.09	0.09	0.38	0.40	0.29	0.00	0.44	1.90	0.78
✓	**0.23**	**0.61**	**1.28**	**0.71**	**0.33**	**0.91**	**1.88**	**1.04**	**0.00**	**0.08**	**0.18**	**0.10**	**0.00**	**0.29**	**1.70**	**0.66**

4.Experiments – Number of scene queries

Visualize 8 out of the 16 BEV square attention maps $\varpi(B_t^{navi})$ from the STL module. The results reveal that **each query focuses uniformly on a distinct region of the BEV space**, with different queries attending to different areas. **the sum-attention map surprisingly covers the entire scene** in a balanced manner



4.Experiments – Visualization of Scene Queries

Fig. 7(a), the activation positions primarily focus on the **overtaked vehicle** and the left rear area, anticipating potential risks. Fig. 7(b), the scene queries are more dispersed, with attention directed towards a front-right vehicle, potentially **anticipating a cut-in**. Fig. 7(c)), the scene queries not only activate around **the right rear vehicle but also pay attention to the left crosswalk**, where pedestrians might appear

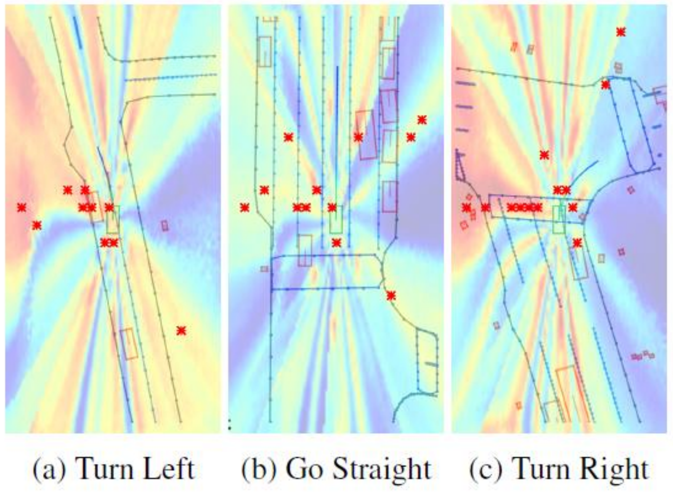


Figure 7: **Visualization of Scenes Queries in Different Scenarios.** The central green box represents ego vehicle. The red boxes indicate the ground truth object, while the dotted lines denote ground truth map. The red star marker is the most activated position of each scene query.

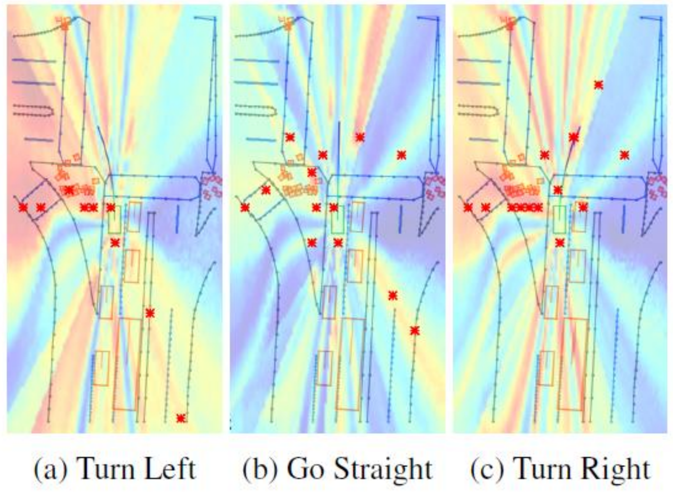


Figure 8: **Visualization of Scene Queries for Different Navigation Commands in Same Scene.** Different navigation commands are input to the SSR within the same scene to investigate their impact on scene queries. The original command is *go straight*.

4.Experiments – Visualization

A qualitative result of SSR on planning trajectories, demonstrating strong alignment with the ground truth compared to VAD-Base.

