Semantic-Guided Radar-Vision Fusion for Depth Estimation and Object Detection

Wei-Yu Lee
WeiYu.Lee@UGent.be
Ljubomir Jovanov
Ljubomir.Jovanov@UGent.be
Wilfried Philips
Wilfried.Philips@UGent.be

TELIN-IPI
Ghent University-imec
Gent, Belgium

1 Network Architecture

In our depth estimation network, we follow [6] to use two independent ResNet-18 [3] with a late fusion scheme to build the encoder. The extracted features are fused by concatenation. In addition, we also use UpProj [5] to build the decoder. A task identity layer t is manually appended in the latent space before fed into the decoder. Furthermore, we apply softmax and sigmoid as the output layer activation function according to the different task identity t.

In Table 1, we list the network architecture of the feature extractor F_i and F_d in our proposed FusionYOLO, which is based on the Darknet-53 [7]. We follow the three different detection scales of YOLOv3 [7] to fuse the depth features. Specifically, after the residual connections listed in Table 1, we append the feature maps from F_d to F_i. And then, we pass the combined features to the detection network, which is as same as YOLOv3 [7], to perform multi-scale object detection.

2 Implementation Details

Semantic segmentation pseudo-labels In order to supervised the semantic segmentation learning, we use an external pre-trained model [8] to generate pseudo-labels. We choose the model pre-trained on Cityscapes [2] that labels 19 semantic classes across 5000 high resolution images, which is similar to our evaluation dataset nuScenes [1]. Please note that this external network is only used for providing pseudo-labels, and it is not involved in our model's training process.

Dataset and training details Our models are trained using Pytorch on a Nvidia Geforce 2080Ti and a Nvidia TITAN RTX with mixed precision. We use Stochastic Gradient Descent (SGD) for our optimizer, with a batch size 16, learning rate of 0.001 and a momentum of 0.9 for 100 epochs. In order to fairly compare with the state-of-the-art method [6], we follow their settings to downscale the images to 450×800 resolution to train our depth estimation

Table 1: Network architecture of the feature extractor F_i and F_d. The depth and RGB features are concatenated after the residual connections at three different scales. And then, we pass the combined features to the detection network, which is as same as YOLOv3 [7], to perform multi-scale object detection.

Layer type (F_i / F_d)	Filters	Size / Stride	Repeat	Fusion scale
Conv	32	3×3 / 1	1	
Conv	64	3×3 / 2	1	
Conv Conv Residual	32 64	1×1 / 1 3×3 / 1	1	
Conv	128	3×3 / 2	1	
Conv Conv Residual	64 128	1×1 / 1 3×3 / 1	2	
Conv	256	3×3 / 2	1	
Conv Conv Residual	128 256	1×1 / 1 3×3 / 1	8	
Concat / Output				**1st scale**
Conv	512	3×3 / 2	1	
Conv Conv Residual	256 512	1×1 / 1 3×3 / 1	8	
Concat / Output				**2nd scale**
Conv	1024	3×3 / 2	1	
Conv Conv Residual	512 1024	1×1 / 1 3×3 / 1	4	
Concat / Output				**3rd scale**

model, and re-scale back to original 900×1600 resolution for object detection model. The radar point-clouds is also projected to the 2D image plane with 450×800 resolution as the input x_{radar}. We accumulate the radar points from 3 nearest timestamps to increase the point counts. All of the models in our method are implemented based on the codes provided by [6] and [4]. For more details of the sensor setup, calibration, and synchronization, please refer to the official site [1].

3 Experiments

3.1 Depth Estimation

Day-Night experiment In this section, we provide the day-night experiment results. As we can see in Table 2, we separate the validation data of nuScenes [1] to do the evaluation and compare with the state-of-the-art model [6]. Even during the nighttime with the semantic segmentation loss, our method still can estimate depth maps with comparable performance,

Table 2: Day-night performance comparison.

Methods	$\delta_1 \uparrow$	$\delta_2 \uparrow$	RMSE $\downarrow$	MAE $\downarrow$	REL $\downarrow$	$\text{MAE}_{log} \downarrow$
Daytime experiments						
Lin et al. (single-stage) [5]	0.894	0.957	5.271	2.157	0.107	0.043
Lin et al. (two-stage) [5]	0.901	0.962	5.030	1.941	0.095	0.038
Ours	**0.902**	**0.965**	**5.028**	**1.934**	**0.093**	**0.035**
Nighttime experiments						
Lin et al. (single-stage) [5]	0.814	0.925	6.402	3.096	0.147	0.060
Lin et al. (two-stage) [5]	**0.832**	**0.932**	**6.290**	**2.933**	**0.135**	**0.056**
Ours	0.823	0.929	6.324	3.004	0.142	0.058

Table 3: Per-class object detection performance comparison.

