Valeo

valeo.ai

Driving with ViT registers

Trajectory planning and scoring

Alexandre Boulch

Workshop on 3D Perception and Navigation 2026

End-to-end driving from cameras

Navsim scene (nuPlan)

Inputs

Sensors

- Cameras
  (at current time and optionally previous)

Ego status

- Speed
- Acceleration

High level command
(e.g., “Turn” right)

Outputs

- Trajectory for the next 4 seconds (8 timesteps)

Previous work

Modular end-to-end trained methods

ViP3D E2E Till Forecast. **UniAD** Among first E2E Planning

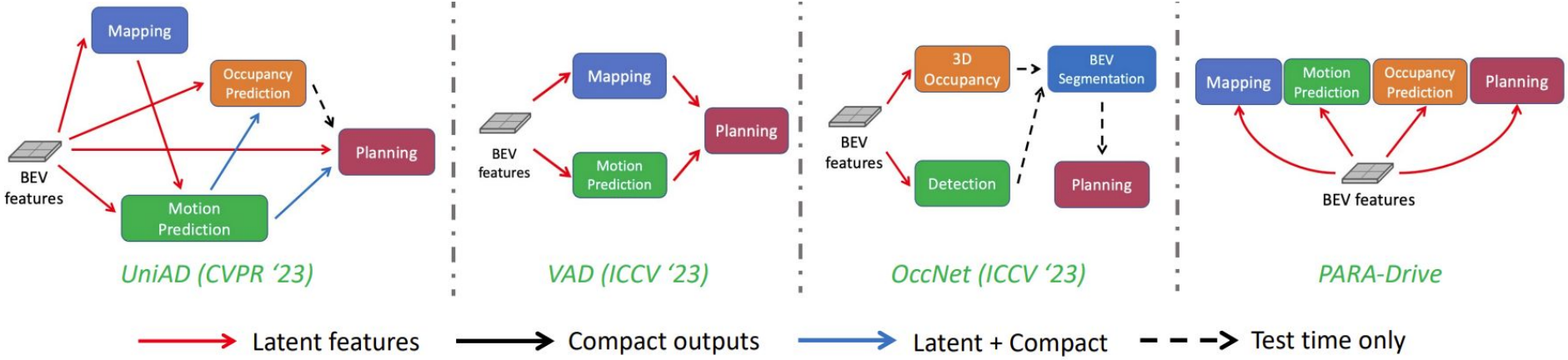
FusionAD (Lidar-Cam)
VAD (vectorized Rep.)
ParaDrive (seq to paral. pipeline)
...

Modular: multiple perception sub-tasks

- Tracking, mapping, forecasting, occupancy planning

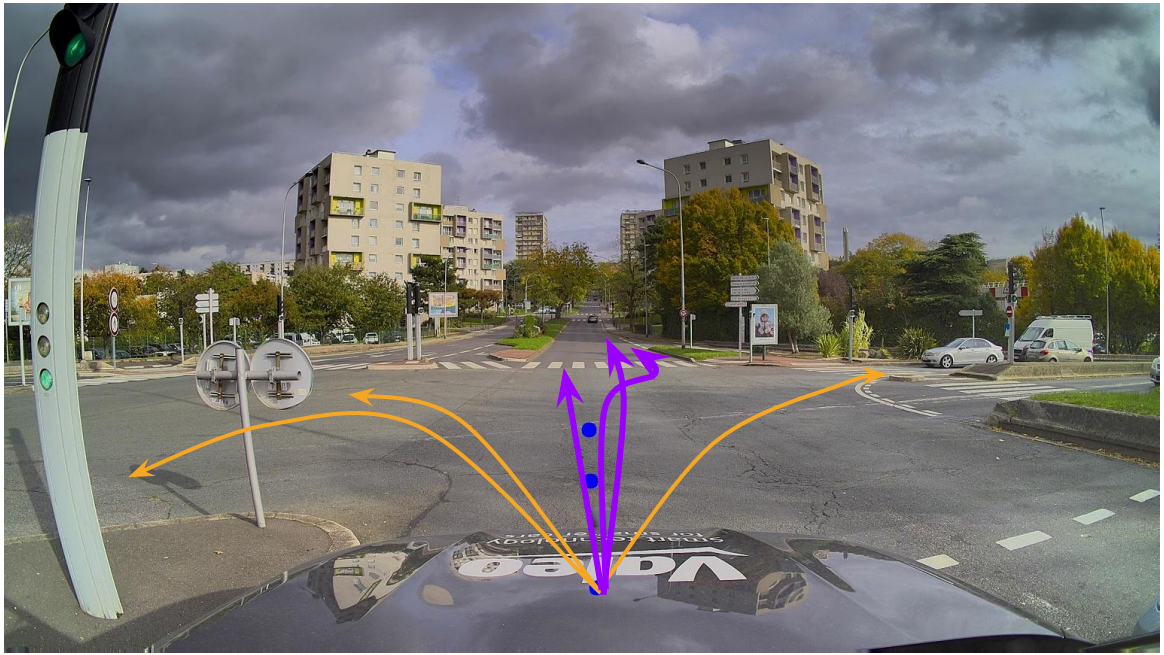
E2E trained: jointly trained via queries

- All tasks are learnt jointly and grad flow from planning objective to image backbone.



Ambiguity in End-to-End Driving

What is the correct driving behavior in this situation?



Previous work

Scoring-based methods with fixed vocabulary

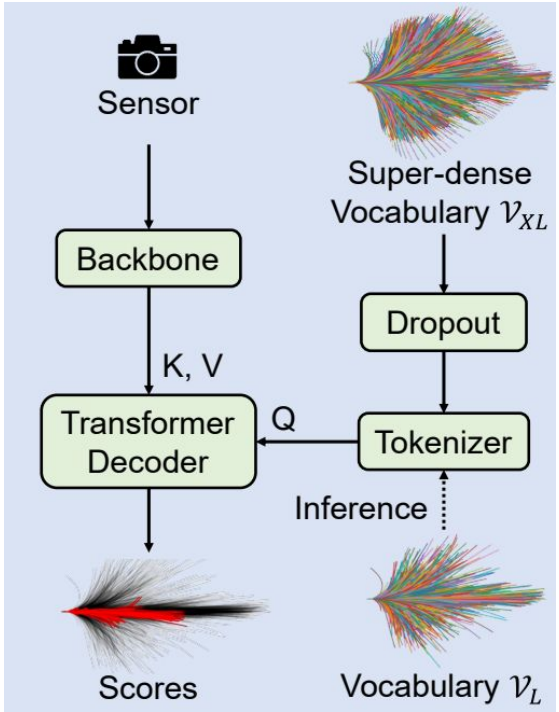
MultiPath □ VADv2 □ Hydra-MDP □ DriveSuprim □GTRS □ZTRS ...

Pre-computed vocabulary

- from traj., clustering of the dataset
- Optionally refined with a light regression head

Scoring head

- A scoring head interacts with sensor information
- predicts scores for each vocab.



