Supplementary Material – V2VNet: Vehicle-to-Vehicle Communication for Joint Perception and Prediction

Tsun-Hsuan Wang[1,2*] Sivabalan Manivasagam[2,3] Ming Liang[2,3]
Bin Yang[2,3] Wenyuan Zeng[2,3] Raquel Urtasun[2,3]
[1]National Tsing Hua University [2]Uber Advanced Technologies Group [3]University of Toronto
{tsunhsuan.wang, manivasagam, ming.liang, byang10, wenyuan, urtasun}@uber.com

Abstract

*In this supplementary, we cover additional details of the V2V-sim dataset and our novel V2VNet. In Sec. 1 we visualize how we collect V2V data using a high-fidelity LiDAR simulator and show more dataset statistics. In Sec. 2 we provide implementation details on different components of V2VNet. In Sec. 3, we show performance on objects at different distances, conduct ablation studies, and perform qualitative analysis. Additionally, we include a video (**submission_3838.mp4**) that contains* i*) a brief overview of our paper* ii*) a demonstration of the V2V-sim data collection process and* iii*) Perception and Prediction (P&P) results with V2V communication.*

1. V2V-Sim

In Fig. 1, we demonstrate how we collect V2V data using a high-fidelity LiDAR simulator [3]. On the left, we show the 3D scene recreated from a real-world snippet where actors' 3D-assets are placed based on temporal tracks for all actors in the scene. The green vehicle is the ego-vehicle, *i.e.*, the vehicle that recorded the snippet in the real world. The vehicles colored in red, yellow, and cyan represent the candidate self-driving vehicles (SDV) that could be transmitting information. Candidate V2V vehicles are non-parked vehicles that are within the 70-meter broadcast range of the ego-vehicle. On the right, we show the LiDAR measurements of the four SDVs (ego-vehicle plus three candidate vehicles) generated from their own viewpoints, where the color of point cloud corresponds to an SDV with the same color in the 3D scene on the left. Note that although only four SDVs are shown here, the total number of SDVs in a scene are not confined to four in the data collection.

In Fig. 2 left, we show the histogram for the number of candidate vehicles in V2V-sim. Note that it is nearly gaussian-distributed. Most frames have around 5 to 10 candidate vehicles in the scene. In some very extreme cases, where the traffic is heavy and many vehicles cluster together, there may be up to sixty candidate vehicles. In Fig. 2 right, we show percentage of SDVs among all vehicles in the scene as a function of distance from the ego-vehicle. As we increase the broadcast range, there is a larger proportion of vehicles that can be included as candidate vehicles. The percentage of candidate vehicles roughly scales linearly with broadcast range.

2. Implementation Details

Compression: We use a similar design as Ballé *et al.*'s variational image compression algorithm [2]. The compression model consists of two convolutional autoencoders and an entropy model. One autoencoder learns a compressed representation for the P&P intermediate representation. The other autoencoder is trained to compress side information, which serves as additional information to signal modification in the entropy model. The entropy model estimates the probability of every discrete symbol, which is required in entropy coding. The first encoder/decoder consists of two 3x3 convolution/deconvolution filters with generalized divisive normalization (GDN) [1] in between. The second encoder/decoder consists of one 3x3 convolution/deconvolution filter, two 5x5 convolution/deconvolution filters with stride 2, and ReLU as activation function. The

