ThinkTwice: Towards Scalable Decoders for End-to-End Autonomous Driving

Authors: Xiaosong Jia, Penghao Wu, Li Chen, et al.
Institutions: Shanghai Jiao Tong University, Shanghai AI Laboratory, University of California at San Diego

Reinforcement Learning for Human-Computer Interaction

M11352023 徐彥崴

2024/11/19

Agenda

- Introduction and Background
- Research Objectives
- ThinkTwice Methodology
  - Overall Architecture
  - Key Modules: Look, Prediction, Refinement
- Experiments and Results
- Contributions and Future Directions
- Conclusion

Introduction & Background

End-to-End Autonomous Driving

- Maps raw sensor data (e.g., cameras, LiDAR) to control signals or future trajectories.
- Efficient and avoids error accumulation in traditional modular pipelines.

Challenges in Current Methods

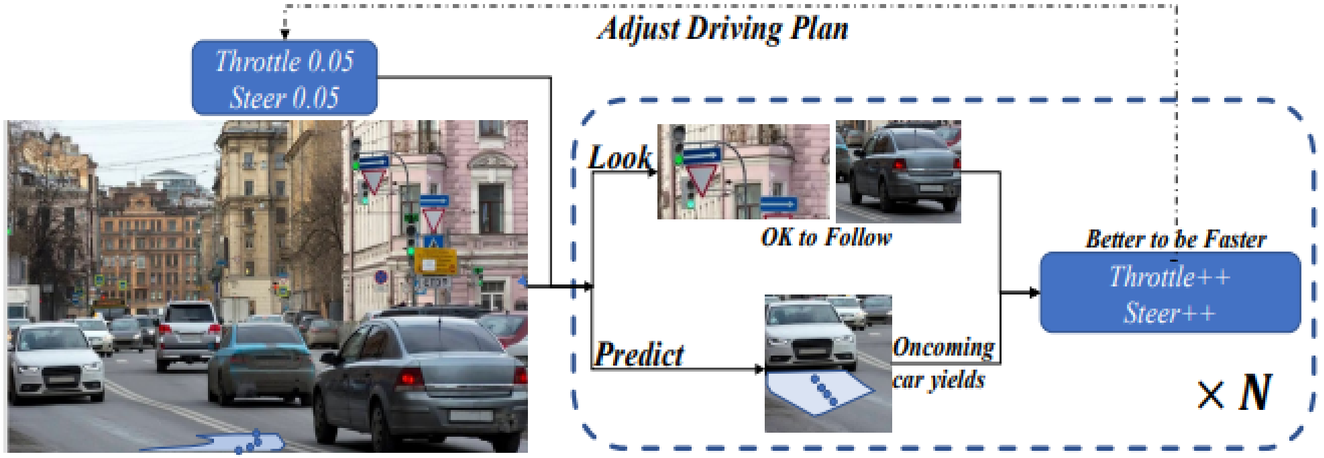
- Encoder-Decoder Paradigm:
  - Encoder: Extracts features from sensor data.
  - Decoder: Predicts future trajectories or actions.
  - Imbalance: Encoders are sophisticated (e.g., ResNet, Transformer), while decoders are simple (e.g., MLPs, GRUs).
- Burden on Decoders:
  - Identify safety-critical regions from massive input fields.
  - Infer future situations accurately.

Research Objectives

1.Fully utilize the encoder's capacity by conditioning it on future predictions.

2.Increase decoder capacity to handle spatial-temporal reasoning effectively.