Hydra-MDP

Previous work

Trajectory prediction and scoring

iPad DETR-like regression–based queries + PDM scorer

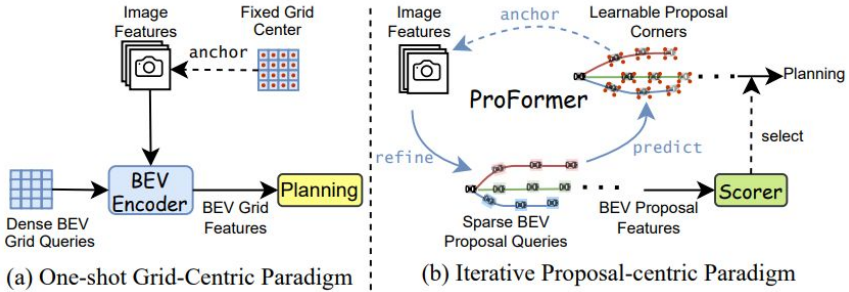
DiffusionDrive diffusion-based queries + classifier

Sparse query-based traj. prediction

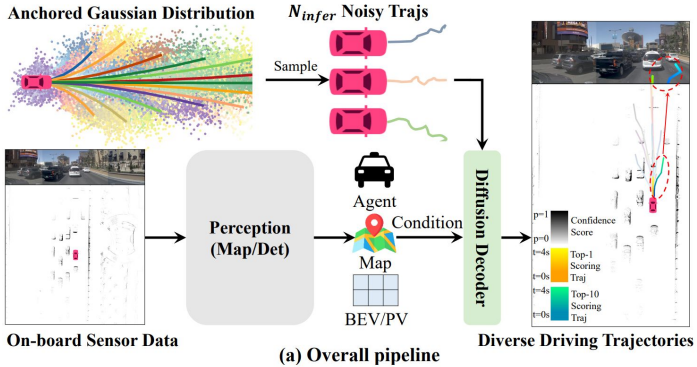
- Fixed-number of traj., queries
- Cross attention with sensor inputs

Scoring head

- A scoring head interacts with sensor information
- predicts (PDM) scores for each predicted traj.



iPad



DiffusionDrive

DrivoR: Driving on Registers

CVPR 2026

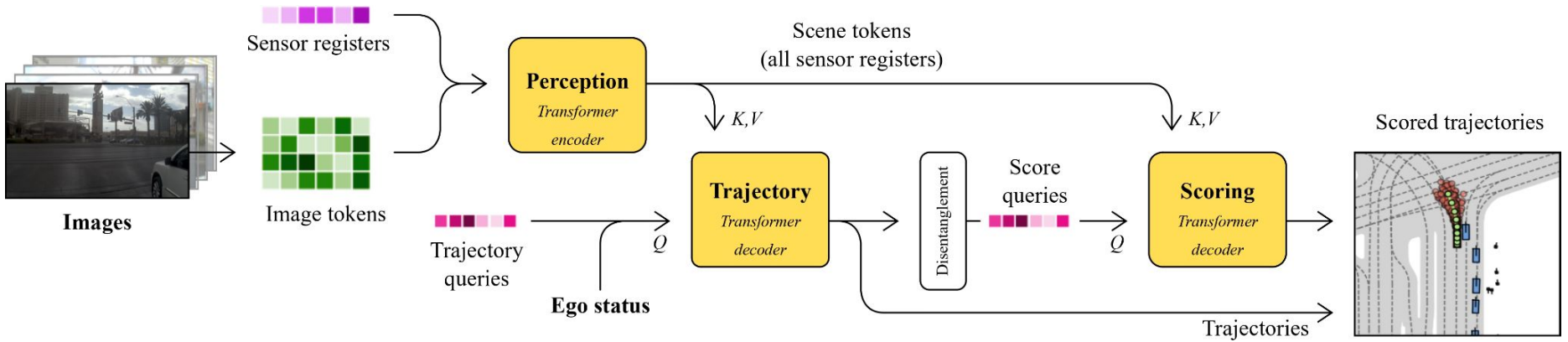
Ellington Kirby*, Alexandre Boulch*, Yihong Xu*, Yuan Yin, Gilles Puy, Eloi Zablocki, Andrei Bursuc, Spyros Gidaris, Renaud Marlet, Florent Bartoccioni, Anh-Quan Cao, Nermin Samet, Tuan-Hung VU, Matthieu Cord

How to reduce the amount of camera information?

How to reduce architecture complexity? (any hardware)

How to estimate the quality of a trajectory?

Compress sensor data using “Scene Tokens” (registers)



Perception

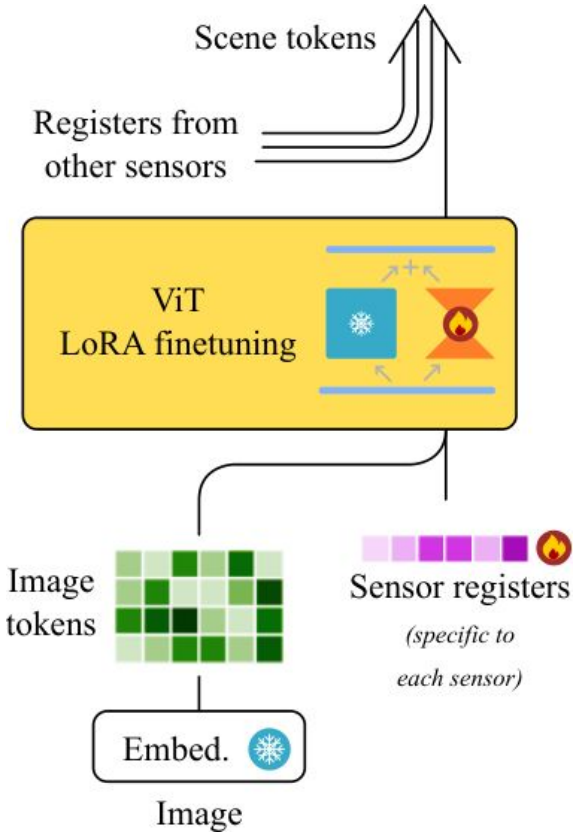
Use efficient pretrained models

DINOv2 with LoRA finetuning

Registers: perceptual compression

Use additional registers (trained)
Keep only the registers (<< #image tokens)

250x Compresion: 4x4096 → 4x16 tokens



How to reduce the amount of camera information?

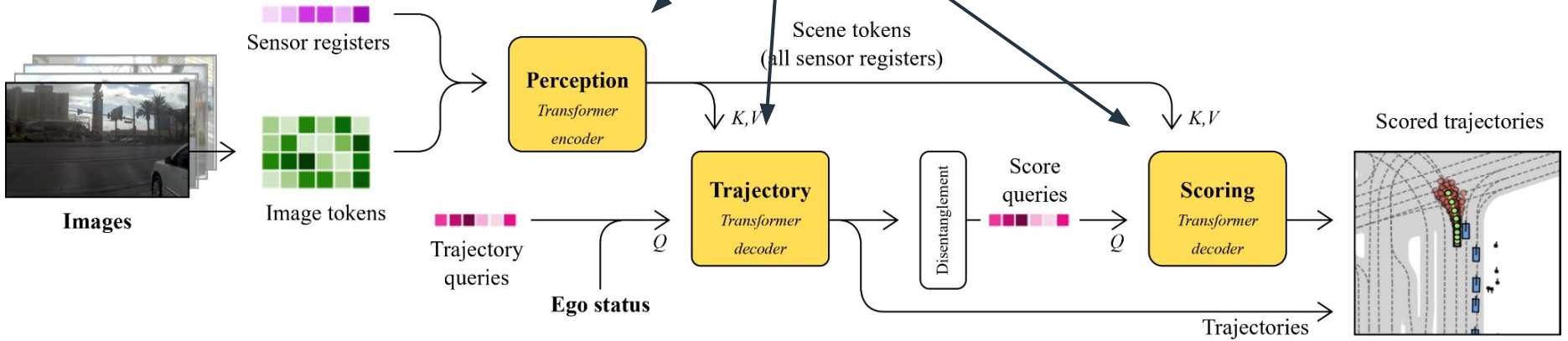
Compress sensor data using “Scene Tokens” (registers)

How to reduce architecture complexity? (any hardware)

No BEV projection

Transformer only

How to estimate the quality of a trajectory?



How to reduce the amount of camera information?