*work done during internship in Uber ATG

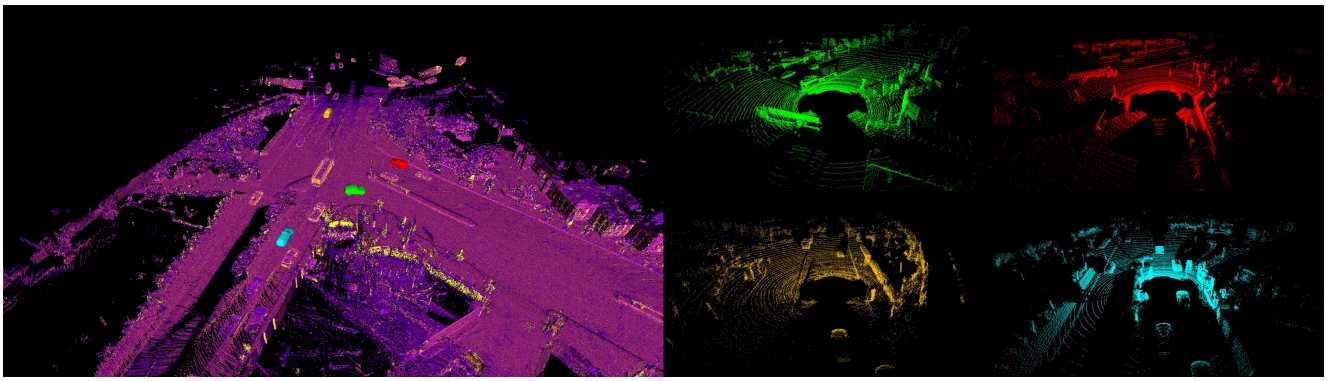

Figure 1: **Data collection of V2V-sim**: (left) Virtual 3D scene (right) LiDAR measurements for each self-driving vehicle present in the scene. Note that the red LiDAR point cloud represents the red vehicle on the scene depicted on the left. Same for all other colors.

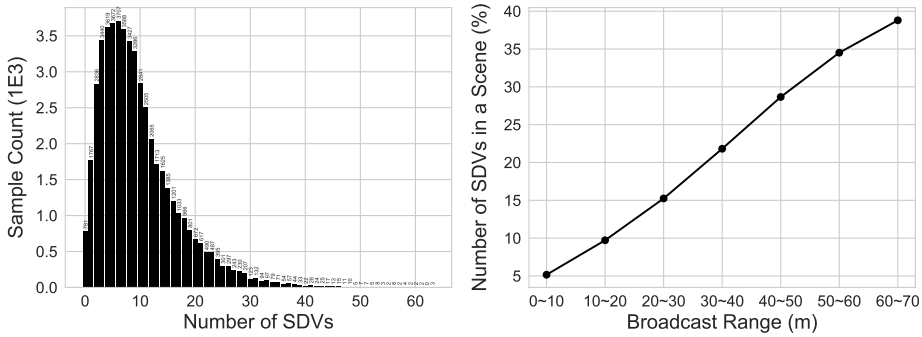

Figure 2: **Statistics of our V2V dataset**: (left) histogram of the number of candidate vehicles. (Right) percentage of SDVs among all vehicles in the scene as a function of distance from the ego-vehicle.

Method	AP@IoU ↑		L2 Error (m) ↓			TCR ↓
	0.5	0.7	1.0s	2.0s	3.0s	τ =0.01
Vanilla GNN	86.6	80.7	0.37	0.60	0.89	3.33
Our Spatially-transformed Messages	**93.1**	**89.9**	**0.29**	**0.50**	**0.78**	**2.25**

Table 1: Effectiveness of adopting spatially-transformed messages in GNN.

entropy model is constructed as a learnable cumulative distribution function with a neural network formulated in a way to satisfy the monotonicity constraint. See Appendix 6.1 in [2].

Cross-vehicle Aggregation: We first apply a CNN with two 3x3 convolution filters with group normalization [5] to compensate for time delay in the received intermediate representation. In the graph neural network, we use one 3x3 convolution filter in the $ConvGRU$ to update the node feature and one 1x1 convolution filter in the MLP to produce per-pixel updated intermediate representations. We construct star-like graphs centered at the main ego-vehicle during training to save memory and construct fully-connected graphs during testing.

Output Network: We send the updated intermediate feature representation to four Inception-like [4] convolutional blocks, where we extract multi-scale features and pass information from all scales to features at each scale. The feature representation that captures multi-scale context is then sent to detection and prediction network branches, each of which is modeled by four convolution filters with group normalization.

