Camera input | Ground truth | VPN [35] | PON [39] | Lift-Splat [37] | **Ours Seg.** | **Ours Inst.**

Figure 5: Qualitative comparison of bird's-eye view prediction with published methods on NuScenes. The predictions of our model are much sharper and more accurate. Contrary to previous methods, FIERY can separate closely parked cars and correctly predict distant vehicles (near the top of the bird's-eye view image).

A. Additional Results

A.1. Comparison with published methods

Figure 5 shows a qualitative comparison of the predictions from our model with previous published methods, on the task of present-frame bird's-eye view semantic segmentation.

A.2. Benefits of temporal fusion

When predicting the present-frame bird's-eye view segmentation, incorporating information from the past results in better predictions as shown in Figure 6.

A.3. Probabilistic modelling

Generalised Energy Distance. Let $(\hat{Y},\ \hat{Y}')$ be samples of predicted futures from our model, (Y, Y') be samples of ground truth futures and d be a distance metric. The Generalised Energy Distance D_{GED} is defined as:

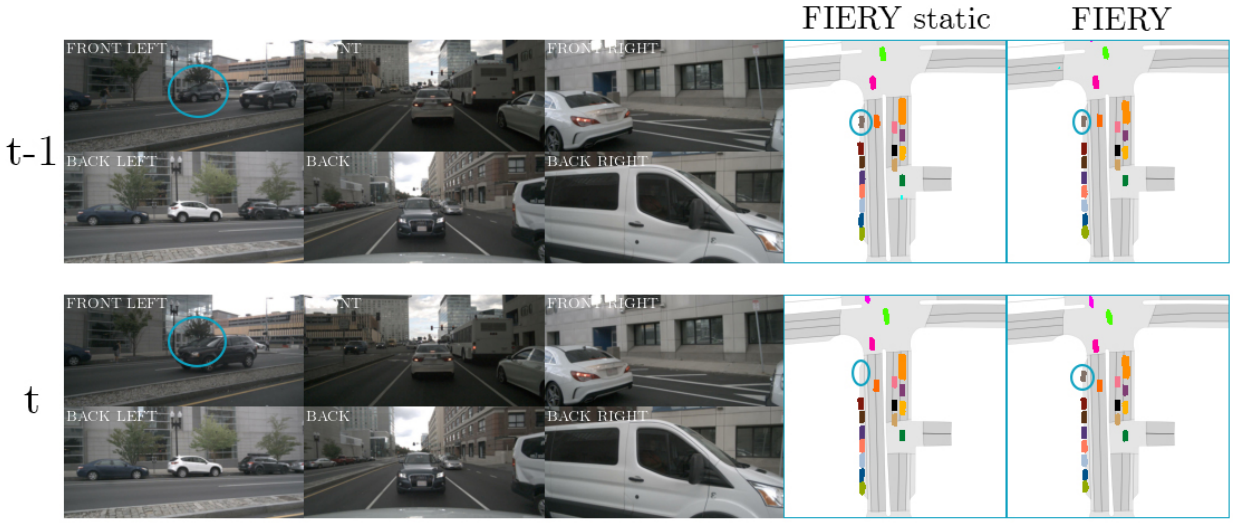
$$D_{\text{GED}} = 2\mathbb{E}[d(\hat{Y}, Y)] - \mathbb{E}[d(\hat{Y}, \hat{Y}')] - \mathbb{E}[d(Y, Y')] \quad (7)$$

We set our distance metric d to $d(x, y) = 1 - \text{VPQ}(x, y)$. Since we only have access to a unique ground truth future Y, D_{GED} simplifies to:

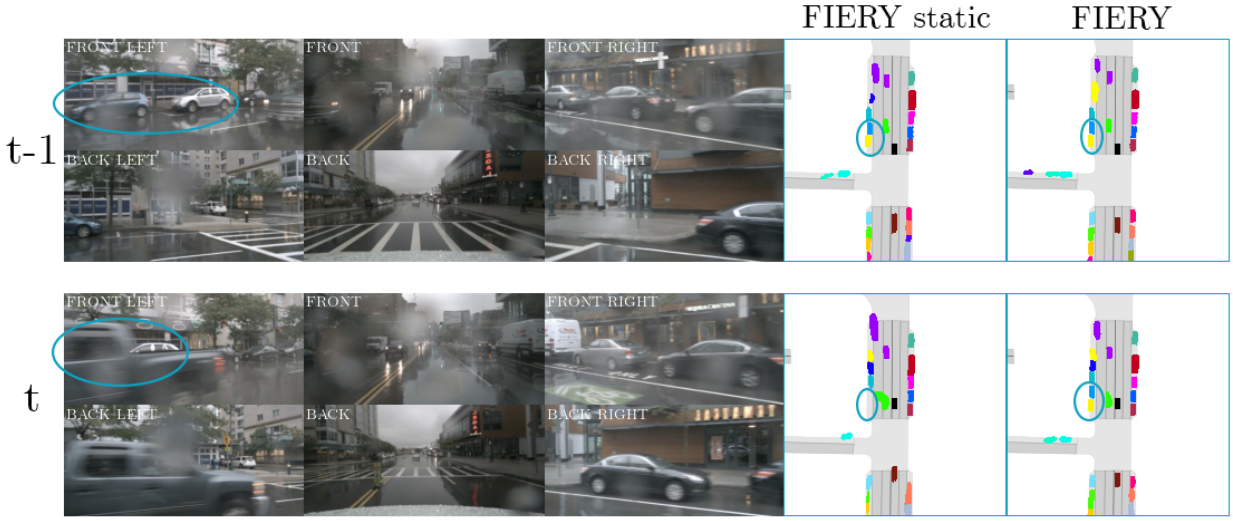
$$D_{\text{GED}} = 2\mathbb{E}[d(\hat{Y}, Y)] - \mathbb{E}[d(\hat{Y}, \hat{Y}')] \quad (8)$$

Baselines. We describe below the baselines we compare our model to in Table 4.

- **M-Head.** The M-head model inspired by [41] outputs M different futures. During training, the best performing head backpropagates its loss with weight $(1 - \epsilon)$



(a) The vehicle parked on the left-hand side is correctly predicted even through the occlusion.



(b) The two vehicles parked on the left are heavily occluded by the 4x4 driving on the opposite lane, however by fusing past temporal information, the model is able to predict their positions accurately.

Figure 6: Comparison of FIERY Static (no temporal context) and FIERY (1.0s of past context) on the task of present-frame bird's-eye view instance segmentation on NuScenes. FIERY can predict partially observable and occluded elements, as highlighted by the blue ellipses.

while the other heads are weighted by $\frac{\epsilon}{M-1}$. We set $\epsilon = 0.05$.

- **Bayesian Dropout.** We insert a dropout layer after every 3D temporal convolution in the temporal model. We also insert a dropout layer in the first 3 layers of the decoder, similarly to [2]. We set the dropout parameter to $p = 0.25$.
- **Classical VAE.** We use a Centered Unit Gaussian to constrain our probability distribution similarly to the technique used in [3]. Different latent codes are sampled from $\mathcal{N}(0, I_L)$ during inference.

	Generalised Energy Distance (↓)	
	Short	Long
M-Head	96.6	122.3
Bayesian Dropout	92.5	116.5
Classical VAE	93.2	109.6
FIERY	**90.5**	**105.1**

Table 4: Generalised Energy Distance on NuScenes, for 2.0s future prediction and $M = 10$ samples, showing that our model is able to predict the most accurate and diverse futures.

A.4. Visualisation of the learned states

We run a Principal Component Analysis on the states s_t and a Gaussian Mixture algorithm on the projected features in order to obtain clusters. We then visualise the inputs and predictions of the clusters in Figures 7, 9 and 10. We observe that examples in a given cluster correspond to similar scenarios. Therefore, we better understand why our model is able to learn diverse and multimodal futures from a deterministic training dataset. Since similar scenes are mapped to the same state s_t, our model will effectively observe different futures starting from the same initial state. The present distribution will thus learn to capture the different modes in the future.

A.5. Temporal horizon of future prediction

Figure 8 shows the performance of our model for different temporal horizon: from 1.0s to 8.0s in the future. The performance seems to plateau beyond 6.0s in the future. In such a large future horizon, the prediction task becomes increasingly difficult as (i) uncertainty in the future grows further in time, and (ii) dynamic agents might not even be visible from past frames.

