

## Supplementary Material

### A. Derivation of Cross-Modal Embedding

In this section, we provide a complete derivation of the objective function of shared cross-modal embedding. Based on Eqn. (9), the KL divergence approximating the conditional posteriors is written as follows:

$$KL(q(\mathbf{z}|\mathbf{y}, \mathbf{c})||p(\mathbf{z}|\mathbf{y}, \mathbf{c})) \\ = - \int q(\mathbf{z}|\mathbf{y}, \mathbf{c}) \log \left( \frac{p(\mathbf{z}|\mathbf{y}, \mathbf{c})}{q(\mathbf{z}|\mathbf{y}, \mathbf{c})} \right) d\mathbf{z} \geq 0, \quad (17)$$

where  $\mathbf{y}$  and  $\mathbf{c}$  respectively denotes the data point and condition. Assuming multiple input sources  $i \in \{\text{LiDAR, RGB, ...}\}$  are available, we derive the objective for an arbitrary number of modalities, yielding

$$\sum_i KL \left( q_i(\mathbf{z}|\mathbf{y}_i, \mathbf{c}_i) || p(\mathbf{z}|\mathbf{y}_i, \mathbf{c}_i) \right) \\ = \sum_i - \int q_i(\mathbf{z}|\mathbf{y}_i, \mathbf{c}_i) \log \left( \frac{p(\mathbf{z}|\mathbf{y}_i, \mathbf{c}_i)}{q_i(\mathbf{z}|\mathbf{y}_i, \mathbf{c}_i)} \right) d\mathbf{z} \geq 0, \quad (18)$$

which is still under the premise that the KL divergence is non-negative. We apply Baye's theorem

$$p(\mathbf{z}|\mathbf{y}, \mathbf{c}) = \frac{p(\mathbf{y}|\mathbf{z}, \mathbf{c})p(\mathbf{z}|\mathbf{c})}{p(\mathbf{y}|\mathbf{c})} \quad (19)$$

to Eqn. (18) and employ

$$\int q(\mathbf{z}|\mathbf{y}, \mathbf{c}) d\mathbf{z} = 1, \quad (20)$$

yielding

$$\sum_i KL \left( q_i(\mathbf{z}|\mathbf{y}_i, \mathbf{c}_i) || p(\mathbf{z}|\mathbf{y}_i, \mathbf{c}_i) \right) \\ = \sum_i \left( - \int q_i(\mathbf{z}|\mathbf{y}_i, \mathbf{c}_i) \log \left( \frac{p_i(\mathbf{y}_i|\mathbf{z}, \mathbf{c}_i)p(\mathbf{z}|\mathbf{c}_i)}{q_i(\mathbf{z}|\mathbf{y}_i, \mathbf{c}_i)} \right) d\mathbf{z} \right. \\ \left. + \log p(\mathbf{y}_i|\mathbf{c}_i) \right) \geq 0. \quad (21)$$

By transferring integral to the other side, we get

$$\sum_i \log p(\mathbf{y}_i|\mathbf{c}_i) \\ \geq \sum_i \left( \int q_i(\mathbf{z}|\mathbf{y}_i, \mathbf{c}_i) \log \left( \frac{p(\mathbf{z}|\mathbf{c}_i)}{q_i(\mathbf{z}|\mathbf{y}_i, \mathbf{c}_i)} \right) d\mathbf{z} \right. \\ \left. + \int q_i(\mathbf{z}|\mathbf{y}_i, \mathbf{c}_i) \log p_i(\mathbf{y}_i|\mathbf{z}, \mathbf{c}_i) d\mathbf{z} \right). \quad (22)$$

Therefore, the evidence lower bound of multiple data over the conditional likelihood is given by

$$\log \left( \prod_i p(\mathbf{y}_i|\mathbf{c}_i) \right) \geq \sum_i \left( -KL(q_i(\mathbf{z}|\mathbf{y}_i, \mathbf{c}_i)||p(\mathbf{z}|\mathbf{c}_i)) \right. \\ \left. + \mathbb{E}_{\sim q_i(\mathbf{z}|\mathbf{y}_i, \mathbf{c}_i)} [\log p_i(\mathbf{y}_i|\mathbf{z}, \mathbf{c}_i)] \right), \quad (23)$$

which completes the proof.