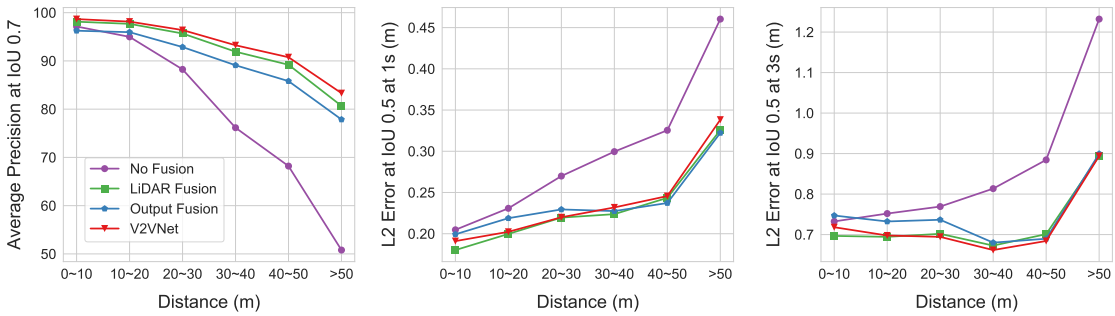

Figure 3: Performance as a function of distance to the ego-vehicle.

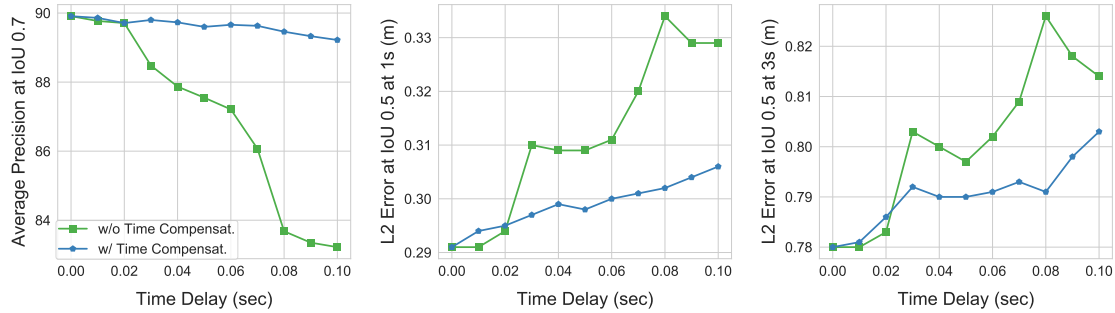

Figure 4: Effectivenss of time delay module.

Graph Topology	AP@IoU ↑		L2 Error (m) ↓			TCR ↓
	0.5	0.7	1.0s	2.0s	3.0s	τ =0.01
Star-like	93.0	89.7	0.30	0.51	**0.78**	2.28
Fully-connected	**93.1**	**89.9**	**0.29**	**0.50**	**0.78**	**2.25**

Table 2: Performance of different graph topologies.

3. Experiments

Performance vs. Object Distance: In Fig. 3, we show performance as a function of the actor's distance to the ego-vehicle. We can see that all V2V methods provide a large performance boost on distant objects. This is expected since LiDAR sensors have sparser measurements on far-away objects. Other SDVs may be potentially closer to objects that are far away from the ego-vehicle and are able to provide better information through V2V communication.

Spatially-transformed Message: In Table. 1, we demonstrate the effectiveness of adopting spatial transformation in message computation. A straightforward approach to incorporate multi-view data with a graph neural network (GNN) is to first spatially align data from multiple views to a single canonical view, then construct a graph with all spatially-aligned data as node features, and finally perform message passing (while still considering overlapping field of views with masks obtained from the spatial transformation). We refer this method to *Vanilla GNN*. However, this approach fails to preserve the local structure of multi-view data in their ego-centric space since spatial warping often induces subpixel error and convolution operator is not rotation-invariant, i.e., performing convolution on a rotated feature map is different from rotating a feature map after convolution. Besides, performing spatial transformation in node initialization stage discards all areas with non-overlapping field of view, which may provide useful information at the boundaries (e.g., padding in convolution). We can see that the introduction of spatially-transformed messages in GNN brings substantial improvement in comparison to its vanilla baseline.