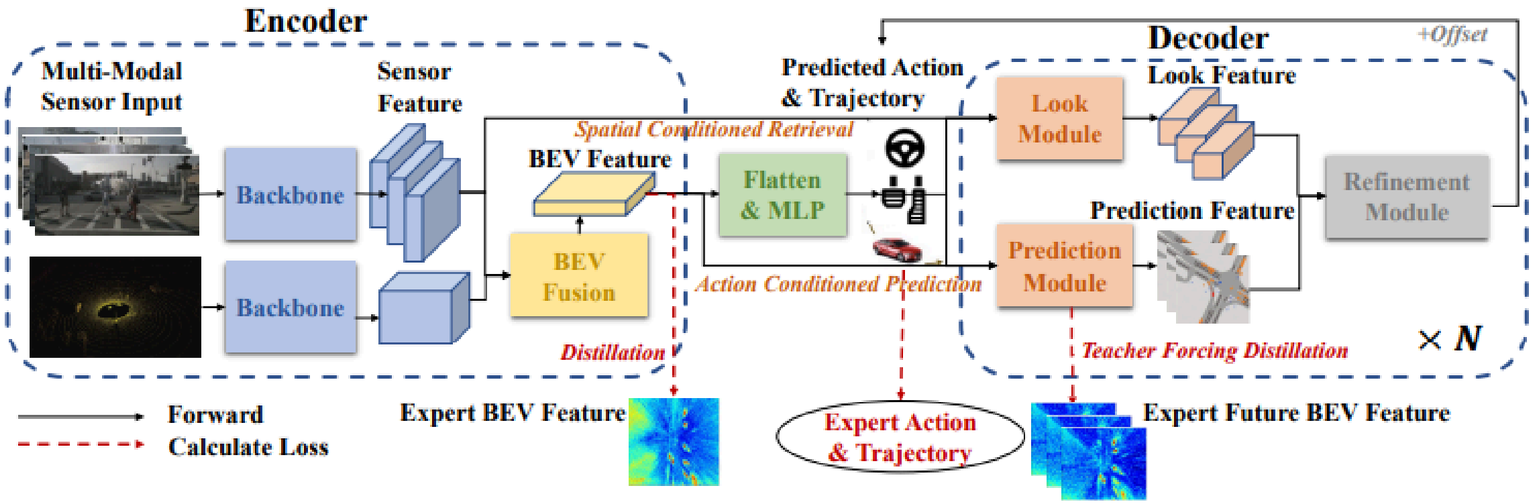
ThinkTwice Architecture



Overview:

- **Input: Multi-modal sensor data(cameras, LiDAR).**
- **Output: Accurate trajectories and control actions.**
- **Modules:**
  - **Coarse Prediction**
  - **Look Module**
  - **Prediction Module**
  - **Refinement Module**

ThinkTwice Architecture



Overview:

- **Input: High-dimensional features extracted from multi-modal sensor data (e.g., cameras, LiDAR).**
- **Output: Structured intermediary representations for coarse trajectory prediction.**
- **Key Processes:**
  - **Feature Conditioning: Applies spatial-temporal priors to encoder outputs.**
  - **Hierarchical Data Flow: Organizes raw sensory features into actionable predictions.**
  - **Intermediate Validation: Ensures refined information before reaching the coarse prediction module.**

Coarse Prediction Module

Role: Initial trajectory and action prediction based on encoder features.

Process:

- **Encoder extracts Bird's Eye View (BEV) features.**
- **Multi-Layer Perceptron (MLP) generates coarse trajectory and action.**
- **Output serves as input to subsequent modules.**

Core Mathematical Equations in the Coarse Prediction Module

1. **BEV Feature Map from the Encoder**:

$$\boldsymbol{H}_{\text{BEV}} \in \mathbb{R}^{C \times BH \times BW},$$

- C: Number of channels in the feature map.
- BH, BW: Height and width of the BEV grid.

2. **Environment Feature Processing and Flattening**:

$$\boldsymbol{H}_{\text{env}} = \text{Flatten}(\text{Conv2D}(\boldsymbol{H}_{\text{BEV}})),$$

- Applies a 2D convolution to downsample $\boldsymbol{H}_{\text{BEV}}$, then flattens the result into a 1D vector $\boldsymbol{H}_{\text{env}}$.
- $\boldsymbol{H}_{\text{env}}$: Encodes the environmental information from cameras and LiDAR.

3. **Routing Information Encoding**:

$$\boldsymbol{H}_{\text{mst}} = \text{MLP}(\text{Routing Information}),$$

- Encodes routing information such as the target point, high-level commands (e.g., go straight, turn left), and current speed into a compact vector.