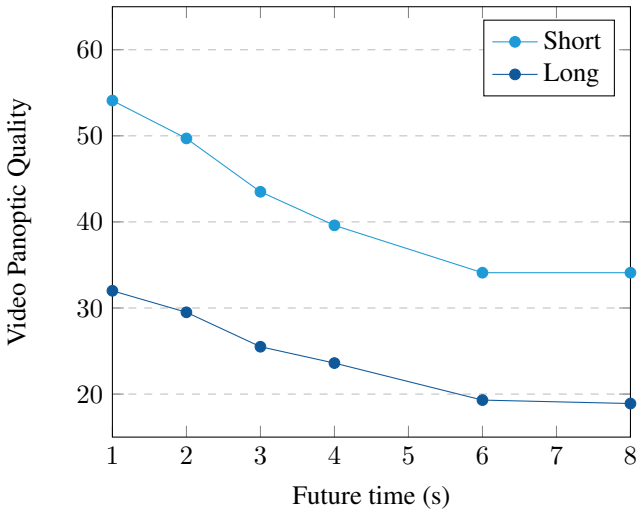
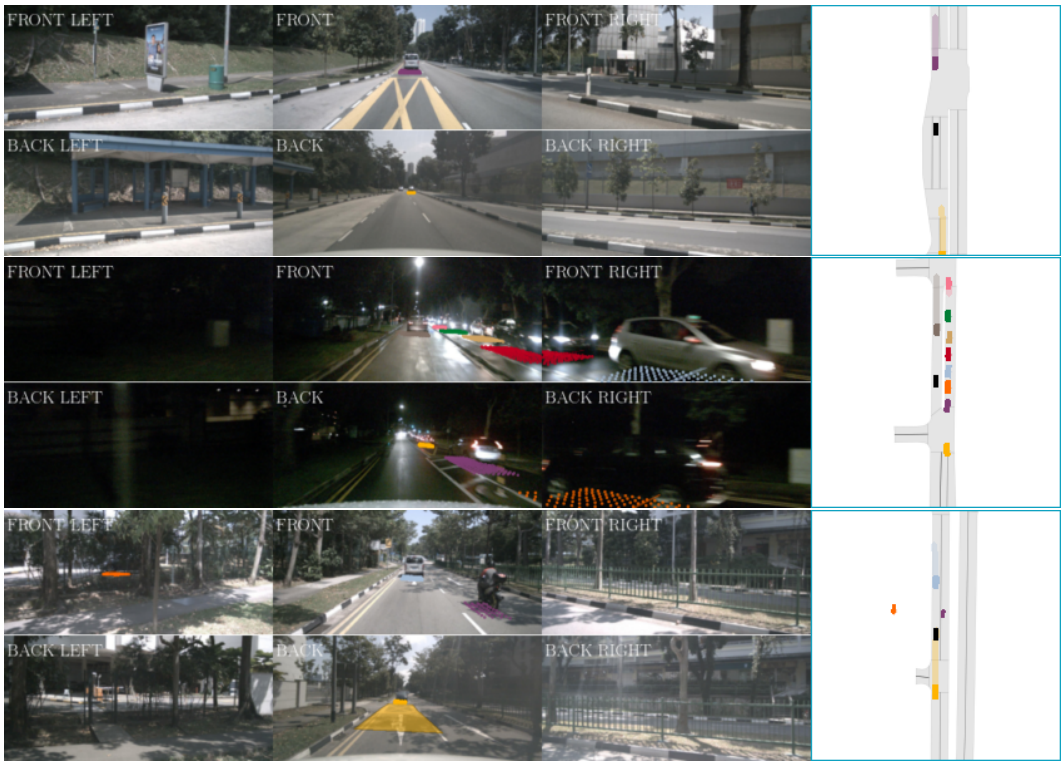


Figure 8: Future prediction performance for different temporal horizons. We report future Video Panoptic Quality on NuScenes at different capture sizes around the ego-car: $30\text{m} \times 30\text{m}$ (Short) and $100\text{m} \times 100\text{m}$ (Long).

(a) Approaching an intersection.

