Generating Multimodal Driving Scenes via Next-Scene Prediction

Supplementary Material

In this supplementary material, we provide the following sections:

- **Additional visualizations** generated by the AMA module to offer deeper insights into its functionality and demonstrate how it ensures consistency between ego-action and map modalities.
- **Detailed explanations** of the tokenization process, token embedding mechanisms, the configuration of hyperparameters for both the model architecture and the training setup, and decoder design.
- **Comprehensive quantitative evaluation** of the generated modalities.
- **Additional visualizations** of the comparative image quality with and without diffusion decoder

A. Visualization of the transformed map in AMA module

To better illustrate the purpose and effect of the transform operation within the AMA module, we employ visualizations to demonstrate how map features are dynamically updated in response to the ego-vehicle's motion.

As shown in Fig. 1, the ego-vehicle performs a forward motion at the current timestep. Consequently, in the transformed map for the next frame, the features from the current map shift backward relative to the ego-vehicle's new position. Regions previously outside the forward view are filled in to account for the unseen areas. This visualization highlights how the transform operation aligns the map features with the vehicle's actions, ensuring spatial and temporal consistency.

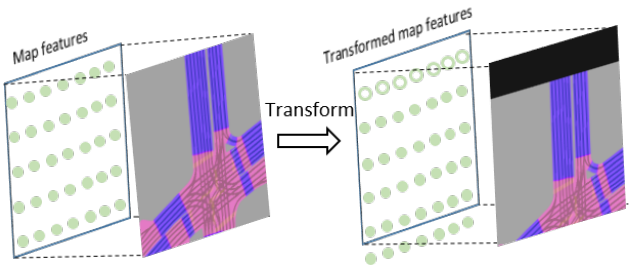

Figure 1. **Visualization of the map and map features, both before and after transformation.** Solid points represent existing map features, while hollow points indicate filled-in features for previously unseen regions. The ego-vehicle's forward motion causes the map features to shift backward in the transformed map of the next frame, illustrating the alignment between the map feature and the ego-vehicle's action

B. Implementation Details

1. Tokenization Method

The tokenization standardizes and discretizes various elements of the driving scenario, including ego-actions, raster maps, agents, and images. This ensures compatibility across modalities and facilitates efficient sequence generation.

1.1 Normalization and Discretization

Continuous variables are normalized to the range $[0, 1]$ using:

$$v_{\text{norm}} = \frac{v - v_{\text{min}}}{v_{\text{max}} - v_{\text{min}}}, \tag{1}$$

where v is the original value, v_{min} and v_{max} are predefined bounds, and v_{norm} is the normalized result.

After normalization, the token ID v_{ID} is assigned by discretizing the normalized value v_{norm} into one of 1024 equal intervals within the range $[0, 1]$. Each interval corresponds to a unique token ID. Specifically, the token ID is determined by identifying the interval that contains v_{norm}:

$$v_{\text{ID}} = i \quad \text{such that} \quad \frac{i}{1024} \leq v_{\text{norm}} < \frac{i+1}{1024}, \tag{2}$$

where $i \in \{0, 1, \ldots, 1023\}$.

1.2 Latent Encoding for Image-like Data

For raster maps and images, two separate pre-trained VQ-GAN models [2] are used to encode the data into latent vectors, respectively. Each latent vector $\mathbf{z}$ is quantized to the nearest codebook entry from a set of $N + 1$ learned embeddings $\{\mathbf{e}_0, \mathbf{e}_1, \ldots, \mathbf{e}_N\}$, assigning the token ID t:

$$t = i \quad \text{such that} \quad \|\mathbf{z} - \mathbf{e}_i\|^2 \leq \|\mathbf{z} - \mathbf{e}_j\|^2, \quad \forall j \neq i, \tag{3}$$

where $i, j \in \{0, 1, \ldots, N\}$.

2. Tokenization Process

The tokenization process for different modalities leverages attribute ranges derived from statistical analysis of the nuPlan dataset [1]. Table 1 provides a summary of these ranges, which serve as the basis for consistent normalization and discretization across modalities.

Table 1. Predefined minimum and maximum values for attribute

Attribute	v_{min}	v_{max}
Position (x)	-64 m	64 m
Position (y)	-64 m	64 m
Position (z)	-5 m	5 m
Agent Length	0 m	15 m
Agent Width	0 m	4 m
Agent Height	0 m	5 m
Heading	$-\pi$ rad	π rad
Speed (v_x)	-20 m/s	20 m/s
Speed (v_y)	-20 m/s	20 m/s
Speed (v_z)	-0.3 m/s	0.3 m/s
Ego-action Displacement (dx)	0 m	10 m
Ego-action Displacement (dy)	-0.5 m	0.5 m
Ego-action Angular Change	-0.25 rad	0.25 rad

2.1 Ego-actions

Ego-vehicle actions include displacements (x, y) and angular changes relative to the previous timestep. These are tokenized using the normalization and discretization process.

2.2 Raster Maps

A 128×128-meter map centered on the ego-vehicle is rasterized into a grid. Each cell represents a fixed spatial resolution and encodes one of six road types: lane, stop line, crosswalk, intersection, middle lane line, and lane connector. This produces a tensor $\mathbf{m} \in \mathbb{R}^{256\times256\times6}$.

A pre-trained VQ-GAN model processes the tensor, quantizing each latent vector to produce token IDs that represent the map structure.

2.3 Agents

Agents—including vehicles, pedestrians, and cyclists—are represented as 11-dimensional vectors. Attributes include position (x, y, z), speed (v_x, v_y, v_z), and heading, all defined relative to the ego-vehicle coordinate system, as well as dimensions (length, width, height) and category.