4. **Initial Action and Trajectory Prediction**:

$$\mathbf{Ctrl}_0, \mathbf{Traj}_0 = \text{MLP}([\boldsymbol{H}_{\text{env}}; \boldsymbol{H}_{\text{mst}}]),$$

- Combines $\boldsymbol{H}_{\text{env}}$ and $\boldsymbol{H}_{\text{mst}}$ to predict the initial action $\mathbf{Ctrl}_0$ and trajectory $\mathbf{Traj}_0$.
- $\mathbf{Ctrl}_0$: Initial action prediction.
- $\mathbf{Traj}_0$: Initial trajectory prediction.

Look Module

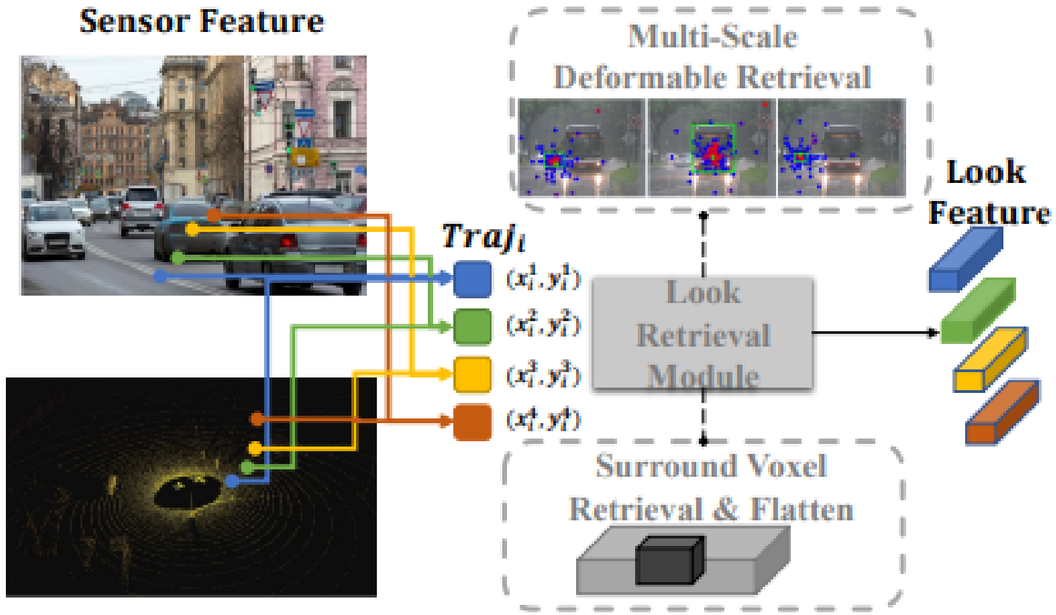


Figure 3. **Overall architecture of Look Module.**

Purpose: Mimics human drivers' behavior by checking target regions for safety.

Steps:

1. **Identify critical regions based on predicted trajectories.**
2. **Retrieve fine-grained features from cameras and LiDAR.**
3. **Integrate features into enhanced representation.**

Core Mathematical Equations in the Look Module

1. **Image Feature Retrieval with Deformable Attention**:

$$\boldsymbol{H}_{i+1}^{\text{img-look}} = \text{DeformAttn}(\boldsymbol{H}_{\text{Img}}; \mathbf{Traj}_i; \boldsymbol{H}_{\text{env}}, \boldsymbol{H}_{\text{mst}}),$$

- $\boldsymbol{H}_{\text{Img}}$: The multi-scale image feature maps.
- $\mathbf{Traj}_i$: The trajectory from the previous layer, serving as the reference point for deformable attention.
- $\boldsymbol{H}_{\text{env}}$: The environment feature, used as the query in the attention mechanism.
- $\boldsymbol{H}_{\text{mst}}$: The multi-scale transformer feature, also used as the query.