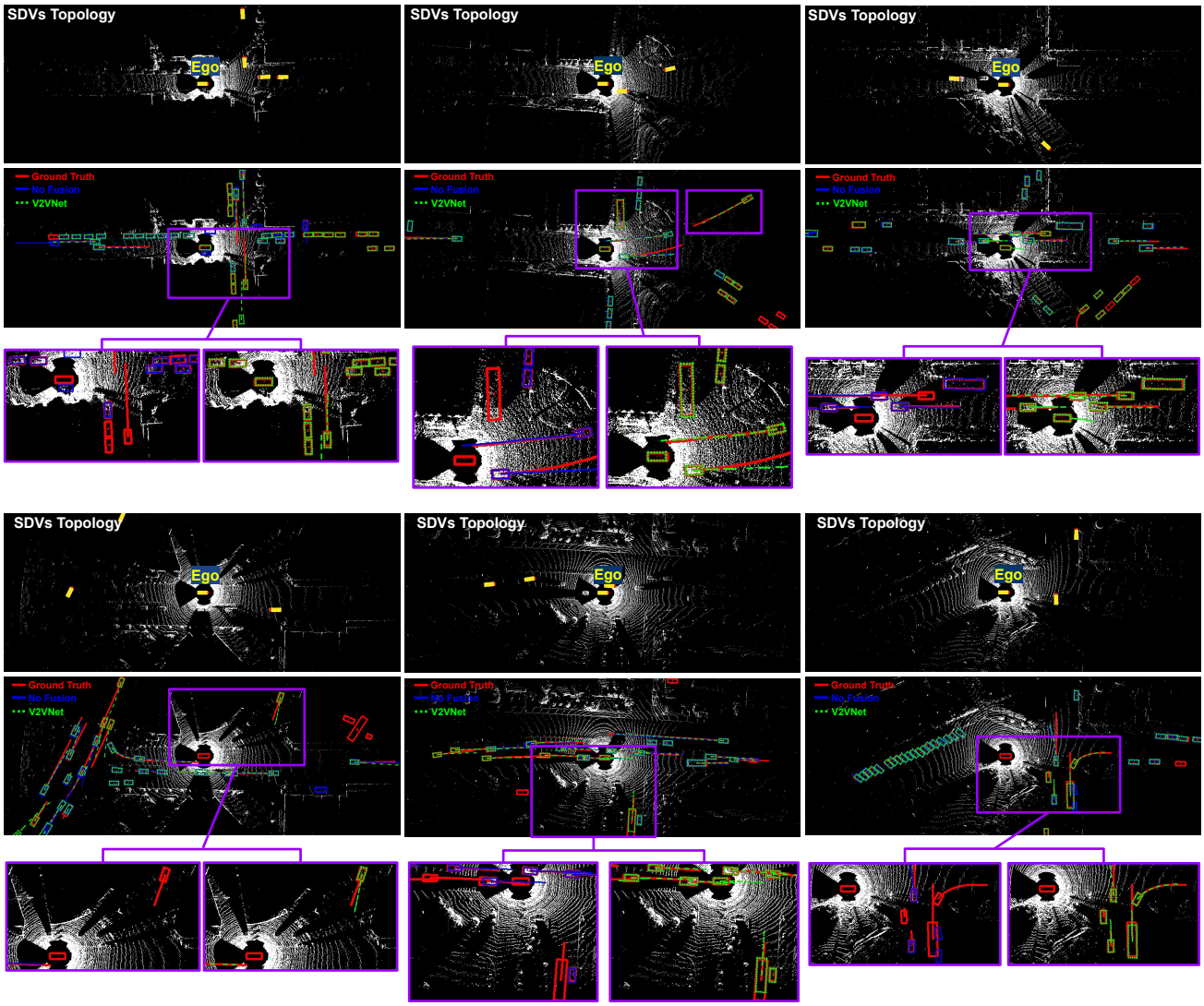

Figure 5: Qualitative comparison between *No Fusion* and *V2VNet* showcasing how V2V communication helps safety-critical cases in P&P.

Time Delay Module: In Fig. 4, we compare perception and prediction performance of V2VNet with and without the module that compensates time delay between the SDVs. While the performance gradually decreases with larger time delay for both approaches, the time delay module of V2VNet compensates for time delay in the feature map and it is thus much more robust.

