

SUPPLEMENTARY MATERIAL FOR “DELTA-TRIPLANE TRANSFORMERS AS OCCUPANCY WORLD MODELS”

Anonymous authors

Paper under double-blind review

The contents of this supplementary material are organized as follows:

- Section S1 presents additional visualization examples.
- Section S2 provides additional implementation details.
- Section S3 provides additional ablation study.
- Section S4 analyzes the limitations of our approach.
- Section S5 discusses potential directions for future work.

S1 ADDITIONAL EXPERIMENTAL RESULTS

More visual comparison results and their corresponding analysis are presented in Figures S1 and S2.

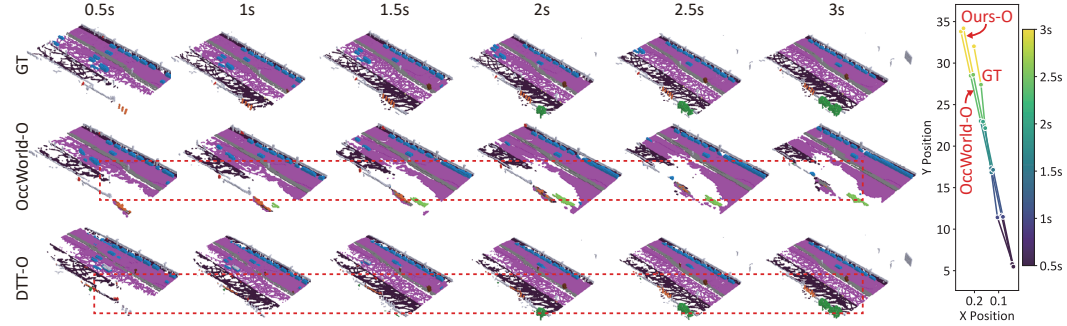


Figure S1: In this scenario, it is clear that the GT contains radially sparse occupied voxels on the ground. Due to the use of VQ-VAE (Van Den Oord et al., 2017) in OccWorld-O, the sparse occupancy data is prone to loss during the reconstruction process. As a result, OccWorld-O’s predictions exhibit large blank areas on the ground, with trees and road boundaries missing in these regions. These details are crucial for future motion planning. In contrast, our method preserves a consistent road layout throughout a 3-second prediction period, and the predicted number of vehicles is more aligned with the GT over time.

S2 ADDITIONAL IMPLEMENTATION DETAILS

S2.1 NETWORK ARCHITECTURE

Figures S3 and S4 illustrate the detailed structure of the encoder and decoder used to obtain triplane representations, respectively.

Figure S5 shows the detailed structure of the scene forecasting model Φ_{fut} . For each feature plane, we perform prediction independently. Taking the xy plane as an example, the input consists of four historical frames that are concatenated along the temporal dimension, resulting in a tensor of size (4, 8, 100, 100). This input is then processed with a standard U-Net architecture (Ronneberger et al., 2015), which is widely used and open-sourced. The U-Net includes a downsampling path and an upsampling path, where the DoubleConv and Down modules are applied, each parameterized by the number of input and output channels. The exact network parameters are provided in our code, while the output dimensions are indicated in gray in Figure S5.

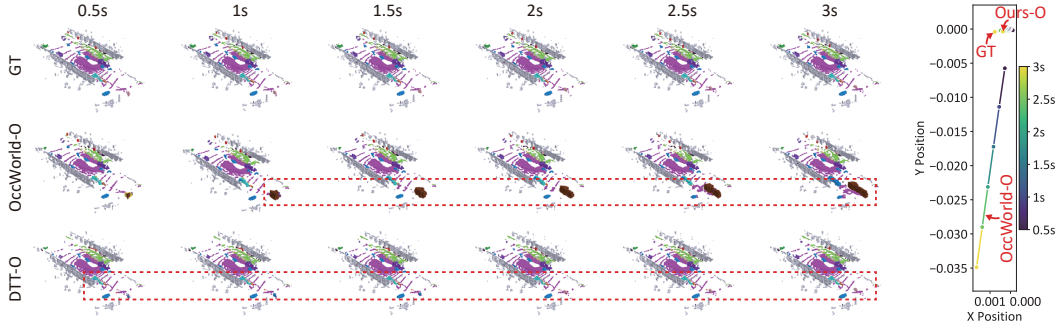


Figure S2: In this scenario, the occupancy data exhibits a sparser radial distribution of LiDAR point clouds, resulting in fewer semantic voxels for distant objects. As a consequence, OccWorld-O incorrectly classifies the small bicycle as a trailer, leading to subsequent predictions of a trailer. In contrast, our method consistently identifies the bicycle correctly. Due to OccWorld-O’s semantic prediction error, it erroneously anticipates a trailer ahead, causing a significant deviation between its predicted trajectory and the GT, while our method remains aligned with GT.