Compress sensor data using "Scene Tokens" (registers)

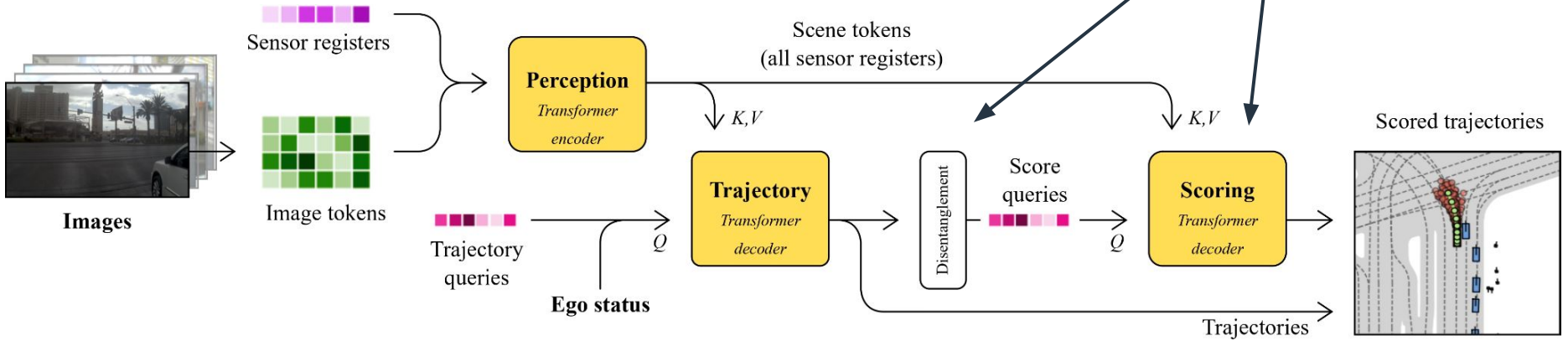
How to reduce architecture complexity? (any hardware)

No BEV projection

Transformer only

How to estimate the quality of a trajectory?

Traj. / scores disentanglement + subscores



Scorer

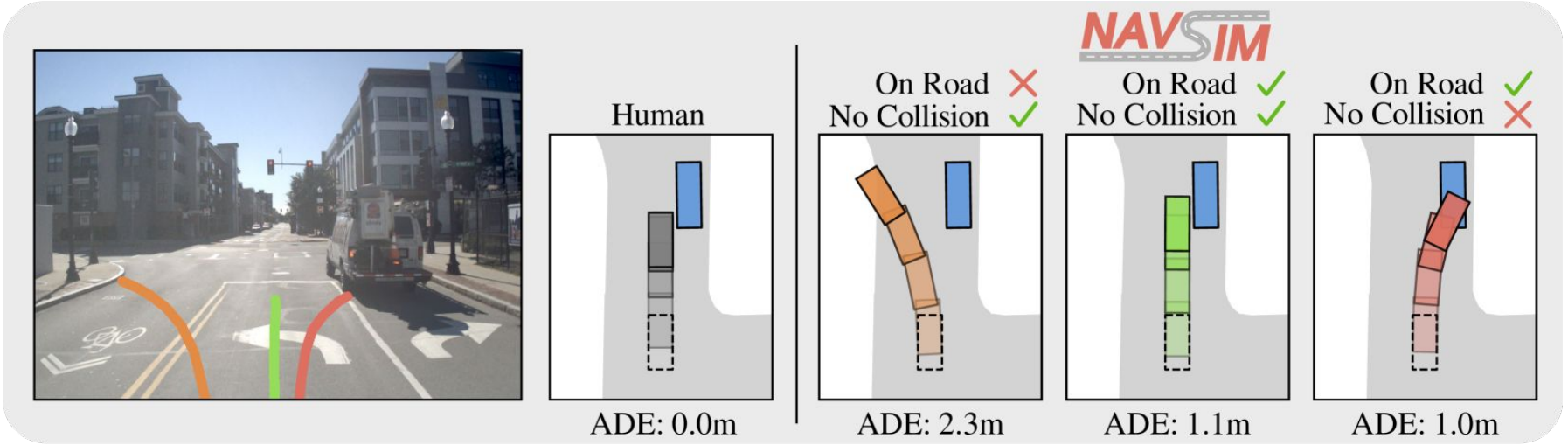
How to score trajectories? **Alternative to pure Imitation Learning**

Safety & Compliance

- No-at-fault collision (NC)
- Drivable area compliance (DAC)

Progress & Experience

- Time-to-collision (TTC)
- Ego Progress (EP)
- Comfort (C)



Scorer

How to score trajectories? **Alternative to pure Imitation Learning**

Safety & Compliance	Progress & Experience
• No-at-fault collision (NC) • Drivable area compliance (DAC)	• Time-to-collision (TTC) • Ego Progress (EP) • Comfort (C)

$$\text{PDMS} = \underbrace{\left(\prod_{m \in \{\text{NC,DAC}\}} \text{score}_m \right)}_{\text{penalties}} \times \underbrace{\left(\frac{\sum_{w \in \{\text{EP,TTC,C}\}} \text{weight}_w \times \text{score}_w}{\sum_{w \in \{\text{EP,TTC,C}\}} \text{weight}_w} \right)}_{\text{weighted average}}$$

Training the scorer

How to score trajectories? **Alternative to pure Imitation Learning**

Safety & Compliance

- No-at-fault collision (NC)
- Drivable area compliance (DAC)

Progress & Experience

- Time-to-collision (TTC)
- Ego Progress (EP)
- Comfort (C)

Subscore index c, and trajectories index i

$$\mathcal{L}_{\text{traj}} = \min_i \|\tau_i - \hat{\tau}\|_1 \qquad \mathcal{L}_{\text{score}} = \sum_c \lambda_c \sum_i \text{BCE}\left(\mathcal{G}_{\theta_c}(\tau_i), \mathcal{G}_c(\tau_i)\right)$$

$$\mathcal{L} = \mathcal{L}_{\text{traj}} + \lambda_s \mathcal{L}_{\text{score}}$$