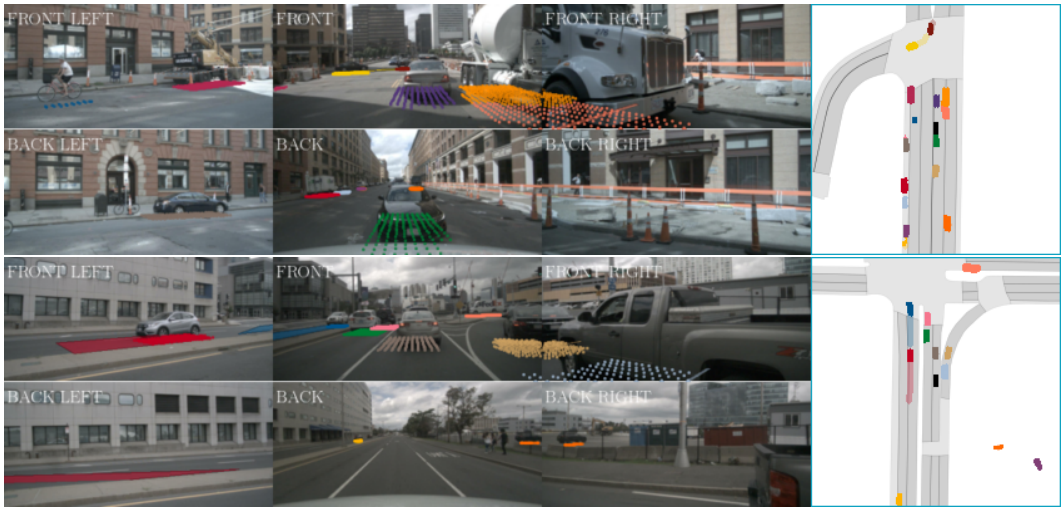
Figure 7: An example of cluster obtained from the spatio-temporal states s_t by running a Gaussian Mixture algorithm on the NuScenes validation set. Our model learns to map similar situations to similar states. Even though the training dataset is deterministic, after mapping the RGB inputs to the state s_t, different futures can be observed from the same starting state. This explains why our probabilistic paradigm can learn to predict diverse and plausible futures.



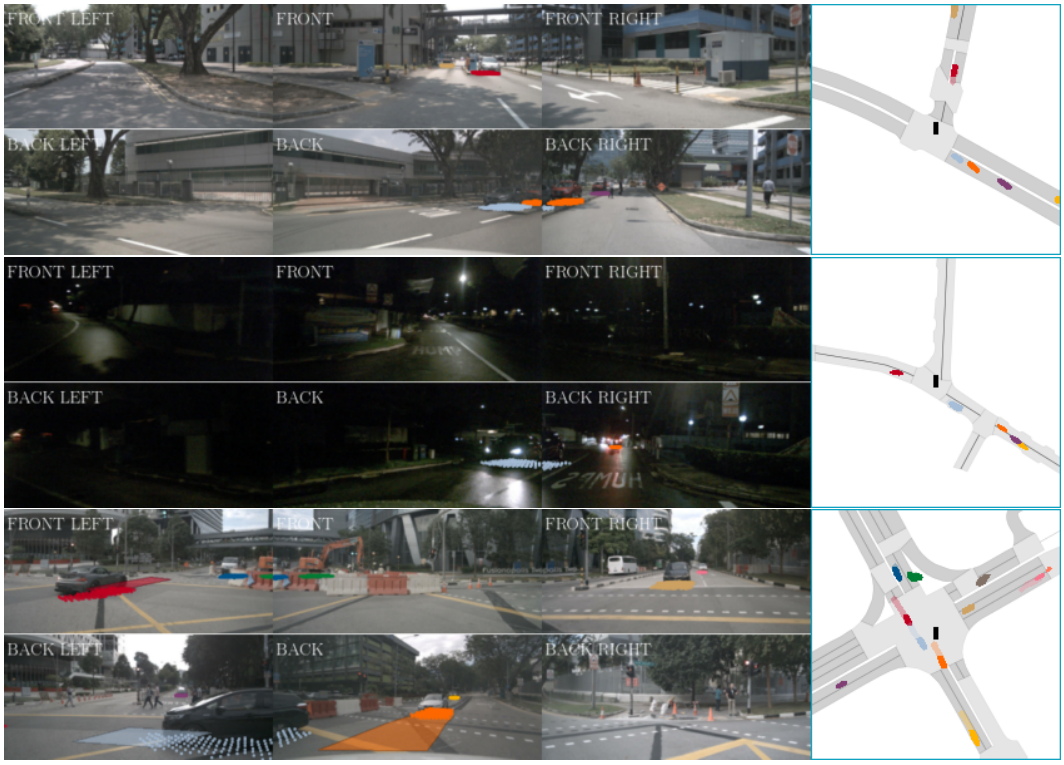
(a) Cruising behind a vehicle.

(b) Driving on open road.

Figure 9: More example of clusters.



(a) Stuck in traffic.



(b) Turning right at an intersection.

Figure 10: More example of clusters.

B. Model and Dataset

B.1. Model description

Our model processes $t = 3$ past observations each with $n = 6$ cameras images at resolution $(H_{\text{in}}, W_{\text{in}}) = (224 \times 480)$, i.e. 18 images. The minimum depth value we consider is $D_{\text{min}} = 2.0\text{m}$, which corresponds to the spatial extent of the ego-car. The maximum depth value is $D_{\text{max}} = 50.0\text{m}$, and the size of each depth slice is set to $D_{\text{size}} = 1.0\text{m}$.