### B. Dataset Preprocessing

#### B.1. Frontal View Input

**Segmentation Map** Frontal view RGB images (top row in Figure. 5) are used to estimate the semantic labels in the scene. We first run the DeepLab-V2 [5] model trained on the Cityscapes dataset [10]. Then, we leave the background labels with stationary structures (*i.e.*, road, sidewalk, building, etc.) to get the background map as shown in the second row of Figure 5. For the static stream, we directly use the background map at the first observation time of each scenario as segmentation input  $S$  to the model.

**Ego-future Elimination** To eliminate the effect of ego-future from frontal view prediction, we introduce the absolute coordinate where the motion of traffic agents is not influenced by the ego-vehicle. To eliminate the ego-motion for a dynamic stream, we define the new coordinates at the first observation time step  $t = 1$  for every trajectory segments  $\mathbf{T}^i$  (of length  $\tau + \delta$ ), and all foreground objects are projected into this space. For this, we conduct the following procedure. First, the point cloud in the world coordinates is projected into the image space to grab the corresponding RGB information. Next, we transform the point cloud to the first frame using GPS/IMU position estimates. Finally, the transformed point cloud is projected into the blank image and displayed using previously acquired RGB information. The transformed RGB images generated from this procedure are shown in the third row of Figure 5.

**Optical Flow Images** We first locate the ground truth bounding box in the image space. Then, we go through ego-future elimination steps using these images with corresponding point clouds. Output foreground images (last row in Figure 5) are used to compute optical flow  $\mathcal{O}$  by running the TV-L1 [48] algorithm. The positions of each traffic agent is also computed by this procedure. Images have a dimension of  $414 \times 125$ .

#### B.2. Top-down View Image

**Segmentation Map** From each trajectory segment, we grab the segmentation label for the point cloud from the RGB-based background map. Then, the point cloud is transformed onto the top-down image space that is discretized with a resolution of  $0.5 \text{ m}$  as illustrated in the second row

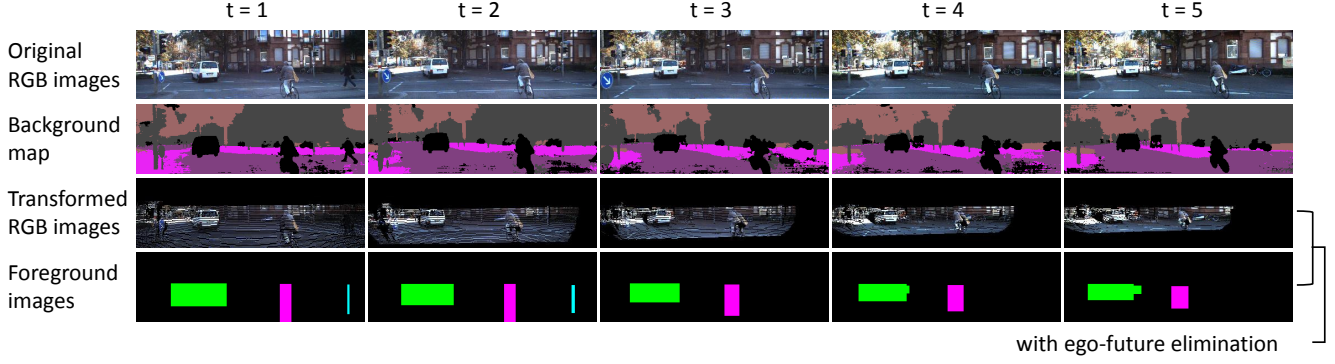


Figure 5: Frontal view input.

