SceneControl: Diffusion for Controllable Traffic Scene Generation Supplementary Materials

Jack Lu[3⋆], Kelvin Wong[1,2⋆], Chris Zhang[1,2], Simon Suo[2], and Raquel Urtasun[1,2]

I. Additional Results

A. Qualitative Results

Generating actors in occluded regions: Handling partially- and/or fully-occluded actors is an important capability for safe self-driving. We demonstrate how SceneControl can be used to generate traffic scenes that exhibit such occlusion *automatically*. Specifically, given an existing traffic scene, we first determine a set of occlusion regions c_{regions} in the scene by ray-casting from the SDV's perspective. Then, we control SceneControl's generation process using a variant of the spatial region guidance function g_{region} described in the main text. In particular, since there may be multiple disjoint occlusion regions in a scene, we use the minimum signed distance function from an actor's centroid to any polygon boundary,

$$g_{\text{regions}}(\mathbf{s}_i, c_{\text{regions}}) = \min\{g_{\text{region}}(\mathbf{s}_i, c) \colon c \in c_{\text{regions}}\} \quad (1)$$

In Fig. 1, we show how using the guidance function g_{regions}, SceneControl can generate realistic scenes with actors that are partially or fully-occluded from the SDV's perspective. Since occlusion regions are automatically extracted from a given scene via ray-casting, we can generate such scenes at scale, opening up exciting possibilities to leverage SceneControl for data augmentation; *e.g.*, to train perception models that are more robust to occlusions.

Interactive scene generation tool on Highway: We use an interactive tool to perform controllable generation of the Highway dataset. As shown in Fig. 5, SceneControl can be used to generate realistic highway scenes that inserts actors in a user-specified spatial region, varies the density/number of actors, and constrains those actors to specific size and speed ranges.

B. Distribution of Scene Statistics

In Fig. 3 and Fig. 4, we visualize ATISS, ATISS++, SceneGen, and SceneControl's distributions of scene statistics in Argoverse2 and Highway respectively. These figures depict the histograms used to compute our distribution JSD metrics. Overall, we observe that traffic scenes generated using SceneControl induces distributions that better match their real counterparts, reaffirming our quantiative results. In particular, the baselines generally induce higher entropy distributions compared to SceneControl. This is especially noticeable on lane deviation in Highway, where ATISS and ATISS++ fail to capture the fact that most vehicles drive near their lane centerlines. Looking at nearest distance in Argoverse2, we can see that SceneControl also captures high density traffic better than ATISS, ATISS++, and SceneGen, which are more likely to generate scenes where vehicles are in collision or are more spread out.

C. *Realism* vs. *controllability*

Fig. 2 depicts the realism *vs*. controllability trade-off as we vary the guidance strength. We observe that the constraint satisfaction success rate increases with guidance strength, with little impact on realism.

II. Model Details

A. Input Parameterization

Actor state: We represent each actor's state $\mathbf{s}_i$ by its centroid location $(x_i, y_i) \in \mathbb{R}^2$, bounding box length and width $(l_i, w_i) \in \mathbb{R}_+$, heading angle $\theta_i \in [0, 2\pi)$, and speed $s_i \in \mathbb{R}_{\geq 0}$. Before processing this representation with our model, we re-scale each attribute in $\mathbf{s}_i$ to $[-1, 1]$. We normalize an actor's centroid location, bounding box length and width, and speed using their respective minimum and maximum values seen in the training split. For heading angle θ_i, we use a biternion representation $(\cos\theta_i, \sin\theta_i)$.

Lane graph: We represent the HD map $\mathbf{m}$ as a lane graph $G = (V, E)$, where each node $u \in V$ is a lane segment and an edge $(u, v) \in E$ indicates that v is a successor, predecessor, or left/right neighbour of u. Following [2], we build each lane graph by discretizing the centerlines of the HD map at regular intervals (3*m* for Argoverse2 and 10*m* for Highway). Each lane segment is annotated with a feature vector describing its location, length, width, heading, curvature, speed limit, lane boundary type, *etc*. We encode crosswalk polygons into the lane graph with regularly sampled nodes and we add edges between a lane segment and a crosswalk node if their distance is less than 2.5*m*.

