DrivingStereo: A Large-Scale Dataset for Stereo Matching in Autonomous Driving Scenarios –Supplementary Material

Guorun Yang[1⋆] Xiao Song[2⋆] Chaoqin Huang[2,3] Zhidong Deng[1] Jianping Shi[2] Bolei Zhou[4]
[1]Department of Computer Science and Technology, Tsinghua University[†]
[2]SenseTime Group Limited [3]Shanghai Jiao Tong University [4]The Chinese University of Hong Kong
{ygr13@mails, michael@mail}.tsinghua.edu.cn huangchaoqin@sjtu.edu.cn
{songxiao, shijianping}@sensetime.com bzhou@ie.cuhk.edu.hk

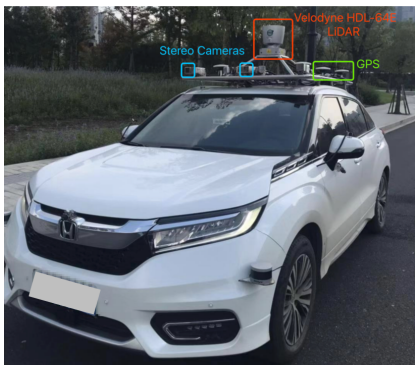


Figure 1. Data acquisition vehicle with sensors.

We provide more details of our model, datasets, and experiments in this supplementary material. Firstly, we describe the method of calibrations. Secondly, we give a structural definition of GuideNet. Thirdly, more details of our DrivingStereo dataset are described, including the information of collected sequences, segmentation examples, and qualitative examples of disparity prediction.

1. Calibration

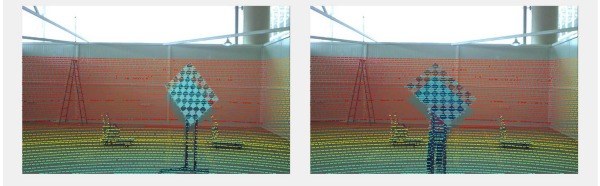
Our data acquisition vehicle and mounted sensors are shown in Fig. 1. For the calibration of cameras and rectification of stereo pairs, we use MATLAB Toolbox to compute related parameters. For the joint calibration of LiDAR and camera, we adopt the PnP method [6], where the corners of checkboard are located in the two systems. In Fig. 2, the LiDAR points are accurately projected onto images, which shows our quality.

⋆ indicates equal contribution.

† State Key Laboratory of Intelligent Technology and Systems, Beijing National Research Center for Information Science and Technology, and Center for Intelligent Connected Vehicles and Transportation.

2. GuideNet

The GuideNet follows the encoder-decoder architecture as [7, 4]. We utilize the ResNet [5] as the backbone and plug in the single-direction correlation module [8] to compute matching cost between pair or image features. As shown in Tab. 1, the GuideNet can be divided into three parts: feature extractor, feature aggregator and disparity encoder-decoder. The feature extractor is the shallow part of ResNet-50 [5] whose weights are pretrained on ImageNet dataset [3]. A correlation layer [8] is used to compute matching cost on extracted features. Here, both max displacement and padding size in correlation layer are set to 32. Then the feature aggregator concatenates the left feature and cost volumes together. The disparity encoder contains eight residual blocks and one convolutional block to learn disparity features. Most of the residual blocks adopt dilated pattern as [1] to expand the receptive field. In order to recover the original resolution of the input image, three deconvolutional blocks and one single-channel convolutional layer are appended to regress the disparity map.



(a) Indoor examples

(b) Outdoor examples

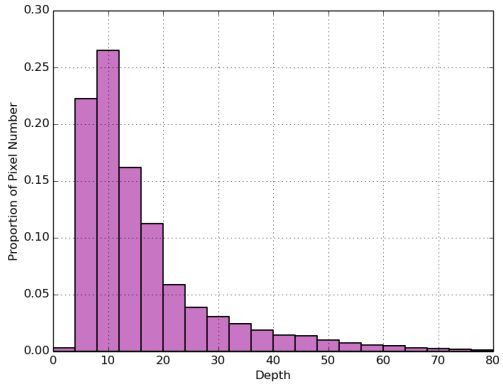
Figure 2. Examples of joint calibration

Our GuideNet enables end-to-end disparity learning with high speed and accuracy. We implement the model in PyTorch and train it with one NVIDIA 1080 Ti. The inference time is as small as 12ms so that the model is possible to be deployed in real-time applications of autonomous driving. In this paper, the main goal of GuideNet is to perform guided filtering on the original disparity map projected from multi-frame fused LiDAR point clouds.