2. **LiDAR Feature Retrieval**:

$$\boldsymbol{H}_{i+1}^{\text{lidar-look}} = \text{MLP}(\text{Flatten}(\text{Voxels around } \mathbf{Traj}_i)),$$

- Retrieves voxel features surrounding the predicted trajectory $\mathbf{Traj}_i$.
- Applies a flattening operation followed by an MLP to produce the LiDAR-based Look feature $\boldsymbol{H}_{i+1}^{\text{lidar-look}}$.

3. **Fusion of Image and LiDAR Look Features**:

$$\boldsymbol{H}_{i+1}^{\text{look}} = \text{MLP}([\boldsymbol{H}_{i+1}^{\text{img-look}}; \boldsymbol{H}_{i+1}^{\text{lidar-look}}]),$$

- Concatenates the image-based Look feature $\boldsymbol{H}_{i+1}^{\text{img-look}}$ and the LiDAR-based Look feature $\boldsymbol{H}_{i+1}^{\text{lidar-look}}$.
- Passes the concatenated features through an MLP to produce the final Look feature $\boldsymbol{H}_{i+1}^{\text{look}}$.

4. **Environment Feature Update**:

$$\boldsymbol{H}_{\text{env}} = \text{MLP}(\boldsymbol{H}_{i+1}^{\text{look}}),$$

- Updates the environment feature $\boldsymbol{H}_{\text{env}}$ using the final Look feature $\boldsymbol{H}_{i+1}^{\text{look}}$.

Prediction Module

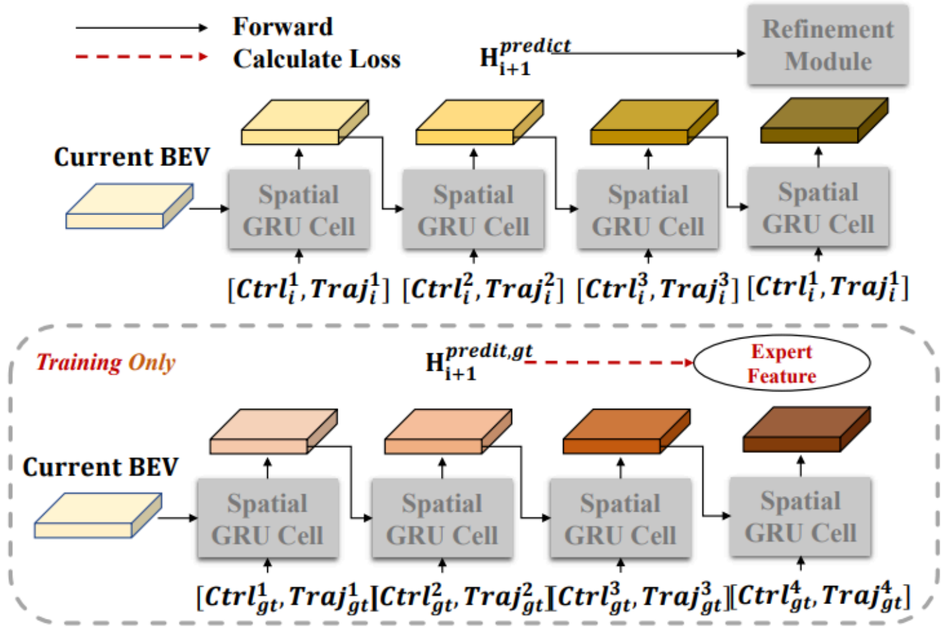


Figure 4. **Overall architecture of Prediction Module.**

Purpose: Anticipates future scenarios conditioned on the vehicle's predicted actions.

Process:

1. **Uses spatial GRU to model the future environment.**
2. **Predicts agent interactions and potential collisions.**
3. **Supervises future scene features via teacher forcing.**

Core Mathematical Equations in the Prediction Module

Output of the Spatial GRU:

$$\boldsymbol{H}_{i+1}^{\text{predict}} = \text{SpatialGRU}(\boldsymbol{H}_{\text{BEV}}; \mathbf{Ctrl}_i^t, \mathbf{Traj}_i^t),$$

- $\boldsymbol{H}_{\text{BEV}}$: The current BEV feature, used as the initial state.
- $\mathbf{Ctrl}_i^t$ and $\mathbf{Traj}_i^t$: The predicted action and trajectory from the previous layer, used as input at each time step.