B. Architecture

We parameterize the noise prediction model $\epsilon_\theta(\mathbf{x}_t, t, \mathbf{m})$ as a transformer-based architecture with a lane graph GNN [2]. At a high level, our architecture consists of three main components: (1) a set of encoders to featurize the input actor states, HD map, and diffusion timestep; (2) a transformer decoder to model actor-to-actor and actor-to-map interactions; and (3) a decoder to predict the forward diffusion noise. We describe each module in detail next.

⋆Indicates equal contribution. [1]Waabi, [2]University of Toronto, [3]New York University. Research done at Waabi. Contact: kwong@waabi.ai

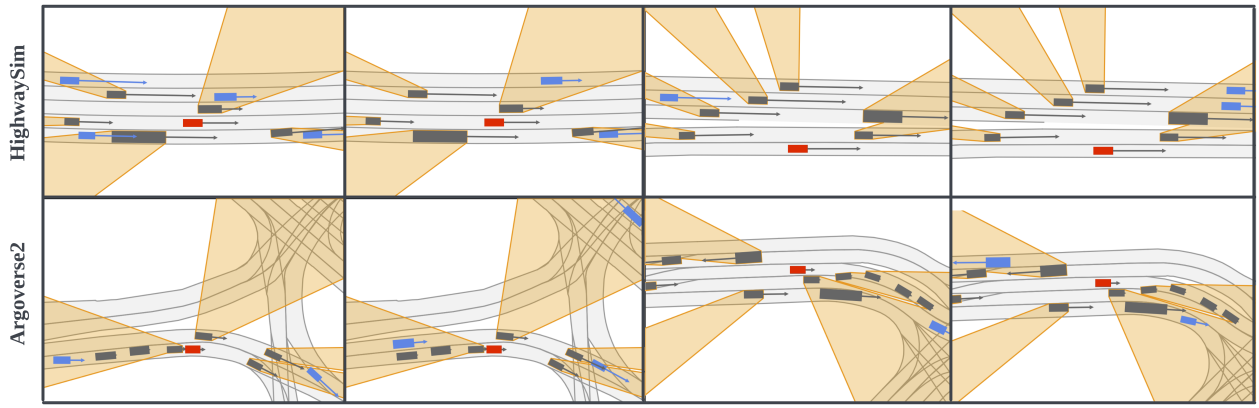

Fig. 1. SceneControl can generate realistic traffic scenes with partially and/or fully-occluded actors from the SDV's perspective. First, we automatically extract occluded regions (in orange) in a scene by ray-casting from the SDV (in red). Then, we use spatial region guidance to control SceneControl to insert actors (in blue) into the occluded regions.

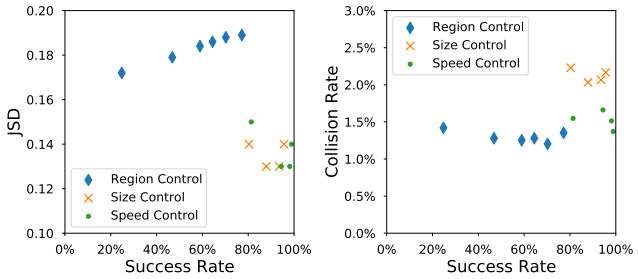

Fig. 2. Ablation study of guidance strength on Argoverse2. Increasing the strength of the guidance function improves constraint satisfaction success rate with little impact on realism, as measured by distribution JSD (left) and collision rate (right).

Actor state encoder: We use an MLP to encode each actor's state separately (but batched together for efficiency.). The MLP consists of four linear layers with layer normalization [1] and ReLU activation in between every two layers.

Lane graph encoder: We use the lane graph encoder proposed in [2] to encode the input lane graph. The encoder consists of four heterogeneous message passing (HMP) layers, followed by layer normalization [1] and a linear layer to produce feature vectors for each lane graph node. Notably, whereas the HMP layers in [2] use pair-wise relative positional information only, we also include each node's absolute position as node attributes, allowing the downstream interactive module to better capture actor-to-actor and actor-to-map interactions in the global traffic scene.

Diffusion timestep encoder: We encode the diffusion timestep t using sinusoidal embeddings [12] followed by two linear layers with a ReLU activation in between.