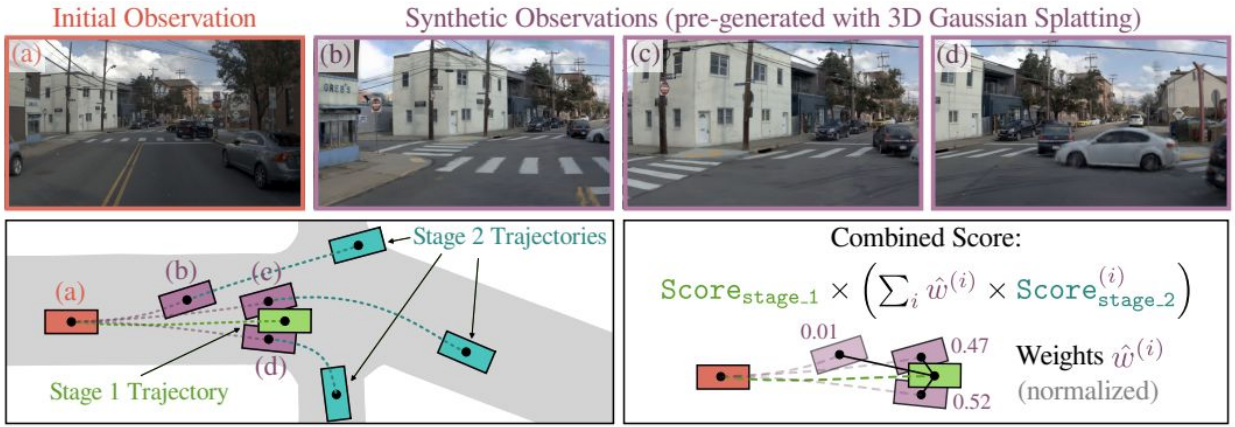
Pseudo closed-loop: NAVSIM-v2

Safety & Compliance

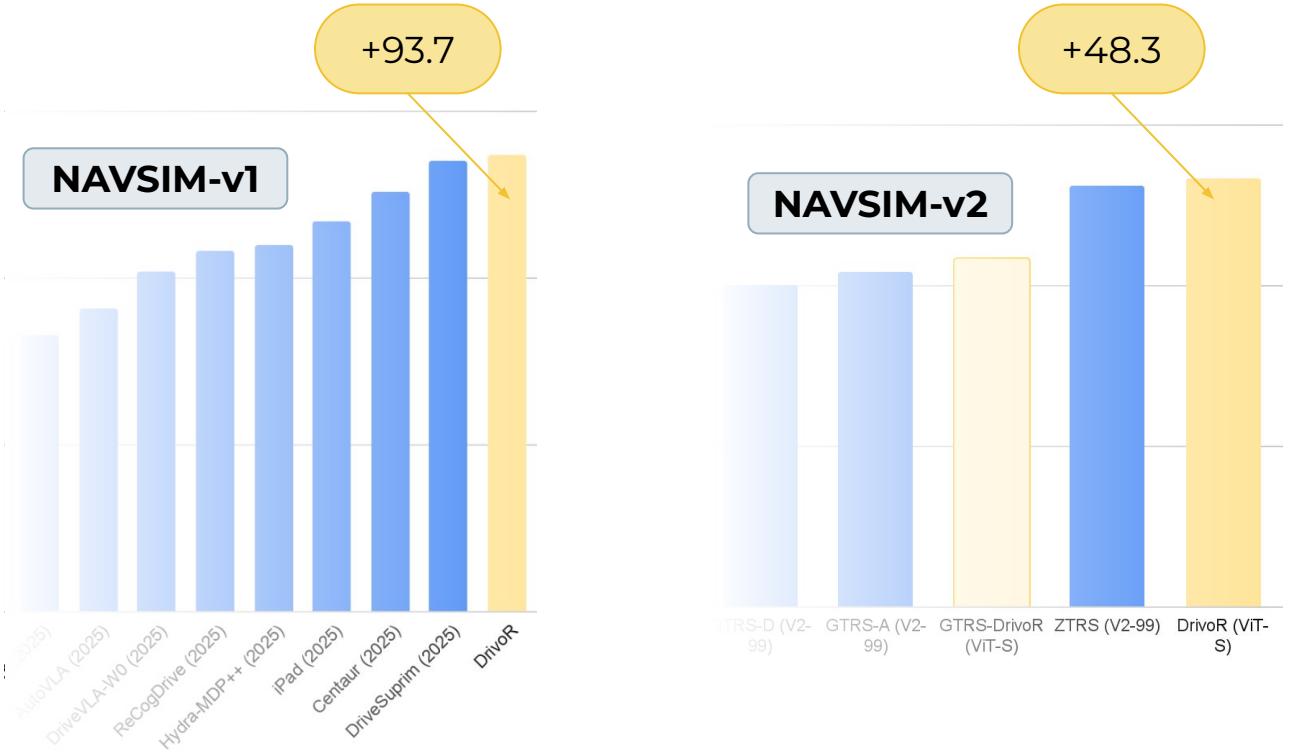
- No-at-fault collision
- Drivable area compliance
- Driving Direction Compl. (DDC)
- Traffic Light Compl. (TLC)

Progress & Experience

- Time-to-collision (TTC)
- Ego Progress (EP)
- Lane Keeping (LK)
- History Comfort (HC)
- Extended Comfort (EC)



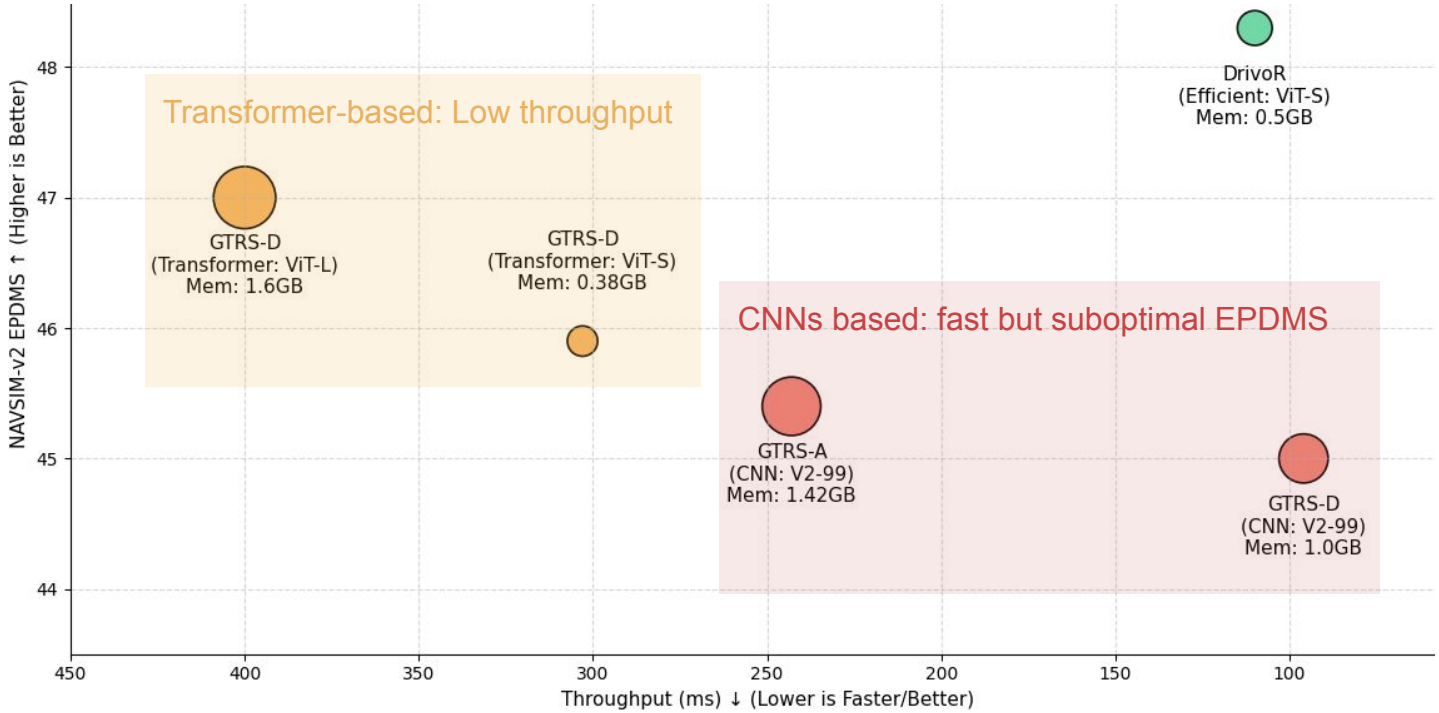
Benchmarks: a simplified pipeline competitive with previous work



Impact of image encoders - NAVSIM-v1

Method	NC	DAC	TTC	Comf.	EP	**PDMS**
Human driver	100	100	100	99.9	87.5	94.8
DrivoR (trainval, DINOv3)	99.1	98.2	96.7	100	88.0	92.5
DrivoR (train, DINOv2)	98.9	98.3	96.2	100	89.1	93.1
DrivoR (trainval, DINOv2)	99.0	98.9	96.7	100	90.0	93.7
DrivoR (trainval, Franca)	98.9	98.9	96.2	100	91.2	93.9