Methods	Input data	person	bicycle	car	motorcycle	bus	truck	all
YOLOv3 [7]	RGB	25.32	20.44	52.91	23.52	49.43	36.34	34.69
FusionYOLO w/ [5] depth maps	RGB + Radar	25.91	21.85	53.44	24.31	50.12	37.11	35.54
FusionYOLO w/ our depth maps	RGB + Radar	**26.37**	**22.54**	**53.47**	**24.45**	**51.60**	**37.47**	**35.95**

which can be attributed to the contribution of our proposed confidence map ϕ. It not only suppresses the false positives, but also controls the smoothing strength according to whether the objects is recognized by the model or not.

Qualitative Results In this part, we provide more qualitative results of our method, and also show the results of [5] for comparison. In Figure 1, we observe that our model produces excellent results with clear edges and less false positives. Even when the occlusions cause the semantic information loss, our method still can estimate depth maps of decent quality.

3.2 Object Detection

Per-class performance In order to further analysis the details of the object detection performance, we use the state-of-the-art model's depth maps and our depth maps to train FusionYOLO and compare with baseline YOLOv3 [7] to see the differences on per-class AP results. As we expected, in Table 3, the results show that our method outperforms the others in each class. Especially the person, bicycle, and truck class, our method achieve better performance with a wider margin. In contrast, we can observe that the car class shows less performance gap. The RGB baseline model has already achieved high performance, and the introduced depth information only can slightly improve the results.

Multi-scale fusion methods In order to find a proper method to combine the features from depth maps and RGB images during performing object detection, we consider different fusion methods in this experiment to compare their differences. The methods are including element-wised multiplication, element-wised addition, and feature concatenation. Because the feature extractor F_i and F_d use the same network architecture, the feature maps at the three different scales (please refer to Chapter 1) have the same dimensions. For element-wised addition and multiplication, we simply add or multiply the values of each element in