Interaction module: To model actor-to-actor and actor-to-map interactions, we fuse the actor state, lane graph, and diffusion timestep embeddings using four transformer decoder layers [12]. Each layer consists of a multi-head self-attention layer to fuse actor state features and a multi-head cross-attention layer to fuse lane graph features into the actor state features. Before each pair of self-attention and cross-attention layers, we fuse the diffusion timestep embeddings into the actor state embeddings by taking the sum of their features.

Noise decoder: We use an MLP to predict the forward diffusion noise given each actor's fused features. The MLP consists of four linear layers with layer normalization [1] and ReLU activation in between every two layers. The final layer predicts a 7-dimensional noise vector corresponding to the normalized $\{x_i, y_i, l_i, w_i, \sin\theta_i, \cos\theta_i, s_i\}$.

C. Learning

Our noise prediction model is trained using a noise-matching objective [5],

$$L(\theta) = \mathbb{E}_{\mathbf{x}_0,\mathbf{m},t,\epsilon}\left[\|\epsilon - \epsilon_\theta(\mathbf{x}_t, t, \mathbf{m})\|^2\right] \tag{2}$$

where $\mathbf{x}_0$ and $\mathbf{m}$ is the joint actor states and HD map for a real traffic scene, $t \sim \text{Uniform}(1, T)$ is a diffusion step, and $\mathbf{x}_t$ is the actor states $\mathbf{x}_0$ corrupted with noise $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

To minimize this loss, we use the AdamW optimizer [8] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and weight decay $\lambda = 10^{-5}$. We linearly increase the learning rate from 0.0 to 3.2×10^{-3} over the first training epoch and then gradually anneal the learning rate to 0.0 following a cosine schedule [7]. We train the model over 10 epochs (resp., 20 epochs) with a batch size of 32 (resp., 32) for Argoverse2 (resp., Highway). We found it helpful to train over 16 uniformly-sampled diffusion timesteps for each of the 32 scenes per batch and average the loss across all samples.

D. Guided Sampling

To generate traffic scenes that satisfy high-level constraints, we use *guided sampling* [3], [6] to sample from the perturbed distribution $\tilde{p}_\theta(\mathbf{s}_{1:n}|\mathbf{m}) \propto p_\theta(\mathbf{s}_{1:n}|\mathbf{m})g(\mathbf{s}_{1:n}, \mathbf{m})$. Specifically, given the number of actors n, we first sample n random noise vectors, which we denote $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