Fully open source valeo.ai foundation model
https://github.com/valeoai/Franca


Transformer-based: Low throughput
CNNs based: fast but suboptimal EPDMS
DrivoR
(Efficient: ViT-S)
Mem: 0.5GB
GTRS-D
(Transformer: ViT-L)
Mem: 1.6GB
GTRS-D
(Transformer: ViT-S)
Mem: 0.38GB
GTRS-A
(CNN: V2-99)
Mem: 1.42GB
GTRS-D
(CNN: V2-99)
Mem: 1.0GB
NAVSIM-v2 EPDMS ↑ (Higher is Better)
Throughput (ms) ↓ (Lower is Faster/Better)
48
47
46
45
44
450
400
350
300
250
200
150
100

Scaling DrivoR

SimScale Training Data

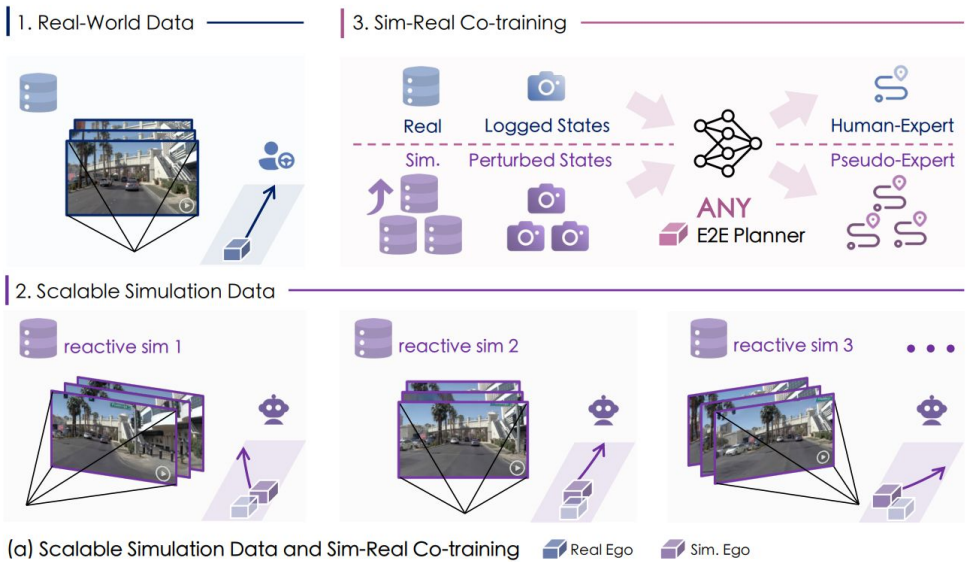
SimScale Framework

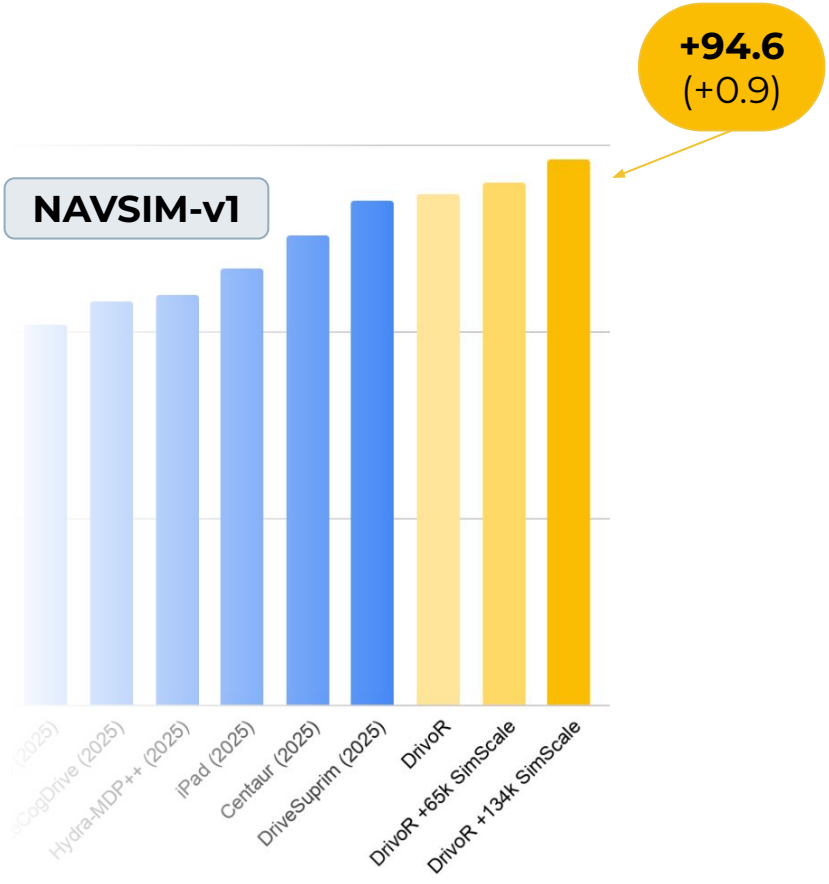
- Augment real-world images using 3DGS rendering
- Pseudo annotations with PDM/Recovery-based trajectories

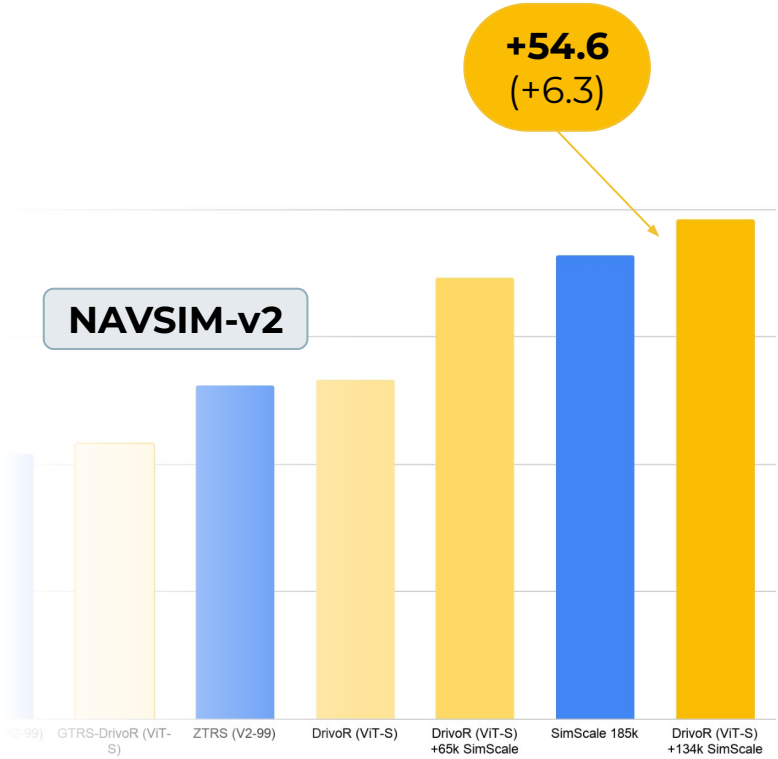
Scalable Design Architecture

Efficient transformer-based design enables high scalability.