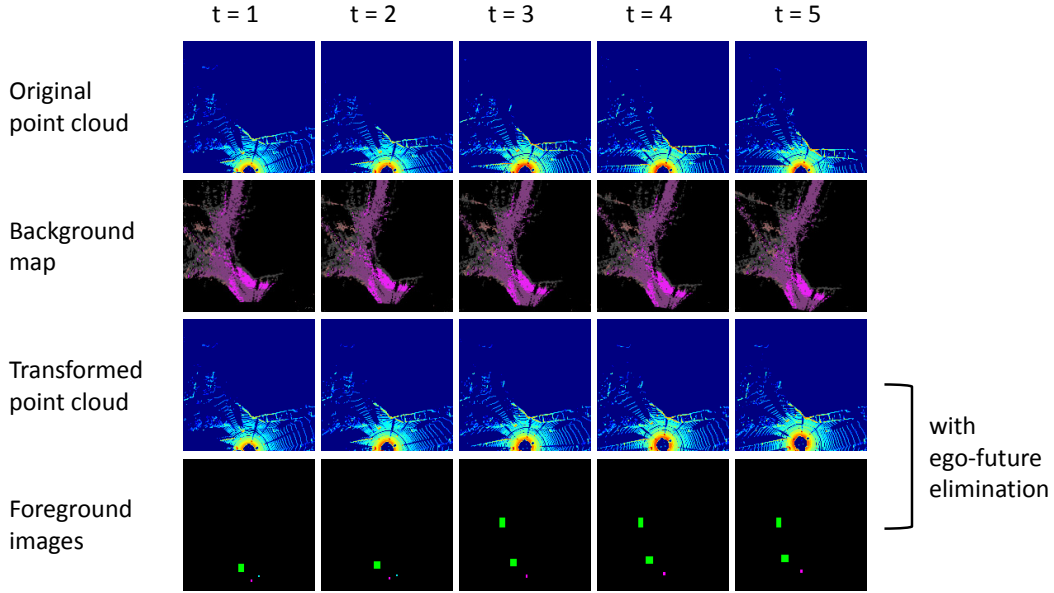


Figure 6: Top-down view input.

of Figure 6. We use the map at time  $t = 1$  as  $S$  in top-down view.

**Ego-future Elimination** Similar to frontal view images, we first transform each point cloud to the local coordinates at  $t = 1$  using GPS/IMU position estimates. The transformed point clouds are projected into the top-down view image space with a resolution of  $0.5\text{ m}$ . The third row of Figure 6 shows the transformed point clouds that are created using the original point cloud in the first row of Figure 6.

**Optical Flow Images** The ground truth bounding box of objects are first processed to eliminate the effect of ego-future. Then, they are drawn in the top-down view image space as displayed in the last row of Figure 6. The final output has a dimension of  $160 \times 160$  that corresponding to  $80\text{ m}$  to the longitudinal direction and  $\pm 40\text{ m}$  to the lateral direction.

### B.3. Absolute vs Relative Motions in Frontal View

In Figure 8, we compare the absolute motions introduced in the proposed method with the relative motions used in other approaches. The absolute trajectory (in red dots) is intuitive by eliminating the ego-future. In contrast, the relative trajectory (in orange dots) is not interpretable without extra information of future ego-motion that should be separately predicted (as in [18]), considering the uncertainty [30].

## C. Additional Evaluation

### C.1. Qualitative Results in Frontal View

We additionally show the qualitative results of the proposed method in frontal view using the KITTI [13] dataset. As in Figure 7, our frontal view prediction is based on the

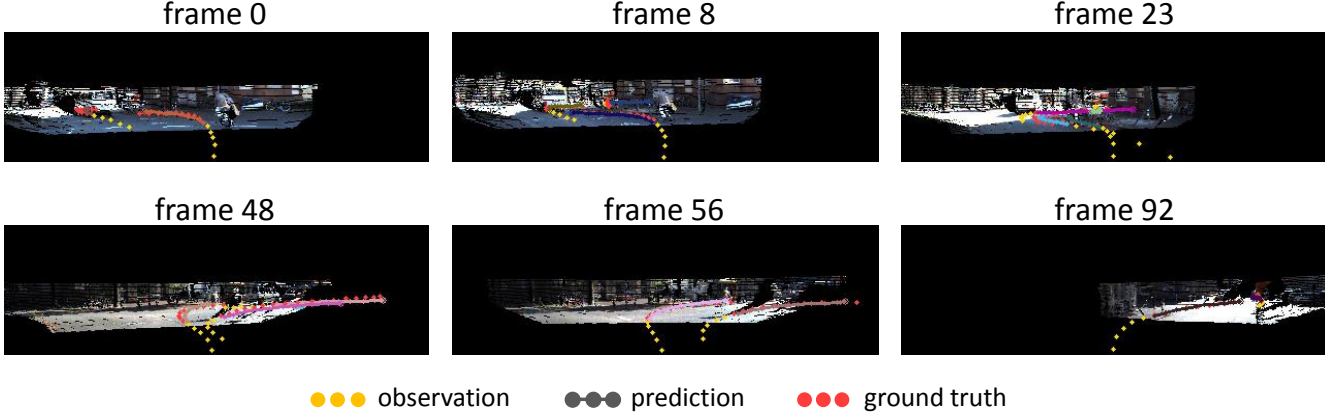


Figure 7: Additional qualitative results evaluated using KITTI in frontal view.