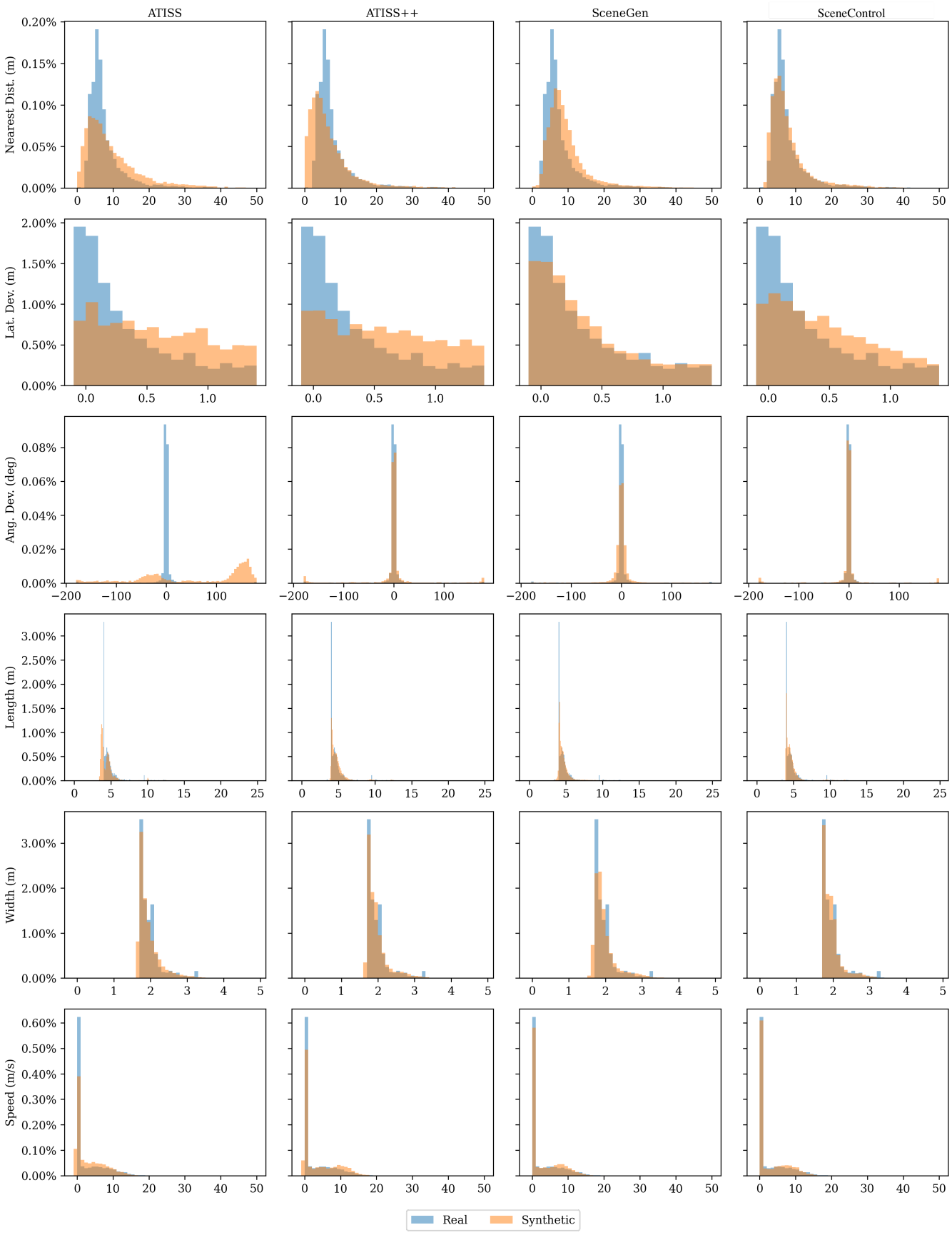

Fig. 3. Distribution of scene statistics on Argoverse2 for ATISS, ATISS++, SceneGen, and SceneControl.