- Training DrivoR with extra 65k and 134k SimScale samples



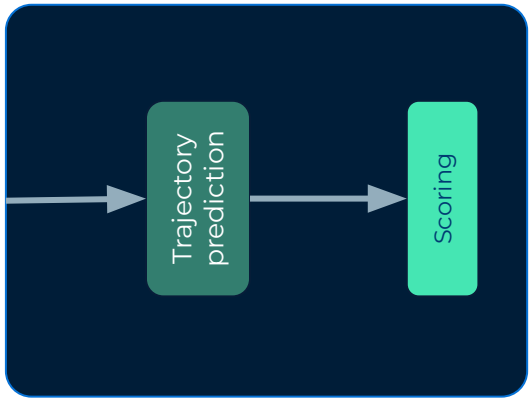

+94.6
(+0.9)
NAVSIM-v1
Hydra-MDP++ (2025)
iPad (2025)
Centaur (2025)
DriveSuprim (2025)
DrivoR
DrivoR +65k SimScale
DrivoR +134k SimScale


+54.6
(+6.3)
NAVSIM-v2
GTRS-DrivoR (ViT-S)
ZTRS (V2-99)
DrivoR (ViT-S)
DrivoR (ViT-S) +65k SimScale
SimScale 185k
DrivoR (ViT-S) +134k SimScale

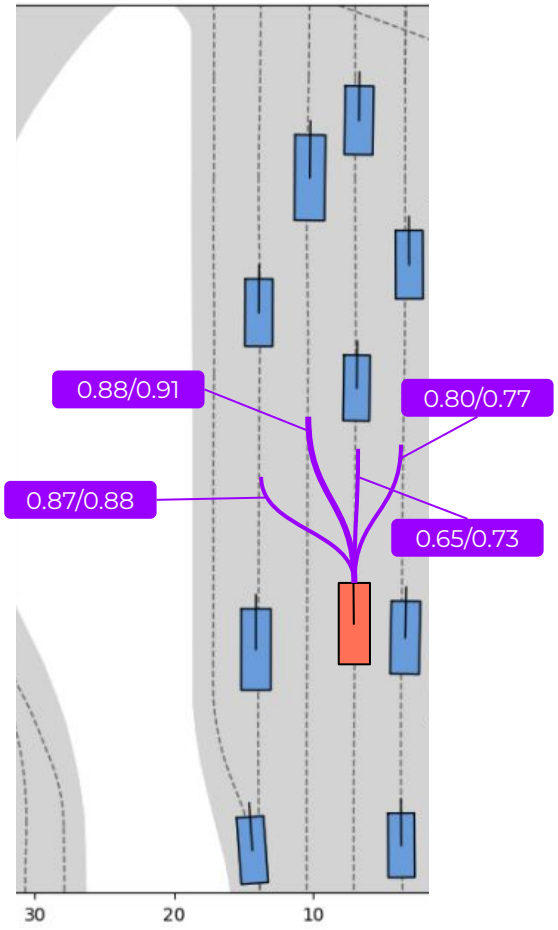
TOAD: Test-time Optimization for Autonomous Driving

Yihong Xu, Éloi Zablocki, Yuan Yin, Elias Ramzi, Ellington Kirby, Alexandre Boulch, Matthieu Cord

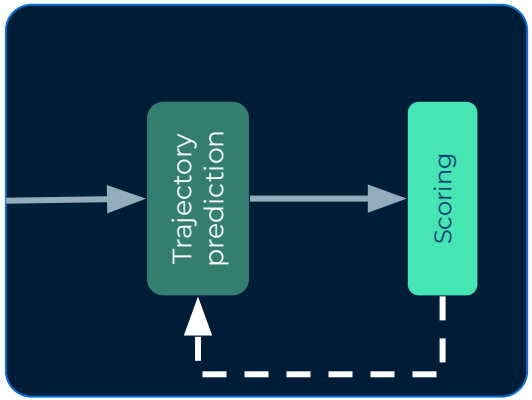
DrivoR



The scorer does not help the trajectory prediction.