Next, the downsampled features are converted into tokens, where each spatial location is treated as one token. Since the feature maps of the first two scales are too large, we apply Adaptive Pooling to reduce them to 25×25 , preventing overly sparse attention and excessive memory usage in the Transformer. The tokenized features are first passed through a Transformer encoder to capture temporal dependencies across tokens, and then through a Transformer decoder, which predicts the future tokens at each scale. These predicted tokens are then detokenized, i.e., reshaped back into 2D feature planes.

Finally, the reconstructed features are progressively upsampled using the Up modules of the U-Net to produce Δ . This Δ is added to the last xy -plane feature after a 1×1 convolution, yielding the predicted xy plane for the next timestep. The same process is applied to each plane, differing only in input dimensions, as illustrated in Figure 3 of the main text.

S2.2 HYPERPARAMETER SETTINGS

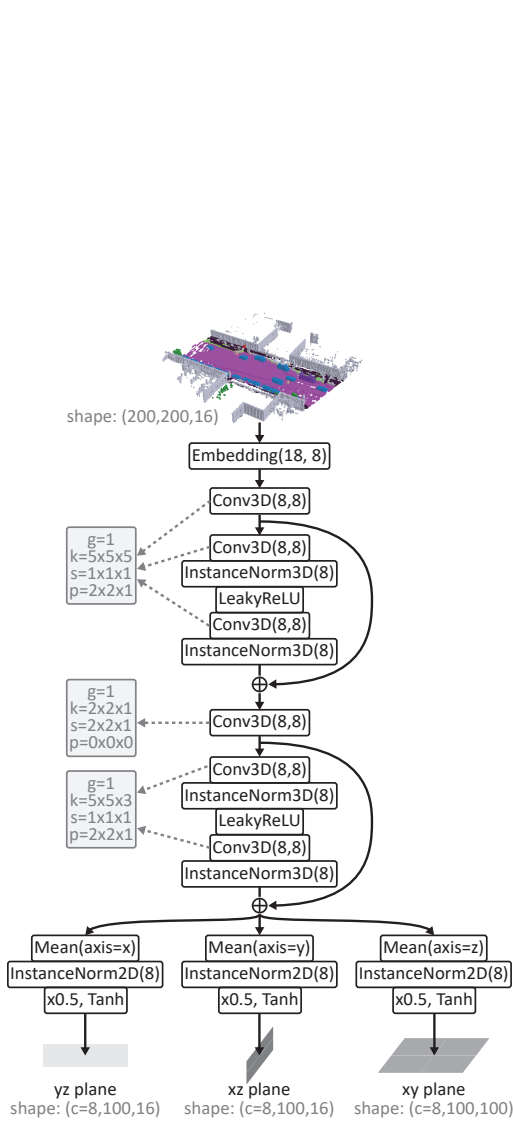
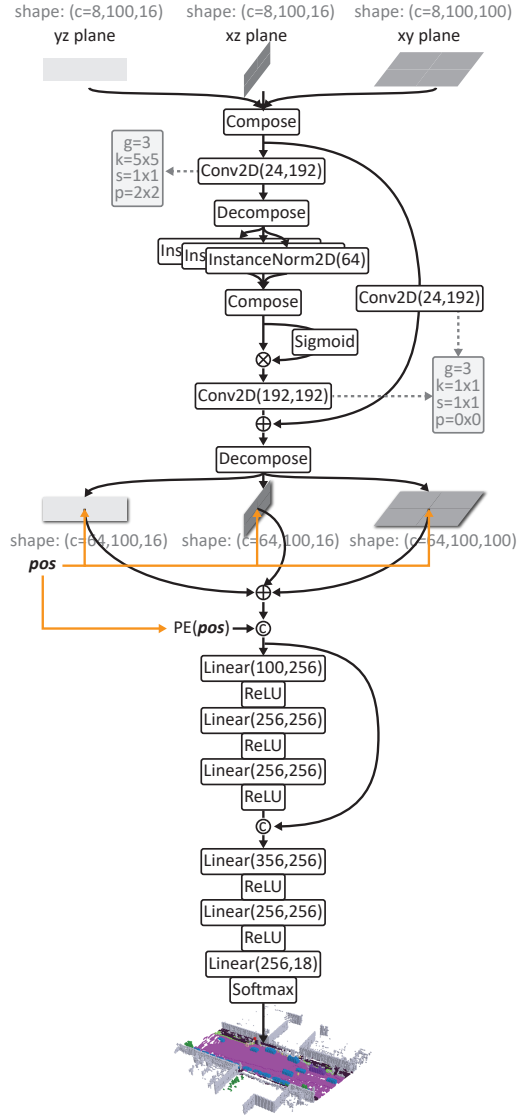
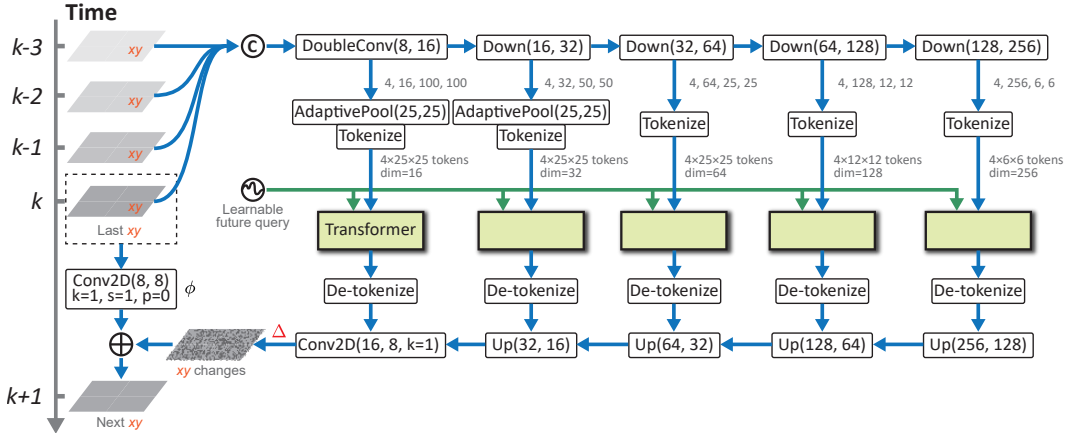
A full list of hyperparameters is provided in Table S1.