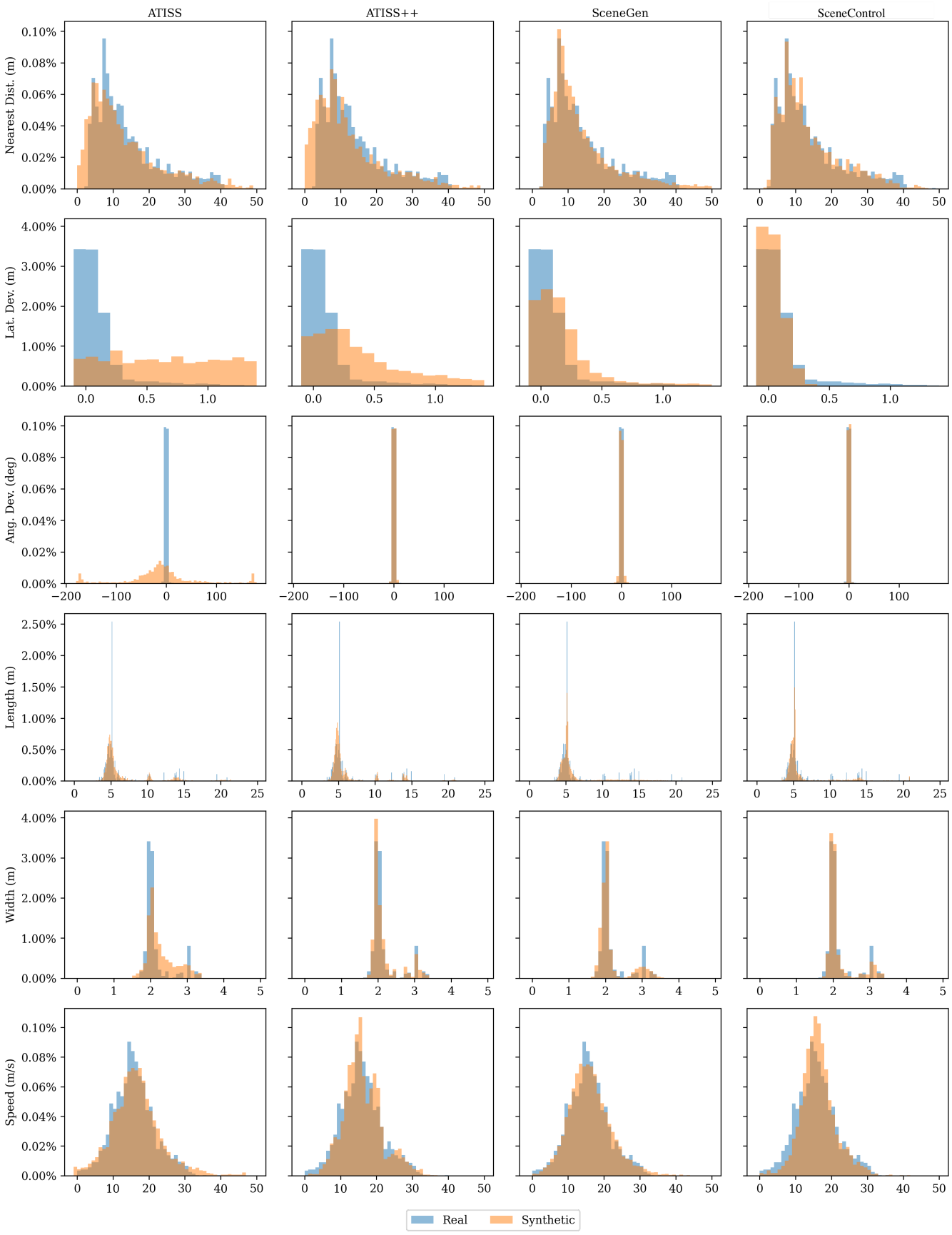

Fig. 4. Distribution of scene statistics on Highway for ATISS, ATISS++, SceneGen, and SceneControl.