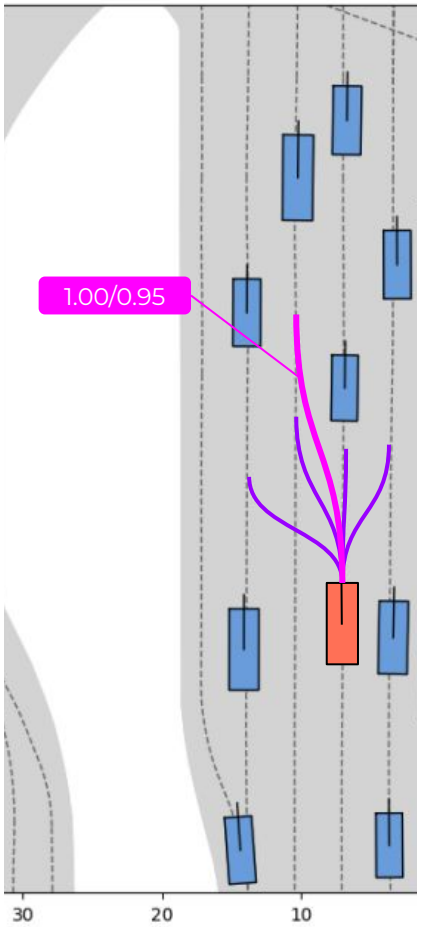


Improve the trajectory via scoring

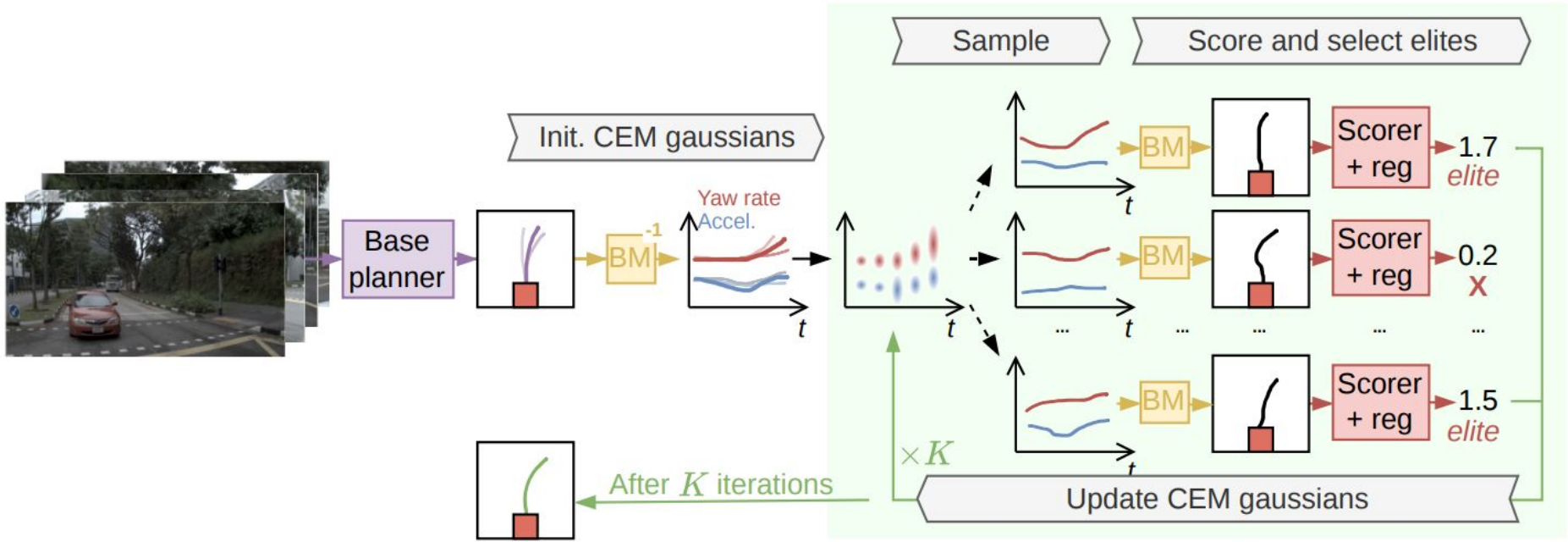


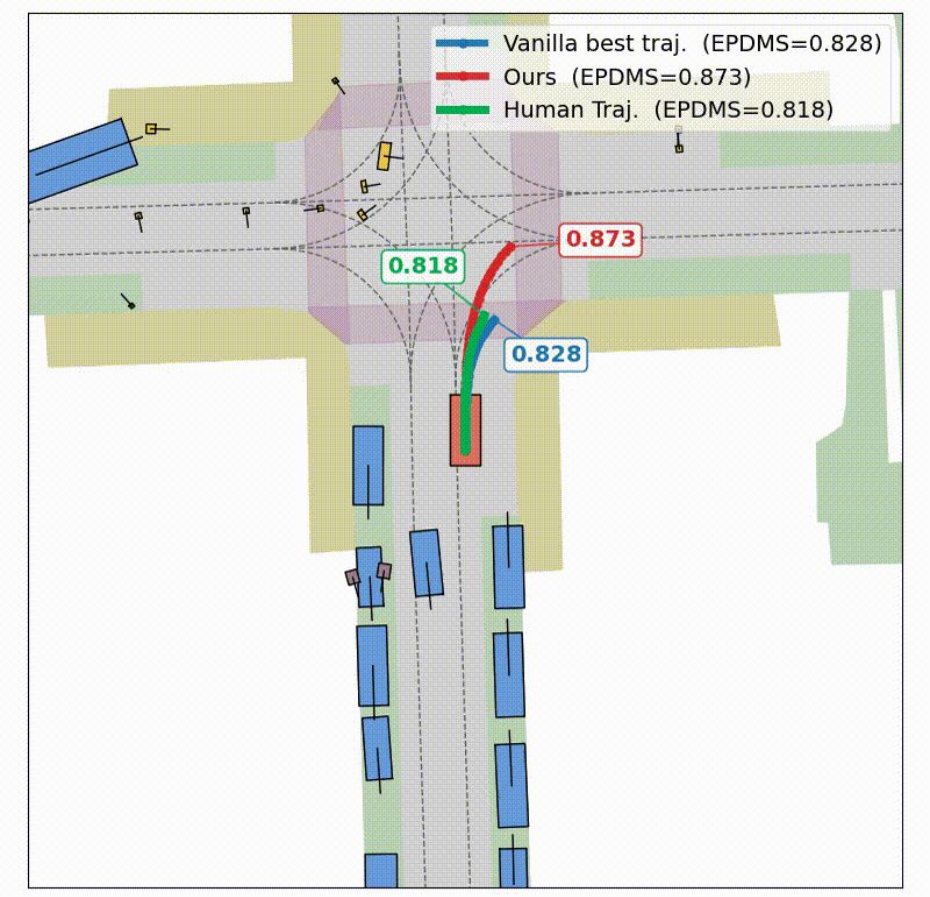
Search in the trajectory space with scorer guidance.

Preferentially as a post processing (no additional training cost)

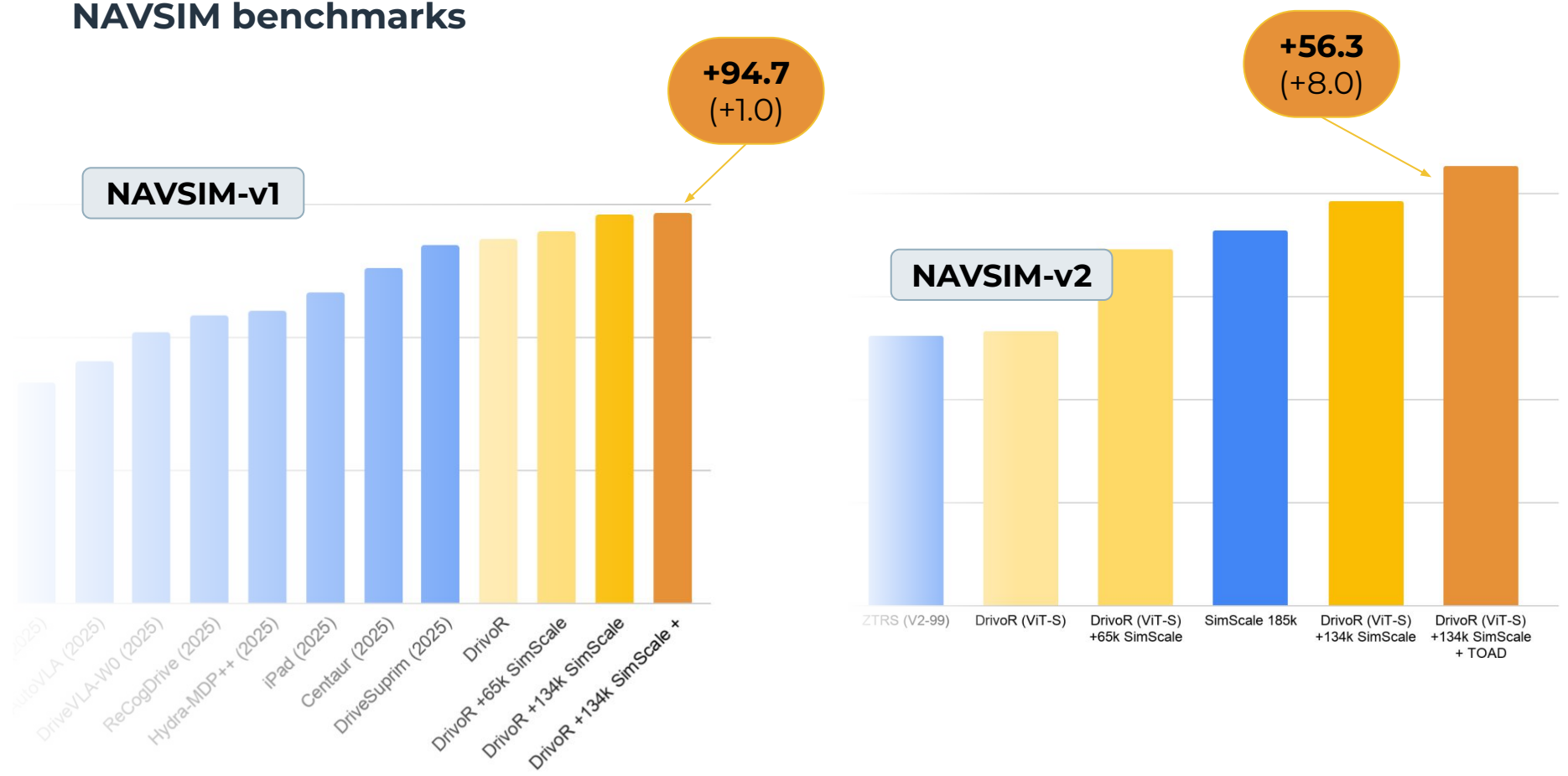


Cross-entropy method for the trajectories




Vanilla best traj. (EPDMS=0.828)
Ours (EPDMS=0.873)
Human Traj. (EPDMS=0.818)
0.873
0.818
0.828

NAVSIM benchmarks



- Smoothing the selected trajectory
- Rescoring with external scorer
- Search with GTRS / Sparse Drive
  → essentially classification methods
  → bad generalization out of vocabulary
- Search with DrivoR
  → trained with dynamic vocabulary
  → better generalization