Shape of the Output:

$$\boldsymbol{H}_{i+1}^{\text{predict}} \in \mathbb{R}^{T \times BH \times BW \times C},$$

- T: The prediction horizon (number of future time steps).
- BH, BW: The height and width of the BEV grid.
- C: The hidden feature dimension.

Teacher Forcing:

$$\boldsymbol{H}_{i+1}^{\text{predict,gt}} = \text{SpatialGRU}(\boldsymbol{H}_{\text{BEV}}; \mathbf{Ctrl}^{\text{gt}}, \mathbf{Traj}^{\text{gt}}),$$

- Uses the ground truth action $\mathbf{Ctrl}^{\text{gt}}$ and trajectory $\mathbf{Traj}^{\text{gt}}$ as input to generate $\boldsymbol{H}_{i+1}^{\text{predict,gt}}$.

Supervision Signal:

$$\mathcal{L}(\boldsymbol{H}_{i+1}^{\text{predict}}, \boldsymbol{H}_{i+1}^{\text{predict,gt}}),$$

- The loss function compares $\boldsymbol{H}_{i+1}^{\text{predict}}$ to the ground truth future scene feature $\boldsymbol{H}_{i+1}^{\text{predict,gt}}$, using Roach BEV features as the target for supervision.

Refinement Module

Role: Refines coarse predictions using insights from Look and Prediction Modules.

Process:

1. **Calculates offset between coarse predictions and ground truth.**
2. **Updates actions and trajectories iteratively.**
3. **Enables stacking for enhanced accuracy.**

Core Mathematical Equations in the Refinement Module

1. **Offset Calculation**:

$$\mathcal{O}_{i+1}^{\text{ctrl}}, \mathcal{O}_{i+1}^{\text{traj}} = \mathbf{MLP}([\boldsymbol{H}_{i+1}^{\text{look}}; \boldsymbol{H}_{i+1}^{\text{predict}}; \mathbf{Ctrl}_i; \mathbf{Traj}_i; \boldsymbol{H}_{\text{env}}; \boldsymbol{H}_{\text{mst}}]),$$

- $\mathcal{O}_{i+1}^{\text{ctrl}}$: Offset for the action prediction.
- $\mathcal{O}_{i+1}^{\text{traj}}$: Offset for the trajectory prediction.
- The offsets are computed using an MLP, based on the Look feature $\boldsymbol{H}_{i+1}^{\text{look}}$, Prediction feature $\boldsymbol{H}_{i+1}^{\text{predict}}$, current action $\mathbf{Ctrl}_i$, current trajectory $\mathbf{Traj}_i$, environment feature $\boldsymbol{H}_{\text{env}}$, and routing feature $\boldsymbol{H}_{\text{mst}}$.

2. **Supervision Signals**:

$$\mathcal{O}_{i+1}^{\text{ctrl}} = \mathcal{L}(\mathbf{Ctrl}_i, \mathbf{Ctrl}^{\text{gt}}), \quad \mathcal{O}_{i+1}^{\text{traj}} = \mathcal{L}(\mathbf{Traj}_i, \mathbf{Traj}^{\text{gt}}),$$

- The loss function $\mathcal{L}$ is the Smooth L1 Loss, which supervises the predicted offsets for both action and trajectory by comparing them to the ground truth.

3. **Action and Trajectory Update**:

$$\mathbf{Ctrl}_{i+1} = \mathbf{Ctrl}_i + \mathcal{O}_{i+1}^{\text{ctrl}}, \quad \mathbf{Traj}_{i+1} = \mathbf{Traj}_i + \mathcal{O}_{i+1}^{\text{traj}}.$$

- The predicted action and trajectory are updated by adding the offsets $\mathcal{O}_{i+1}^{\text{ctrl}}$ and $\mathcal{O}_{i+1}^{\text{traj}}$, respectively.