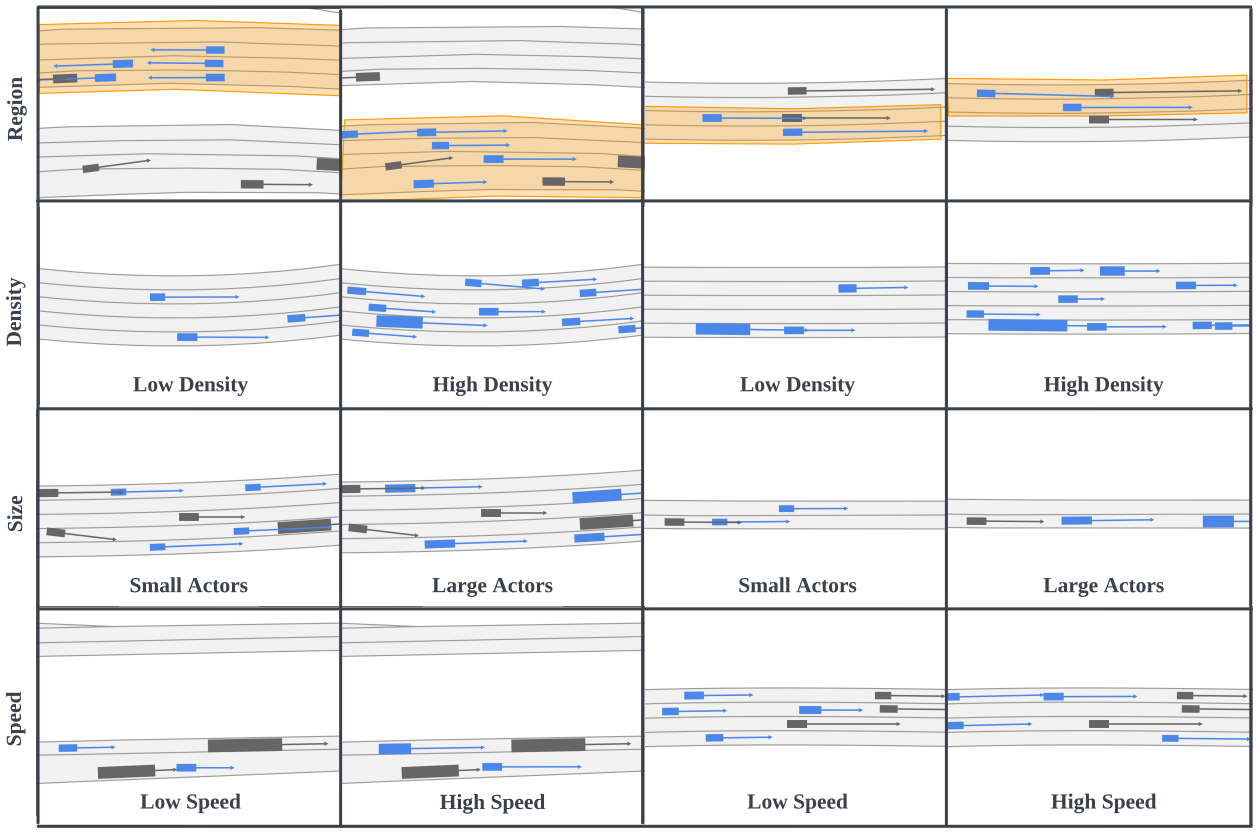

Fig. 5. SceneControl enables controllable traffic scene generation on the Highway dataset. **Row 1** shows realistic scenes that satisfy spatial region constraints; **Row 2** shows scenes with varying numbers of actors; **Row 3** shows scenes with varying actor sizes; and **Row 4** shows scenes with varying traffic speeds. Existing actors in the scene are depicted in gray and inserted actors in blue.

Then, at each step t of the reverse diffusion process, rather than sampling from $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{m}) = \mathcal{N}(\mu_\theta(\mathbf{x}_t, t, \mathbf{m}), \beta_t \mathbf{I})$, we sample from

$$\mathcal{N}(\mu_\theta(\mathbf{x}_t, t, \mathbf{m}) - \gamma_t \beta_t \nabla_{\mathbf{x}_t} g(\mathbf{x}_t, \mathbf{m}), \beta_t \mathbf{I}) \tag{3}$$

where γ_t is a time-varying coefficient that controls the strength of guidance. In our experiments, we use $T = 1000$, a cosine variance schedule β_t [9], and a time-varying guidance strength γ_t that linearly interpolates between γ_T and γ_1. A smaller number of diffusion timesteps T may suffice as well, but we leave this investigation for future work. See Tab. I for the guidance strength settings used for each guidance function we consider. When performing guided sampling over multiple guidance functions, we simply perturb $\mu_\theta(\mathbf{x}_t, t, \mathbf{m})$ with the weighted sum of each guidance function's gradient. We clip each $\mathbf{x}_t$ to [-5.0, 5.0] at each step of the reverse diffusion process, which we find helpful to prevent generating degenerate traffic scenes.

Guidance Function	Argoverse2		Highway	
	γ_1	γ_T	γ_1	γ_T
g_{region}	50.0	1.0	25.0	1.0
g_{attr}	25.0	1.0	25.0	1.0
g_{init}	10.0	10.0	10.0	10.0
$g_{\text{collision}}$	5.0	1.0	10.0	1.0
g_{lane}	0.1	0.0	10.0	1.0

TABLE I: GUIDANCE STRENGTH SETTINGS FOR ARGOVERSE2 AND HIGHWAY. WE USE A TIME-VARYING GUIDANCE STRENGTH γ_t THAT LINEARLY INTERPOLATES FROM γ_T TO γ_1 OVER $T = 1000$ TIMESTEPS.

III. EXPERIMENT DETAILS

A. Datasets

Argoverse2: Our first dataset, Argoverse 2 Sensor [13], has 110,071 urban traffic scenes for training and 23,547 for validation. Each scene provides 3D bounding box labels and an HD map describing the road topology and drivable surfaces of the surrounding area. We define each traffic scene as an $80m \times 80m$ region of interest centered on the SDV. In our experiments, we only consider vehicles that are on a drivable surface. Therefore, we remove all actors that are not vehicles or whose centroid location lies outside of a drivable surface. We classify the following object types as vehicles: `REGULAR_VEHICLE`, `LARGE_VEHICLE`, `RAILED_VEHICLE`, `TRUCK`, `TRUCK_CAB`, `BOX_TRUCK`,

Scene Statistic	Bin Size
Nearest Dist. (m)	1.0
Lat. Dev. (m)	0.1
Ang. Dev. (deg)	5.0
Length (m)	0.1
Width (m)	0.1
Speed (m / s)	1.0

TABLE II: HISTOGRAM HYPERPARAMETERS FOR COMPUTING DISTRIBUTION JSD ON ARGOVERSE2 AND HIGHWAY.