Figure 8: Comparison between absolute (red trajectory) and relative (orange trajectory) motions of a cyclist.

absolute locations (with ego-future elimination) in the local coordinates of the first frame of each trajectory segment. The proposed approach recognizes the road layouts and accurately predict interactive future motions of different types of road agents. Note that we visualize top-1 prediction in this figure.

### C.2. Quantitative Results in Top-down View

Table 6 shows an additional study on extra metrics. We provide FDE at 4.0 *sec* with standard deviation computed from five experiments. It demonstrates that our approach consistently provides robust prediction capabilities with lower standard deviation.

### C.3. Additional Top-down View Videos

We provide additional video clips (*scenario1.mp4*, *scenario2.mp4*, *scenario3.mp4*) to visualize entire 20 predictions of all traffic agents in the scene. The videos are generated using three interactive scenarios of the H3D dataset. Please check attached videos for additional qualitative evaluation (Color codes: Blue - past observation, Green - ground truth, Red - our predictions).

## D. Implementation

We share details of our model architectures.

Method	FDE ( <i>m</i> ) ↓
Const-Vel [40]	1.54
S-STGCNN [31]	1.49±0.0239
Trajectron++ [39]	1.63±0.0126
Ours	<b>0.77±0.0039</b>

Table 6: Additional study using H3D. FDE at 4.0 *sec* with standard deviation is reported in *meters*.

### D.1. Social Behavior Encoding

**Input Layer for External Features** We design a 3D CNN module  $CNN_{3D}$  to extract temporal representations  $\mathbf{f}_T$ . This module is constructed using 4 sets of [3D conv - 1D conv]. The first two sets have 3D conv layer with a filter size of  $5 \times 5 \times 3$  and stride of  $3 \times 3 \times 1$ , and the last two 3D conv layers have a stride of  $2 \times 2 \times 1$ . The final layer merges time channels as one, which results in the output feature of size  $\text{batch} \times 4 \times 6 \times 512$  for frontal view and that of  $\text{batch} \times 5 \times 5 \times 512$  for top-down view.

The spatial features  $\mathbf{f}_S$  are extracted from the stationary environment using a 2D CNN module  $CNN_{2D}$ . We use 4 sets of [2D conv - 1D conv] where the filter sizes are same as those of  $CNN_{3D}$  without depth channel of 3D conv. Note that we use the output of the third set as mid-level semantic context  $\Omega_{x_\tau^k}$  to capture the local environment while encoding the motion of the target agent  $k$ .

**Input Layer for Node Features** Past motion history  $\mathbf{x}^k$  of the target agent  $k$  is encoded into high-dimensional feature representations  $\mathcal{U}_k$  through MLP with 2 fully connected layers. The resulting features of size  $\text{batch} \times 512$  are added to the corresponding local perception  $\Omega_{x_\tau^k}$ . We run LSTM and use the last hidden state of size  $\text{batch} \times 512$  to initialize the node feature of the agent  $k$ ,  $\mathbf{h}_{(0)}^k$ .

For the rest of agents, we follow the similar procedure using MLP and LSTM with the relative motion information. Each individual last hidden state of size  $\text{batch} \times 512$  is used to initialize  $\mathbf{h}_{(0)}^j \forall j \in \{1, \dots, K\} \setminus \{k\}$ .

**GNN Layer** The message generation is implemented using three layers of MLP, all with size 256. We use an additional layer of MLP with a same size during the readout phase.

## D.2. Cross-Modal Embedding

**Encoder** The ground-truth future motion  $\mathbf{y}^k$  of size  $\text{batch} \times 10 \times 2$  is first reshaped to  $\text{batch} \times 20$  and processed through six fully connected layers, each with output size 1024, 6400, 25600, 6400, 1024, and 512. Each layer has a subsequent leaky ReLU layer, and the intermediate features are conditioned by  $\mathbf{c}^k$ . From the output of the penultimate layer, the first 256 dimension is used as mean and the next 256 dimension is used as standard deviation.

**Decoder** The sampled latent variable is decoded through the 8 fully connected layers. The output size is 1024, 6400, 25600, 102400, 25600, 6400, 1024, and 20, respectively. All layers come together with a ReLU activation function, and each intermediate output is conditioned by  $\mathbf{c}^k$ .