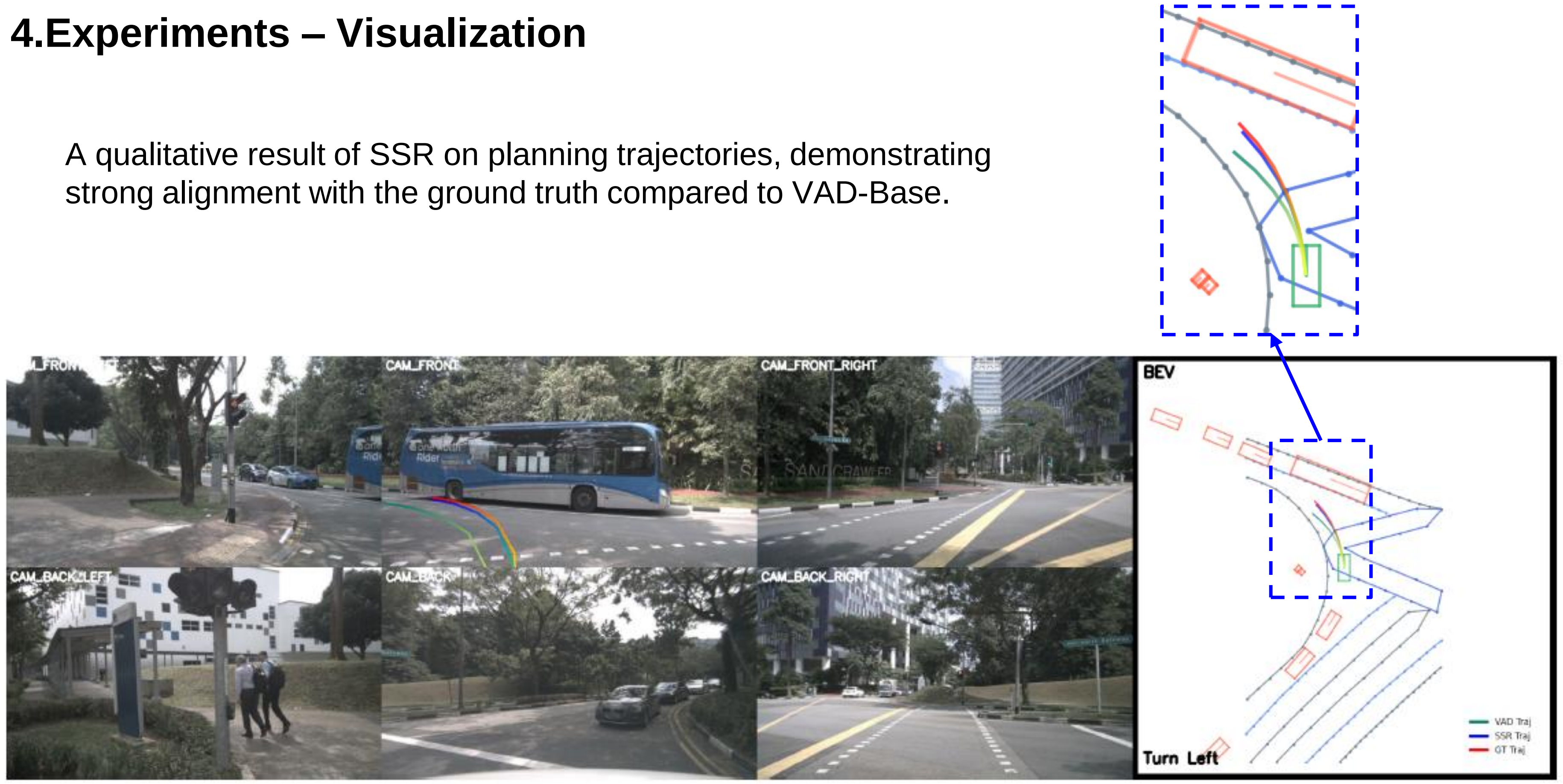


Figure 9: **Visualization of Planning Results.** The perception results are rendered from annotations.