VEHICULAR_TRAILER, MESSAGE_BOARD_TRAILER, BUS, SCHOOL_BUS, and ARTICULATED_BUS.

Highway: Our second dataset, Highway, has 160,000/40,000 highway scenes in its train/validation splits. Each scene provides 3D bounding box labels and an HD map. As in Argoverse2, we define each traffic scene as an $80m \times 80m$ region of interest centered on the SDV.

B. Baselines

SceneGen [11]: Since the source code for SceneGen is not available, we evaluate a re-implementation of the model. Our re-implementation largely follows the description provided in the paper but with hyperparameters adapted to Argoverse2 and Highway. Specifically, we found the following changes to improve training stability and performance: (1) we parameterize the bounding box distribution's covariance matrix using its Cholesky decomposition; (2) we use five mixture components instead of ten; and (3) we add residual connections at each layer of the model's MLPs.

ATISS [10]: We adapt ATISS from indoor scene synthesis to traffic scene generation following its open-source implementation[1]. Our main changes is to: (1) add a decoder to predict an actor's speed; and (2) remove the class decoder since we focus on the vehicles only. Our layout encoder is a ResNet-18 [4] that extracts an average-pooled 64-dimensional feature vector from a bird's eye view raster of the traffic scene, as in SceneGen.

ATISS++: To improve ATISS' performance on traffic scene generation, we make three changes: (1) we extract the lane graph using a lane graph GNN [2]; (2) we use cross-attention to fuse lane graph features into the actor and query features; and (3) we run the transformer decoder autoregressively over each actor attribute. We found that these changes were critical to improving actor-to-actor and actor-to-map interaction reasoning. Specifically, without the first two changes, the model fails to place actors realistically relative to the map. Without the last change, the model tends to learn high entropy distributions that yield unrealistic samples.

C. Metrics

Distribution JSD: We measure the realism of synthetic traffic scenes by the similarity of its scene statistics to those of real traffic scenes. Specifically, we compute the distance between the real distribution of scene statistics $P(x)$ and its synthetic counterpart $Q(x)$ with the Jensen-Shannon divergence,

$$\mathrm{JSD}(P \mid\mid Q) = \frac{1}{2}\,\mathrm{KLD}(P \mid\mid M) + \frac{1}{2}\,\mathrm{KLD}(Q \mid\mid M) \quad (4)$$

where $\mathrm{KLD}(P \mid\mid Q) = \mathbb{E}_{x\sim P}[\log P(x) - \log Q(x)]$ is the Kullback–Leibler divergence of P from Q and $M = \frac{1}{2}(P + Q)$.

Our scene statistics measure each actor's distance to its nearest neighbour (**Nearest Dist.**), lateral deviation to its lane centerline (**Lat. Dev.**), angular deviation from its lane centerline (**Ang. Dev.**), bounding box length (**Length**), bounding box width (**Width**), and speed (**Speed**). For lateral deviatiton and angular deviation, we compute scene statistics only for actors that are within 1.5*m* of a lane centerline; otherwise, the actors are considered off-road. To approximate $P(x)$ and $Q(x)$, we compute histograms of the scene statistic x. We specify the bin size for each statistic manually. See Tab. II for details.

[1] https://github.com/nv-tlabs/ATISS

REFERENCES

[1] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. 2016.

[2] Alexander Cui, Sergio Casas, Kelvin Wong, Simon Suo, and Raquel Urtasun. GoRela: Go relative for Viewpoint-Invariant motion forecasting. Nov. 2022.

[3] Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat GANs on image synthesis. In *NeurIPS*, 2021.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[5] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

[6] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *CoRR*, 2022.

[7] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In *ICLR*, 2017.

[8] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

[9] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *ICML*, 2021.

[10] Despoina Paschalidou, Amlan Kar, Maria Shugrina, Karsten Kreis, Andreas Geiger, and Sanja Fidler. ATISS: Autoregressive transformers for indoor scene synthesis. In *NeurIPS*, 2021.

[11] Shuhan Tan, Kelvin Wong, Shenlong Wang, Sivabalan Manivasagam, Mengye Ren, and Raquel Urtasun. SceneGen: Learning to generate realistic traffic scenes. In *CVPR*, 2021.

[12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. June 2017.

[13] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting. In *NeurIPS Datasets and Benchmarks*, 2021.