Experiments and Results

Setup:

- Platform: CARLA Simulator.
- Benchmarks: Town05 Long, Longest6.
- Metrics:
  - Route Completion (RC): Percentage of completed route.
  - Infraction Score (IS): Measures safety violations.
  - Driving Score (DS): Combines RC and IS.

Method	Encoder	Decoder	Modality	Extra Supervision	DS↑	RC↑	IS↑
CILRS [16]	ResNet + Flatten	MLP	C1	None	7.8±0.3	10.3±0.0	0.75±0.05
LBC [10]	ResNet + Flatten	MLP	C3	Expert	12.3±2.0	31.9±2.2	0.66±0.02
Transfuser [13]	Fusion via Transformer	GRU	C3L1	Dep+Seg+Map+Box	31.0±3.6	47.5±5.3	**0.77**±0.04
Roach [87]	ResNet + Flatten	MLP	C1	Expert	41.6±1.8	**96.4**±2.1	0.43±0.03
LAV [9]	PointPaiting	Multi-layer GRUs	C4L1	Expert+Seg+Map+Box	46.5±2.3	69.8±2.3	0.73±0.02
TCP [78]	ResNet + Flatten	GRU	C1	Expert	57.2±1.5	80.4±1.5	0.73±0.02
ThinkTwice	Geometric Fusion in BEV	Look-Predict-Refine	C4L1	Expert+Dep+Seg+Map	**65.0**±1.7	95.5±2.0	0.69±0.05
MILE*† [29]	ResNet + Flatten	GRU	C1	Map+Box	61.1±3.2	**97.4**±0.8	0.63±0.03
Interfuser* [66]	Fusion via Transformer	Transformer + GRU	C3L1	Map+Box	68.3±1.9	95.0±2.9	-
ThinkTwice*	Geometric Fusion in BEV	Look-Predict-Refine	C4L1	Expert+Dep+Seg+Map	**70.9**±3.4	95.5±2.6	**0.75**±0.05

↑ means the higher the better. * denotes using extra data. † denotes no scenarios are involved in the evaluation, which is a much easier benchmark.

Experiments and Results

Setup:

- Platform: CARLA Simulator.
- Benchmarks: Town05 Long, Longest6.
- Metrics:
  - Route Completion (RC): Percentage of completed route.
  - Infraction Score (IS): Measures safety violations.
  - Driving Score (DS): Combines RC and IS.

Method	DS↑	RC↑	IS↑
WOR [8]	23.6	52.3	0.59
LAV [9]	34.2	73.5	0.53
Transfuser [13, 57]	56.7	**92.3**	0.62
ThinkTwice	61.3	73.0	0.81
ThinkTwice*	**66.7**	77.2	**0.84**

Table 2. **Performance on Longest6 benchmark.**

ID	Method	DS↑	RC↑	IS↑
1	Baseline	43.0±3.9	74.2±3.2	0.56±0.05
2	+Segmentation & Depth Task	45.0±2.2	73.2±4.0	0.60±0.04
3	+Segmentation & Depth Projection	57.8±1.5	78.8±3.3	**0.74**±0.03
4	+Two Frames	**59.3**±1.4	**80.2**±3.6	**0.74**±0.04
5	Expert Feature -> BEV Segmentation	50.5±2.3	73.4±3.2	0.70±0.03

Table 3. **Ablation for Encoder Design.** Baseline means original

ID	Method	DS↑	RC↑	IS↑
4	TCP-Head	59.3±1.4	80.2±3.6	**0.74**±0.04
5	+1 Decoder	61.6±1.4	90.8±2.2	0.68±0.04
6	+5 Decoders	**65.0**±1.7	95.5±2.0	0.69±0.05
7	w/o Look	62.4±1.8	93.5±2.1	0.67±0.08
8	w/o Predict	61.7±1.9	96.2±3.0	0.63±0.03
9	w/o TF	62.0±2.2	**97.1**±2.9	0.62±0.04

Table 4. **Ablation for Decoder Design.** The baseline is Model4

Talk About

01

Contributions

02

Future Directions

03

Conclusion

Thank you For Attentation