We use uncertainty [24] to weight the segmentation, centerness, offset and flow losses. The probabilistic loss is weighted by $\lambda_{\text{probabilistic}} = 100$.

Our model contains a total of 8.1M parameters and trains in a day on 4 Tesla V100 GPUs with 32GB of memory. All our layers use batch normalisation and a ReLU activation function.

Bird's-eye view encoder. For every past timestep, each image in the observation $O_t = \{I_t^1, ..., I_t^n\}$ is encoded: $e_t^k = E(I_t^k) \in \mathbb{R}^{(C+D)\times H_e \times W_e}$. We use the EfficientNet-B4 [45] backbone with an output stride of 8 in our implementation, so $(H_e, W_e) = (\frac{H_{\text{in}}}{8}, \frac{W_{\text{in}}}{8}) = (28, 60)$. The number of channel is $C = 64$ and the number of depth slices is $D = \frac{D_{\text{max}} - D_{\text{min}}}{D_{\text{size}}} = 48$.

These features are then lifted and projected to bird's-eye view to obtain a tensor $x_t \in \mathbb{R}^{C\times H\times W}$ with $(H, W) = (200, 200)$. Using past ego-motion and a spatial transformer module, we transform the bird's-eye view features to the present's reference frame.

Temporal model. The 3D convolutional temporal model is composed of *Temporal Blocks*. Let C_{in} be the number of input channels and C_{out} the number of output channels. A single Temporal block is composed of:

- a 3D convolution, with kernel size $(k_t, k_s, k_s) = (2, 3, 3)$. k_t is the temporal kernel size, and k_s the spatial kernel size.
- a 3D convolution with kernel size $(1, 3, 3)$.
- a 3D global average pooling layer with kernel size $(2, H, W)$.

Each of these operations are preceded by a feature compression layer, which is a $(1, 1, 1)$ 3D convolution with output channels $\frac{C_{\text{in}}}{2}$.

All the resulting features are concatenated and fed through a $(1, 1, 1)$ 3D convolution with output channel C_{out}. The temporal block module also has a skip connection. The final feature s_t is in $\mathbb{R}^{64\times 200\times 200}$.

Present and future distributions. The architecture of the present and future distributions are identical, except for the number of input channels. The present distribution takes as input s_t, and the future distribution takes as input the concatenation of $(s_t, y_{t+1}, ..., y_{t+H})$. Let $C_p = 64$ be the number of input channel of the present distribution and $C_f = 64 + C_y \cdot H = 88$ the number of input channels of the future distribution (since $C_y = 6$ and $H = 4$). The module contains four residual block layers [18] each with spatial downsampling 2. These four layers divide the number of input channels by 2. A spatial average pooling layer then decimates the spatial dimension, and a final (1,1) 2D convolution regress the mean and log standard deviation of the distribution in $\mathbb{R}^L \times \mathbb{R}^L$ with $L = 32$.

Future prediction. The future prediction module is made of the following structure repeated three times: a convolutional Gated Recurrent Unit [4] followed by 3 residual blocks with kernel size $(3, 3)$.

Future instance segmentation and motion decoder. The decoder has a shared backbone and multiple output heads to predict centerness, offset, segmentation and flow. The shared backbone contains:

- a 2D convolution with output channel 64 and stride 2.
- the following block repeated three times: four 2D residual convolutions with kernel size (3, 3). The respective output channels are [64, 128, 256] and strides [1, 2, 2].
- three upsampling layers of factor 2, with skip connections and output channel 64.

Each head is then the succession two 2D convolutions outputting the required number of channels.

B.2. Labels generation

We compute instance center labels as a 2D Gaussian centered at each instance center of mass with standard deviation $\sigma_x = \sigma_y = 3$. The centerness label indicates the likelihood of a pixel to be the center of an instance and is a $\mathbb{R}^{1\times H\times W}$ tensor. For all pixels belonging to a given instance, we calculate the offset labels as the vector pointing to the instance center of mass (a $\mathbb{R}^{2\times H\times W}$ tensor). Finally, we obtain future flow labels (a $\mathbb{R}^{2\times H\times W}$ tensor) by comparing the position of the instance centers of gravity between two consecutive timesteps.

We use the *vehicles* category to obtain 3D bounding boxes of road agents, and filter the vehicles that are not visible from the cameras.

We report results on the official NuScenes validation split. Since the Lyft dataset does not provide a validation set, we create one by selecting random scenes from the dataset so that it contains roughly the same number of samples (6,174) as NuScenes (6,019).