Graph Topology: In Table. 2, we show how graph topology affects the performance of perception and prediction task. We compare *Star-like Graph*, where the receiving vehicle has edges connected to all nearby SDVs, and *Fully-connected Graph*, where each SDV has edges connected to all other SDVs. Fully-connected graph allows more efficient message propagation but requires larger memory, i.e., $\frac{N(N-1)}{2}$ edges while star-like graph required $N-1$ edges, where N is the number of SDVs. We can see that fully-connected graph provides slightly better results. Note that our model is trained with star-like graph.

Qualitative Analysis: In Fig. 5, we compare *No Fusion* and *V2VNet* and highlight how V2V communication can help P&P in some safety-critical cases, including detecting objects occluded at intersections and detecting objects very close to the ego-car. In Fig. 6, we compare *LiDAR Fusion* and *V2VNet*. *LiDAR Fusion* fails to produce accurate detection on objects with

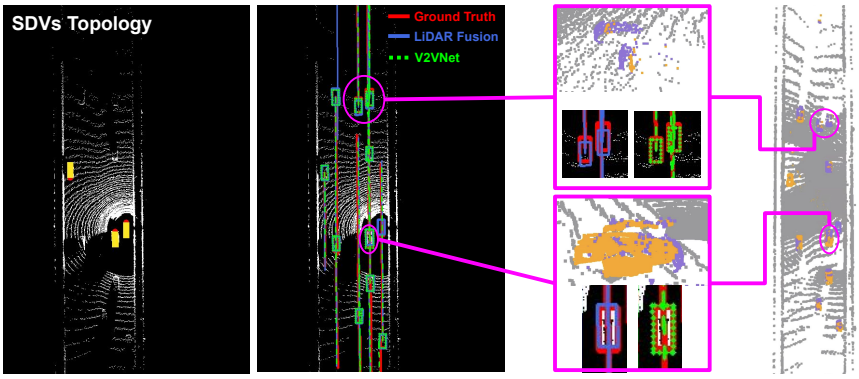

Figure 6: Qualitative comparison between *LiDAR Fusion* and *V2VNet*. The rightmost figure shows the aggregated point cloud from the main ego-vehicle (orange) and another SDV (purple). We paint background point cloud in gray for clarity.

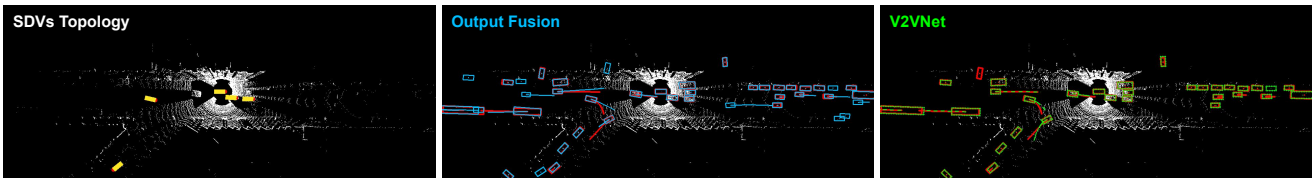

Figure 7: Qualitative comparison between *Output Fusion* and *V2VNet*.

misaligned measurement across SDVs. The rolling shutter effect causes LiDAR sweeps from different viewpoints to observe the same object at different times, resulting in the misaligned point cloud in the figure. In Fig. 7, we compare *Output Fusion* and *V2VNet*. We can see that *Output Fusion* tends to produce more false positives.

References

[1] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. Density modeling of images using a generalized normalization transformation. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. 1

[2] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. In *International Conference on Learning Representations*, 2018. 1, 2

[3] Sivabalan Manivasagam, Shenlong Wang, Kelvin Wong, Wenyuan Zeng, Mikita Sazanovich, Shuhan Tan, Bin Yang, Wei-Chiu Ma, and Raquel Urtasun. Lidarsim: Realistic lidar simulation by leveraging the real world. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1

[4] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 2

[5] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–19, 2018. 2