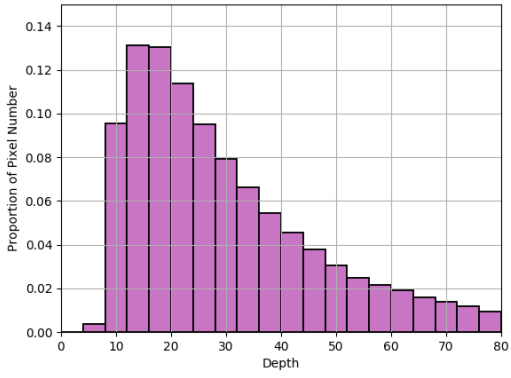
3. DrivingStereo

3.1. Sequences

We collect 42 sequences from July to October in 2018. As shown in Tab. 2, the total number of frames exceeds 180k. To avoid repeated samples, we sample 1 frame from every 5 frames. All static fragments are removed according to the GPS information. Finally, we obtain $182,188$ frames to construct our DrivingStereo dataset. Among the 42 sequences, we select 38 as training dataset, and the remaining 4 sequences are split as testing dataset. Though the mounted sensors on the vehicle are calibrated for every sequence, it still remains errors on the projected images. As a result, for testing split, we manually select $7,751$ high-quality images. In Fig. 3, we also analyze the depth distribution of KITTI Stereo 2015 and our DrivingStereo. Because more forward frames are fused, our disparity maps have farther pixels compared with KITTI Stereo [9], which helps improve the evaluation of distance-aware metrics.



(a) KITTI Stereo 2015 [9]



(b) Our dataset

Figure 3. Depth distribution of KITTI Stereo 2015 [9] and our DrivingStereo.

3.2. Segmentation Examples

In order to generate semantic labels, we employ the PSPNet [10] which is pretrained on Cityscapes [2] to produce primary segmentation results. Since the primary results are not accurate, we adopt a free-space segmentation model to rectify ground labels. A 2D detector is used to localize vehicles and humans to modify related labels. Concretely, the primary labels on vehicles and humans are validated by the bounding boxes predicted from the detector, where the ambiguous labels are masked as zero. We show several qualitative examples in Fig. 4. It is worth noted that the labels in rectangle boxes of vehicles or humans but outside of the instances are marked as zero to reduce possible errors. Though the pixels of vegetation, construction, and others are not exactly accurate, it enables us to evaluate the general performance by semantic-aware metrics on our dataset.

Figure 4. Qualitative segmentation examples in DrivingStereo.

3.3. Qualitative Examples

In Fig. 5, we provide several qualitative examples of disparity maps predicted by our GuideNet. The GuideNet is able to estimate reasonable disparity results in challenging scenarios.

Table 1. Layer-wise structure of GuideNet. The "conv block" indicates the convolutional block, where a convolutional layer is followed by batch normalization and ReLU activation. The "res block" denotes the residual block presented in [5]. The "corr 1D" stands for single-direction correlation [8]. The "deconv block" represents the deconvolutional block that contains a deconvolutional layer, batch normalization, and ReLU activation.