- **Position, Speed, and Heading**: Tokenized via normalization and discretization.
- **Category**: Assigned discrete IDs—1024 for vehicles, 1025 for pedestrians, and 1026 for cyclists.

If fewer than 64 agents are present in a scenario, *pad tokens* are used. Each *pad token* (ID: 1027) has default values for all attributes, ensuring they do not interfere with downstream processing.

Table 2. Hyperparameter configuration of the model architecture

Hyperparameter	Value
Feature Dimension	768
CA_{hist} Layers in Ego-action Prediction Module	12
CA_{env} Layers in Ego-action Prediction Module	12
Temporal Causal Self-attention Layers in TAR	24
Bidirectional Self-attention Layers in TAR	24
Causal Self-attention Layers in OAR	24

Table 3. Training Configuration

Hyperparameter	Value
Learning Rate	1×10^{-4}
Batch Size	192
Optimizer	AdamW
Number of Training Epochs	300
Block Size	20
Dropout Rate	0.15
Temperature	1.0
Top-k	16
Codebook Size of Ego-action	1024
Codebook Size of Raster Map	8192
Codebook Size of Agent	1028
Codebook Size of Image	8192

2.4 Images

Input images of size 512×256 are encoded using a pre-trained VQ-GAN. Each latent vector is quantized, and token IDs form a grid that retains the spatial and semantic structure of the original image.

3. Token Embedding

To enable the model to process multimodal data effectively, tokens from different modalities are concatenated into a single sequence, which serves as the input to the embedding layer. A start token and an end token delineate the boundaries of each modality, ensuring structural clarity and facilitating positional encoding.

Each modality is assigned a separate learnable codebook to obtain embeddings for its tokens. For the *Image* and *Raster map* modalities, the codebooks are initialized with pre-trained codebook weights from their respective VQ-GAN models at the start of training. Subsequently, a positional encoding is added to each token embedding, as described in the main text. This positional encoding ensures that the sequential order and spatial relationships within the tokenized data are preserved, enabling the model to capture both modality-specific features and their contextual dependencies.

4. Model Hyperparameters and Training Settings

4.1 Model Structure

As shown in Table 2, we list out the detailed hyperparameters of our model structure.

4.2 Training Setting

As shown in Table 3, we provide a detailed list of the hyperparameters used in the training setup. The block size denotes the number of timesteps or data points considered for sequence modeling. The temperature and top-k parameters control the stochasticity and diversity of the output during sampling, where temperature adjusts the probability distribution of predictions, and top-k limits the sampling to the k-most probable candidates.

5. Decoder Design

The decoder transforms tokens back into their original representations. For ego-action and agent attributes, it maps tokens to continuous values based on predefined ranges (see Table. 1 in supplementary materials). For map and image tokens, pre-trained VQGAN decoders reconstruct the original data. The architecture is shown in Fig. 2.

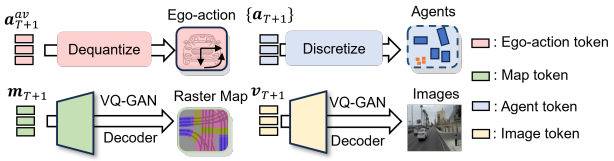

Figure 2. **Visualization of the Decoder.** The decoder maps ego-action and agent attribute tokens to continuous values using predefined ranges, while pre-trained VQGAN decoders reconstruct map and image tokens.

C. Quantitative Evaluation of Generated Modalities

In this section, we present a comprehensive quantitative evaluation of the generated modalities, focusing on image quality, ego-action and agent trajectory prediction, and map realism. For image quality assessment, we employ the Fréchet Inception Distance (FID) metric, as detailed in Tab. 4. To evaluate the accuracy of ego-action and agent trajectory predictions, we utilize the ℓ_2 distance, with results summarized in Tab. 5. As a baseline, we consider a scenario where both agents and the ego-vehicle maintain the velocity from the last frame, providing a reference point for comparison.

Since FID is not well-suited for evaluating multi-channel maps, we instead assess the realism of generated maps using a PatchGAN discriminator derived from our pre-trained VQGAN model. The realism score, averaged across all samples (see Tab. 6), quantifies how convincingly the generated maps resemble real-world counterparts. A higher score indicates greater realism, while a score close to 0 suggests that the discriminator struggles to classify the map as real. Furthermore, we measure the distributional distance between generated map samples and ground truth (GT) map data using Maximum Mean Discrepancy (MMD), providing insights into the alignment of the generated distributions with the real data.

Table 4. FID scores of generated images for different frame lengths. The '-D' indicates the use of a diffusion image decoder.

Method	32 Frames	64 Frames	128 Frames
UMGen	20.91	22.96	27.50
UMGen-D	15.17	18.41	21.86

Table 5. Evaluation of ego-action and agent trajectory predictions.

Method	**ℓ_2 distance to GT (m)**	
	Ego-Action	**Agent Trajectory**
Last Frame Velocity	0.060	1.53
UMGen	0.027	0.54

Table 6. Evaluation of generated map realism.

	Random	UMGen	Ground Truth
Realism Score ↑	-9.3	0.014	0.3692
MMD Score ↓	0.503	0.025	0

D. Image Enhancement via a Diffusion Model

As discussed in the main paper, diffusion models can be a complementary part of our framework—AR ensures structured generation, while diffusion refines image fidelity. This hybrid approach has better image FID scores as in Tab. 4 and visual improvements in Fig. 3.

References

[1] 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. *arXiv preprint arXiv:2106.11810*, 2021. 1

[2] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12873–12883, 2021. 1

Figure 3. **Generated images with and without diffusion models as the image decoder.**