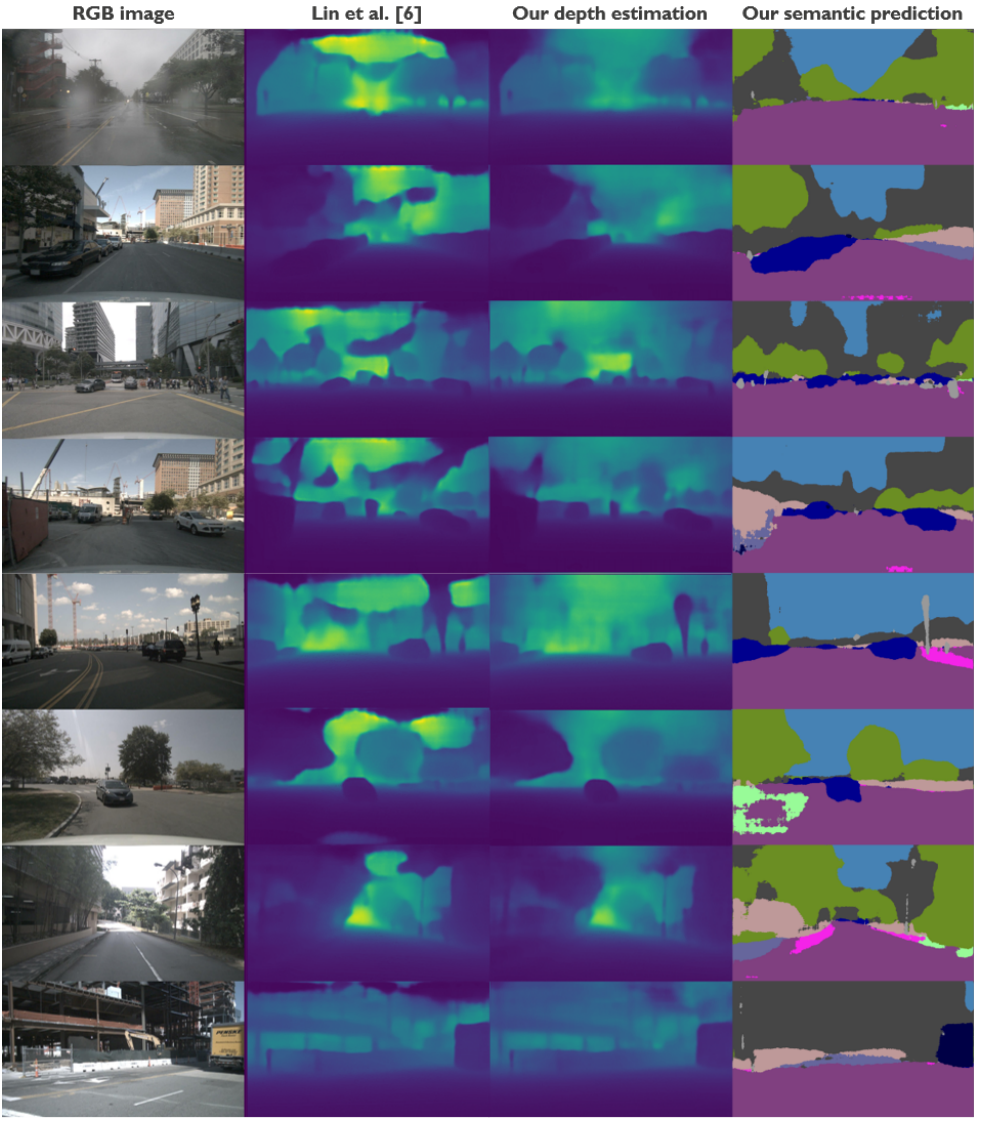

Figure 1: Qualitative results comparison on nuScenes dataset.

the feature maps, and generate the outputs with the same size. For feature concatenation, we append the depth features to the RGB features, which doubles the number of the feature maps and keeps the same height and width. The results show that feature concatenation achieve the best performance and outperforms the other two methods with a wide margin.

Different feature extractor designs In this section, we discuss the design of the feature extractor F_d in FusionYOLO. Instead of simply replacing the Darknet-53 [7] with other existing network architectures to compare the performance, we conducted several experiments to verify the feasibility of some common design directions, including using the same ar-

Table 4: Multi-scale fusion methods performance comparison.

Methods	AP	AP_{50}	AP_{75}
Element-wised multiplication	33.45	63.48	33.84
Element-wised addition	33.12	63.17	33.19
Feature concatenation	**35.95**	**65.77**	**35.84**

Table 5: Different feature extractor F_d designs comparison.

Methods	AP	AP_{50}	AP_{75}
Shared extractor	32.14	57.12	31.15
Shallow Darknet	34.75	64.23	33.84
Same as F_i	**35.95**	**65.77**	**35.84**

chitecture of F_i, sharing extractor with F_i and a shallow network version. For the shared extractor, we tried to use the same network to extract the features from RGB images and depth maps at the same iteration. Specifically, F_i and F_d share the same model parameters, and fuse the features at three different scales with element-wised multiplication. However, due to the RGB images and depth maps having the different meanings of a single pixel (RGB values and distance), it is not straightforward to directly learn the two types of features by using the same network. For the shallow network design, we simply reduce the number of residual blocks to its half size in the Darknet-53 [7] to make a shallow extractor. Nonetheless, the unbalanced size of extractors are unstable and easily over-fitting in our case. The results show that using the same architecture of F_i can achieve best performance.

References

[1] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[2] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[4] Glenn Jocher, Yonghye Kwon, guigarfr, perry0418, Josh Veitch-Michaelis, Ttayu, Daniel Suess, Fatih Baltacı, Gabriel Bianconi, IlyaOvodov, Marc, e96031413, Chang Lee, Dustin Kendall, Falak, Francisco Reveriano, FuLin, GoogleWiki, Jason Nataprawira, Jeremy Hu, LinCoce, LukeAI, NanoCode012, NirZarrabi, Oulbacha Reda, Piotr Skalski, SergioSanchezMontesUAM, Shiwei Song, Thomas Havlik, and Timothy M. Shead. ultralytics/yolov3: v9.5.0 - YOLOv5 v5.0 release compatibility update for YOLOv3, April 2021. URL https://doi.org/10.5281/zenodo.4681234.

[5] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth international conference on 3D vision (3DV)*, 2016.

[6] Juan-Ting Lin, Dengxin Dai, and Luc Van Gool. Depth estimation from monocular images and sparse radar data. In *International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[7] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[8] Andrew Tao, Karan Sapra, and Bryan Catanzaro. Hierarchical multi-scale attention for semantic segmentation. *arXiv preprint arXiv:2005.10821*, 2020.