Layer	Attribute	Channels I/O	Scaling	Input
		1. Feature Extractor		
conv_block1_1	kernel size = 3, stride = 2	3 / 64	1/2	input stereo images
conv_block1_2	kernel size = 3, stride = 1	64 / 64	1/2	conv_block1_1
conv_block1_3	kernel size = 3, stride = 1	64 / 128	1/2	conv_block1_2
max_pooling	kernel size = 3, stride = 2	128 / 128	1/4	conv_block1_3
res_block2_1	kernel size = 3, stride = 1	128 / 256	1/4	max_pool_block1
res_block2_2	kernel size = 3, stride = 1	256 / 256	1/4	res_block2_1
res_block2_3	kernel size = 3, stride = 1	256 / 256	1/4	res_block2_2
res_block3_1	kernel size = 3, stride = 1	512 / 512	1/8	res_block2_3
		2. Feature Aggregator		
conv_block_pre	kernel size = 3, stride = 1	512 / 128	1/8	res_block3_1
corr_1d	max displacement = 32, single direction [8]	256 / 33	1/8	conv_block_pre
conv_block_trans	kernel size = 3, stride = 1	128 / 128	1/8	conv_block_pre
concat	aggregate corr_1d and conv_block3_1	(128 + 33) / 161	1/8	corr_1d, conv_block_trans
		3. Disparity Encoder-Decoder		
res_block3_2	kernel size = 3, stride = 1	161 / 128	1/8	concat
res_block3_3	kernel size = 3, stride = 1	128 / 128	1/8	res_block3_2
res_block4_1	kernel size = 3, stride = 1, dilated pattern	128 / 256	1/8	res_block3_3
res_block4_2	kernel size = 3, stride = 1, dilated pattern	256 / 256	1/8	res_block4_1
res_block4_3	kernel size = 3, stride = 1, dilated pattern	256 / 256	1/8	res_block4_2
res_block4_4	kernel size = 3, stride = 1, dilated pattern	256 / 256	1/8	res_block4_3
res_block5_1	kernel size = 3, stride = 1, dilated pattern	256 / 512	1/8	res_block4_4
res_block5_2	kernel size = 3, stride = 1, dilated pattern	512 / 512	1/8	res_block5_1
conv_block5_3	kernel size = 3, stride = 1	512 / 128	1/8	res_block5_2
deconv_block1	kernel size = 3, stride = 2	128 / 64	1/4	conv_block5_3
deconv_block2	kernel size = 3, stride = 2	64 / 32	1/2	deconv_block1
deconv_block3	kernel size = 3, stride = 2	32 / 16	1	deconv_block2
disp_conv	kernel size = 3, stride = 1	16 / 1	1	deconv_block3

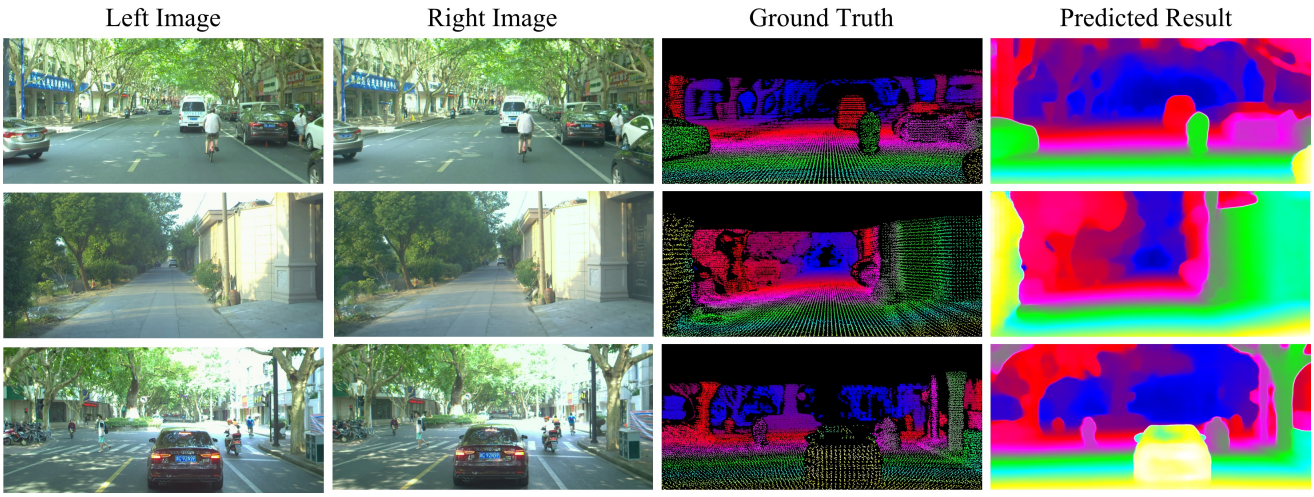


Figure 5. Qualitative examples in the coarse set where calibration errors exist.

Table 2. Details of the collected sequences. We provide the collected time, split set, number of frames, and attributes for each sequence.