S2.3 OPTIMIZATION

To accelerate training and mitigate the accumulation of errors in the model, we adopt the Teacher Forcing technique (Sutskever et al., 2014) for training DTT. Specifically, since the triplane representations for all scenes are pre-trained, the ground truth triplanes for future time steps are already known. As a result, with a certain probability p (we use $p = 0.2$ in this paper), we randomly feed the future GT into the model instead of the predicted values during autoregressive prediction. Concretely, the terms \hat{s}_i^{k-1} in Eqs. (5)(8) are replaced with s_i^{k-1} . This technique accelerates the convergence of training and reduces the error accumulation in the model’s autoregressive predictions.

S3 ADDITIONAL ABLATION STUDY

In Eq. 5, instead of directly adding the latent change element-wise to the most recent historical state, we use a lightweight 1×1 convolutional network mapping ϕ . This operation provides a more flexible aggregation method, introduces only a small computational overhead, and offers a slight performance improvement, as shown in Table S2.

Figure S3: Detailed network structure of Φ_{enc} .Figure S4: Detailed network structure of Φ_{dec} .Figure S5: Detailed network structure of Φ_{fut} .

Hyperparameter	Value	Description
H, W, L	200, 200, 16	Dimensions of the occupancy data
τ_p	4	Number of input past frames
τ_f	6	Number of predicted future frames
c, h, w, l	8, 16, 100, 100	Dimensions of the triplane representation
V	5	Number of scales in DTT
$c_{xy}^{v=1}, w_{xy}^{v=1}, h_{xy}^{v=1}$	16, 100, 100	Feature dimensions for the first scale in the xy plane
$c_{xy}^{v=2}, w_{xy}^{v=2}, h_{xy}^{v=2}$	32, 50, 50	Feature dimensions for the second scale in the xy plane
$c_{xy}^{v=3}, w_{xy}^{v=3}, h_{xy}^{v=3}$	64, 25, 25	Feature dimensions for the third scale in the xy plane
$c_{xy}^{v=4}, w_{xy}^{v=4}, h_{xy}^{v=4}$	128, 12, 12	Feature dimensions for the fourth scale in the xy plane
$c_{xy}^{v=5}, w_{xy}^{v=5}, h_{xy}^{v=5}$	256, 6, 6	Feature dimensions for the fifth scale in the xy plane
$c_{xz}^{v=1}, w_{xz}^{v=1}, h_{xz}^{v=1}$	16, 100, 16	Feature dimensions for the first scale in the xz plane
$c_{xz}^{v=2}, w_{xz}^{v=2}, h_{xz}^{v=2}$	32, 50, 8	Feature dimensions for the second scale in the xz plane
$c_{xz}^{v=3}, w_{xz}^{v=3}, h_{xz}^{v=3}$	64, 25, 4	Feature dimensions for the third scale in the xz plane
$c_{xz}^{v=4}, w_{xz}^{v=4}, h_{xz}^{v=4}$	128, 12, 2	Feature dimensions for the fourth scale in the xz plane
$c_{xz}^{v=5}, w_{xz}^{v=5}, h_{xz}^{v=5}$	256, 6, 1	Feature dimensions for the fifth scale in the xz plane
$c_{yz}^{v=1}, w_{yz}^{v=1}, h_{yz}^{v=1}$	16, 100, 16	Feature dimensions for the first scale in the yz plane
$c_{yz}^{v=2}, w_{yz}^{v=2}, h_{yz}^{v=2}$	32, 50, 8	Feature dimensions for the second scale in the yz plane
$c_{yz}^{v=3}, w_{yz}^{v=3}, h_{yz}^{v=3}$	64, 25, 4	Feature dimensions for the third scale in the yz plane