4.Experiments – Latency Analysis

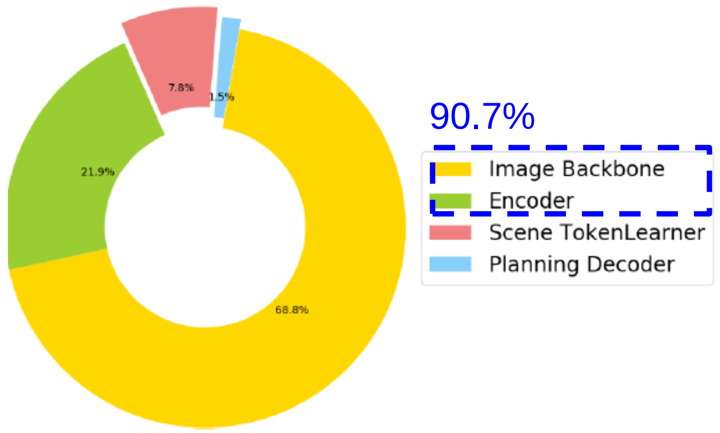
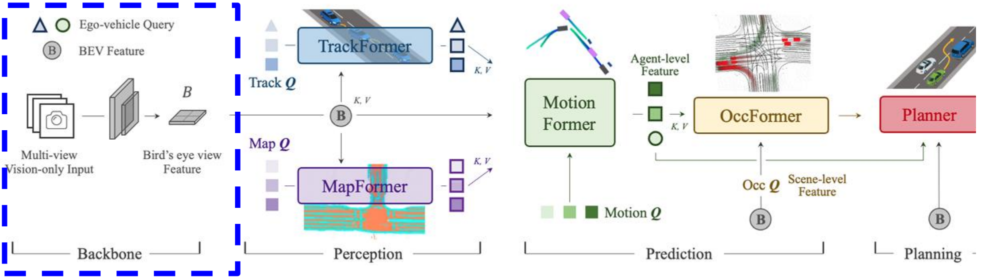
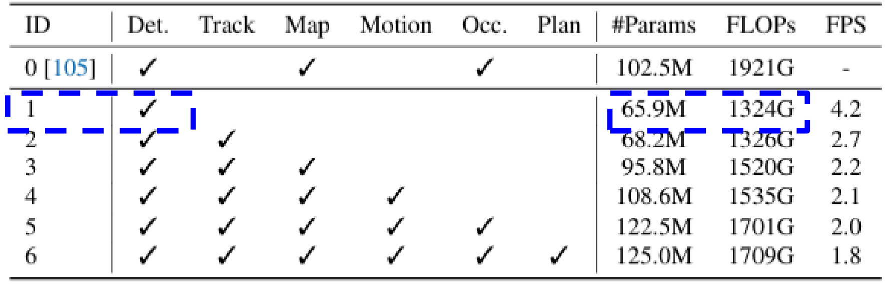


Figure 13: **Latency Analysis.**

NVIDIA GeForce RTX 3090 GPU with a batch size of 1. The **image backbone and encoder, responsible for generating dense BEV features, contribute to 90.7%** of the total latency. In contrast, our proposed **Scenes TokenLearner incurs only 7.8% of the latency**, highlighting its efficiency in extracting useful information from massive dense BEV feature. The planning **decoder, which interacts way point queries** with the scene queries and output final planning trajectory, adds just **1.5% to the latency**, as SSR only utilizes 16 tokens to represent the scenes.



UniAD (77%)



ID	Det.	Track	Map	Motion	Occ.	Plan	#Params	FLOPs	FPS
0 [105]	✓		✓		✓		102.5M	1921G	-
1	✓						65.9M	1324G	4.2
2	✓	✓					68.2M	1326G	2.7
3	✓	✓	✓				95.8M	1520G	2.2
4	✓	✓	✓	✓			108.6M	1535G	2.1
5	✓	✓	✓	✓	✓		122.5M	1701G	2.0
6	✓	✓	✓	✓	✓	✓	125.0M	1709G	1.8

Table 13. **Computational complexity and runtime** with different modules incorporated. ID. 1 is similar to original BEVFormer [55], and ID. 0 (BEVerse-Tiny) [105] is an MTL framework.

5.Conclusion & Limitation

(+) Utilizes learned **sparse query representations(16 tokens) guided by navigation commands**, significantly reducing computational costs by adaptively focusing on essential parts of scenes.

(+) Introduces a **future feature predictor for self-supervision** on dynamic scene changes, eliminating the need for costly perception tasks supervision.

(+) Achieves **state-of-the-art performance on both open-loop and closed-loop** experiments, establishing a new benchmark for real-time E2EAD.

(+) SSR **eliminates all perception tasks**, achieving remarkable performance in both accuracy and efficiency.

The **five reviewers agree** that the paper has sufficient novelty.
(-) **Fixed number of queries** may limit the models ability to handle many other agents/objects.
(-) Can the reliance on **adaptive perception be explained**?
(-) The proposed architecture borrows the idea of a **world model** by predicting future BEV features.
(-) Could we use a **more cost-effective operator**? (e.g., Shift Operation)
(-) How about understanding and incorporating **physics** (gravity, velocity, and environment)?

https://openreview.net/forum?id=Vv76fCYffN

Thanks

Any Questions?

You can send mail to
Susang Kim(healess1@gmail.com)