Sequence		Split	Frames		Attributes
Date	Time		Original	Selected	
2018-07-09	16:11	Train	23,606	2,776	Urban, sunny
2018-07-10	09:54	Train	23,231	1,138	Urban, sunny
2018-07-11	14:48	Test	35,424	1,721	Urban, suburban, cloudy
2018-07-16	15:18	Train	9,789	1,036	Urban, suburban, sunny
2018-07-16	15:37	Train	32,988	5,080	Urban, suburban, sunny, cloudy
2018-07-18	10:16	Train	33,784	4,639	Urban, sunny
2018-07-18	11:25	Train	29,185	3,626	Urban, sunny
2018-07-24	14:31	Train	32,136	4,721	Urban, cloudy
2018-07-27	11:39	Train	16,679	1,522	Urban, cloudy
2018-07-31	11:07	Train	7,461	1,355	Urban, cloudy
2018-07-31	11:22	Train	6,000	811	Urban, cloudy
2018-08-01	11:13	Test	30,591	1,532	Elevated, highway, cloudy
2018-08-07	13:46	Test	109,475	2,861	Suburban, country road, sunny, cloudy
2018-08-13	15:32	Train	24,349	1,052	Urban, suburban, sunny, cloudy
2018-08-13	17:45	Train	10,739	802	Urban, suburban, sunny
2018-08-17	09:45	Train	28,740	1,667	Urban, suburban, rainy, foggy
2018-10-10	07:51	Train	34,812	4,514	Urban, suburban, cloudy
2018-10-11	16:03	Test	33,327	1,637	Suburban, country road, sunny, cloudy
2018-10-11	17:08	Train	7,471	1,119	Suburban, country road, dusky
2018-10-12	07:57	Train	33,284	5,477	Urban, suburban, cloudy
2018-10-15	11:43	Train	32,553	4,950	Country road, cloudy
2018-10-16	07:40	Train	32,248	4,651	Urban, suburban, country road, cloudy
2018-10-16	11:13	Train	16,287	2,315	Urban, suburban, country road, cloudy
2018-10-16	11:43	Train	18,530	2,768	Urban, suburban, country road, cloudy
2018-10-17	14:35	Train	30,786	4,437	Urban, suburban, sunny, foggy
2018-10-17	15:38	Train	34,704	4,625	Urban, suburban, sunny, cloudy
2018-10-18	10:39	Train	31,679	4,670	Urban, suburban, cloudy
2018-10-18	15:04	Train	32,711	5,128	Urban, suburban, sunny, cloudy
2018-10-19	09:30	Train	31,927	7,565	Urban, suburban, sunny
2018-10-19	10:33	Train	34,625	7,937	Suburban, sunny, cloudy
2018-10-22	10:44	Train	32,294	7,808	Suburban, cloudy, foggy
2018-10-23	08:34	Train	33,464	7,842	Urban, suburban, cloudy
2018-10-23	13:59	Train	34,607	6,064	Urban, sunny, cloudy
2018-10-23	15:06	Train	31,185	6,543	Urban, cloudy
2018-10-24	11:01	Train	29,922	6,870	Urban, country road, sunny, cloudy
2018-10-24	14:13	Train	30,911	6,665	Urban, sunny, cloudy
2018-10-25	07:37	Train	33,243	6,905	Urban, sunny, foggy
2018-10-26	15:24	Train	29,747	7,128	Urban, suburban, sunny
2018-10-27	08:54	Train	33,286	7,945	Suburban, sunny
2018-10-27	10:02	Train	31,679	6,672	Suburban, sunny
2018-10-30	13:45	Train	30,383	6,057	Urban,sunny, cloudy
2018-10-31	06:55	Train	31,707	7,557	Suburban, cloudy
Sequences		Total: 42 train: 38 test: 4			
Frames		Total: 182,188 train: 174,437 test: 7,751			

References

[1] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 2016. 1

[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 *CVPR*, 2016. 2

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 1

[4] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *ICCV*, 2015. 1

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1, 3

[6] Joel A Hesch and Stergios I Roumeliotis. A direct least-squares (dls) method for pnp. In *ICCV*, 2011. 1

[7] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1

[8] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016. 1, 3

[9] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *CVPR*, 2015. 2

[10] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017. 2