Method	EPDMS (↑)
iPad [10]	34.7
+ Smoothing (BM)	35.0 (↑ 0.8%)
+ Re-scoring w/ DrivoR	45.6 (↑31.5%)
+ Re-scoring w/ GTRS	34.5 (↓ 0.6%)
+ Re-scoring w/ SparseDriveV2	30.3 (↓12.5%)
+ Search w/ GTRS scorer	23.9 (↓31.1%)
+ Search w/ SparseDriveV2 scorer	34.6 (↓ 0.4%)
+ TOAD (search w/ DrivoR scorer)	49.8 (↑43.6%)
Hydra-MDP [4]	40.9
+ Smoothing (BM)	38.8 (↓ 5.1%)
+ Re-scoring w/ DrivoR	37.3 (↓ 8.8%)
+ Re-scoring w/ GTRS	40.3 (↓ 1.5%)
+ Re-scoring w/ SparseDriveV2	9.5 (↓76.9%)
+ Search w/ GTRS scorer	38.1 (↓ 6.9%)
+ Search w/ SparseDriveV2 scorer	29.6 (↓27.7%)
+ TOAD (search w/ DrivoR scorer)	49.7 (↑21.6%)

Generalization to other approaches

TOAD provide consistent improvement for all tested methods.
→ raising the scores to 49+ on NAVSIM-v2

Method	EPDMS (↑)
PDM-Closed [14]	56.6
iPad [10]	34.7
+ TOAD	49.8 (↑43.6%)
RAP-DINO [9]	39.6
+ TOAD	49.0 (↑23.9%)
Hydra-MDP [4]	40.9
+ TOAD	49.7 (↑21.6%)
GTRS [5]	45.4
+ TOAD	51.7 (↑13.8%)
ZTRS [6]	48.1
+ TOAD	49.2 (↑ 2.3%)
DrivoR [8]	54.6
+ TOAD	**56.3 (↑ 3.1%)**

Random initialization (iPad anchors) → 42.7
→ the scorer must be in domain
→ random trajectories are out-of-domain

TOAD

Lightweight computation

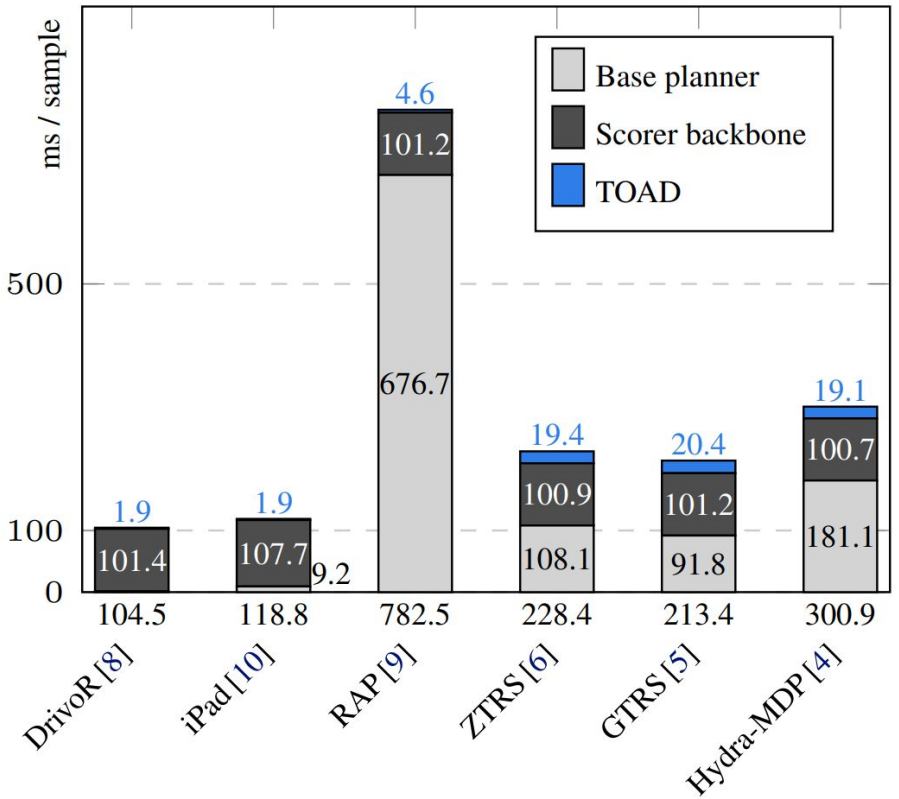
Almost free if sharing the perception backbone.

Test time only

No additional training time

Generalization

All tested methods benefit from TOAD



Driving (and scoring) with registers

ViT backbone, SotA perception

Leverage efficient pretrained models (DINOv2).

Transformer only

Scalable with vanilla transformers. No custom CUDA kernels required.

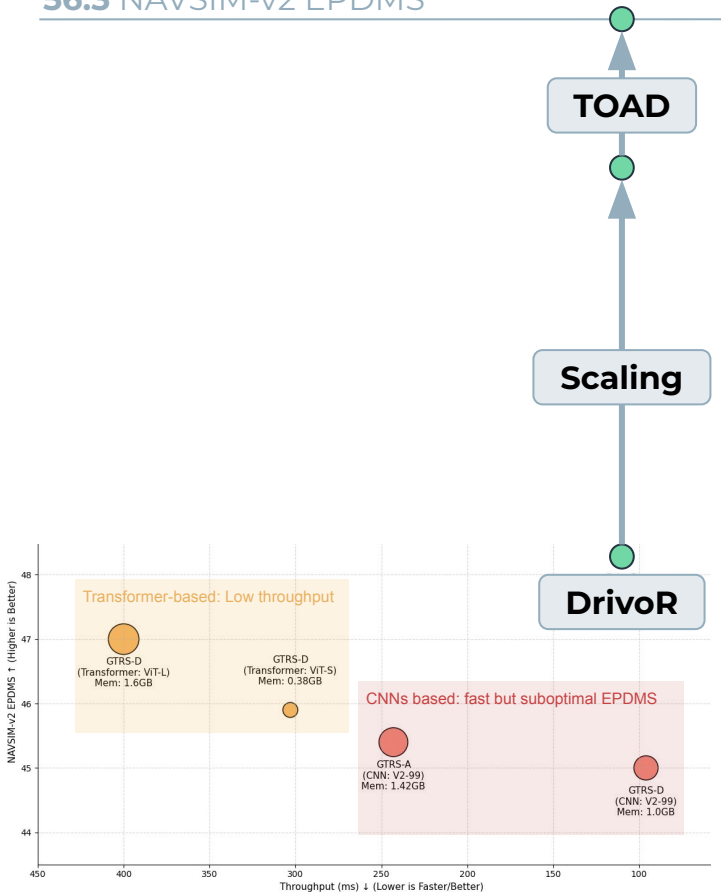
Scalable

Consistent improvement with amount of data

Efficient test-time optimization

Negligible overhead via CEM optimization

56.3 NAVSIM-v2 EPDMS



Open questions

- **From opened-loop to closed-loop?**
  - Simulators (CARLA)
  - Real data? (3DGS reconstruction?)
  - Need a car?
- **Leveraging high framerate simulators**
  - Pictura (ECCV-W 2026)
- **Multi sensors?**
  - Lidar / Radar
    - E.g. Using VaViT (3D processing with vanilla ViT)
- **Robustness?**
  - Different datasets
- **Use time?**
  - No gain to use time so far (benchmark too easy?)

https://valeoai.github.io/


Matthieu


Andrei


Renaud


Tuan-Hung


David


Spyros


Alexandre


Eloi


Yihong


Victor


Nermin

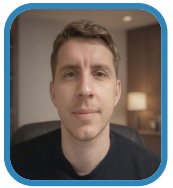

Elias


Yuan


Quan


Ellington


Monika


Marc


Marion


Serkan

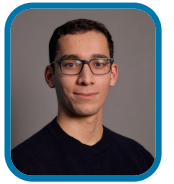

Tetiana


Sophia


Amaia


Salma


Nicolas


Lounès


Florent


Gilles

valeo.ai

https://valeoai.github.io/

Valeo

Driving Change Together

va