$c_{yz}^{v=4}, w_{yz}^{v=4}, h_{yz}^{v=4}$	128, 12, 2	Feature dimensions for the fourth scale in the yz plane
$c_{yz}^{v=5}, w_{yz}^{v=5}, h_{yz}^{v=5}$	256, 6, 1	Feature dimensions for the fifth scale in the yz plane
d_{act}	50	Dimension of the action token
λ	0.1	Trade-off factor for \mathcal{L}_{lz} in \mathcal{L}_{occ}
ξ	0.1	Trade-off factor for \mathcal{L}_{occ} in J_{fut}
AdamW(β_1, β_2)	0.9, 0.999	Optimizer used for training DTT
-	0.001	Initial learning rate
-	0.01	Weight regularization factor
-	10	Batch size for training $J_{enc, dec}$
-	1	Batch size for training J_{fut} and J_{act}

Table S1: Complete list of hyperparameters used in DTT.

Table S2: Effect of ϕ .

Idx.	Models	Avg. mIoU \uparrow	Avg. L2 \downarrow	FPS \uparrow
A0	DTT-O	30.85	1.00	26
A1	w/o ϕ	30.45	1.01	26

S4 LIMITATION ANALYSIS

Figure S6 shows a failure case across three frames, where the road boundary at the end contains sparse occupied voxels, making the prediction challenging. Due to the inherent blurring issue of VAE (Choi & Min, 2022; Higgins et al., 2017), OccWorld predictions tend to smooth and homogenize the sparse regions. In contrast, the triplane consists of fine-grained geometric details, which causes our predictions to also exhibit sparse boundaries, resulting in some deviation in the trajectory.

S5 FUTURE DIRECTION

In future work, we will focus on two main aspects:

- **Controllable action injection.** In DTT, actions are currently injected implicitly through history, which limits the ability to condition scene generation on sudden future interventions (e.g., forcing the vehicle to turn left or right at $t = 1.5$ s). Allowing such arbitrary conditioning would enable more flexible and diverse scene generation, supporting goal-directed forecasting.
- **Explicit modeling of dynamic object regions.** In autonomous driving scenarios, new objects may enter while old objects exit. Explicitly predicting these areas can enhance the accuracy of prediction tasks and provide early warnings to drivers. As a result, we will integrate the triplane with explicit predictions of these areas in our future work.

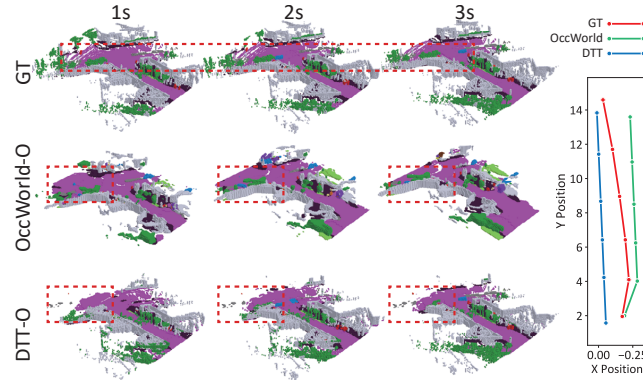


Figure S6: A failure case at a T-junction scenario.

REFERENCES

- Dooseop Choi and KyoungWook Min. Hierarchical latent structure for multi-modal vehicle trajectory forecasting. In *European conference on computer vision*, pp. 129–145. Springer, 2022.
- Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International conference on learning representations*, 2017.
- Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention—MICCAI 2015: 18th international conference, Munich, Germany, October 5–9, 2015, proceedings, part III 18*, pp. 234–241. Springer, 2015.
- Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.
- Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.