Hierarchical deep stereo matching on high-resolution images Supplementary Material

Gengshan Yang[1], Joshua Manela[2], Michael Happold[2], Deva Ramanan[1,2]
[1]Carnegie Mellon University, [2]Argo AI
gengshany@cmu.edu {jmanela, mhappold, dramanan}@argo.ai

A. Data

A.1. High-resolution Virtual Stereo (HR-VS) Dataset

HR-VS dataset contains 780 pairs of rectified **high-resolution** image pairs and ground-truth collected using Carla [3]. It also has several desirable properties as follows:

1. It captures detailed structures in outdoor driving scenarios. A qualitative example is shown in Fig. 1.
2. It contains a variety of realistic images. In Carla, the RGB images are post-processed by default with several camera effects, including "Vignette", "Grain jitter", "Bloom", "Auto exposure", "Lens flares" and "Depth of field", to make them realistic. We visualize 400 out of 780 reference images in HR-VS dataset as shown in Fig. 2.
3. The disparity range is large (from 9.66px to 768px). A histogram of ground-truth disparity is shown in Fig. 3.

A.2. High-resolution Real Stereo (HR-RS) benchmark

HR-RS contains 33 pairs of rectified **real hign-resolution** image pairs with sparse ground-truth collected using LiDAR sensor. Some sample images and groungtruth are shown in Fig. 4. A histogram of ground-truth disparity is included in Fig. 3.

A.3. Profile of stopping distance versus speed

The profile of stopping distance versus speed is shown in Fig. 5, where The safe stopping distance is calculated as follows:

$$d = vt_{lat} + \frac{v^2}{2gmin(a_{dec}, \mu)}$$

where d is the stopping distance, v is the current running speed, t_{lat} represents the latency, g represents the gravity acceleration, a_{dec} represents the decelerating factor and μ is the coefficient of friction. Assuming $t_{lat} = 0.46s$, an aggressive

Figure 1: Example high-resolution rectified image pairs as well as projected ground-truth depth in HR-VS.

Figure 2: Sampled RGB images in HR-VS dataset. It contains a variety of synthesized high-resolution images captured under 4 weather conditions: "ClearNoon", "WetCloudyNoon", "ClearSunset" and "WetCloudySunset".

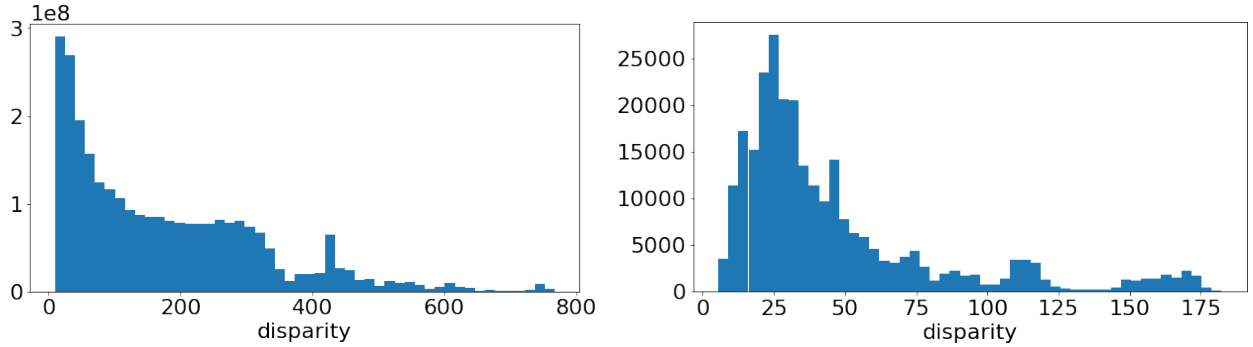

Figure 3: Histogram of disparity ground-truth in HR-VS dataset (left) and HR-RS benchmark (right). The maximum disparity for HR-VS dataset is 768. However for HR-RS the max disparity 182.2px, which is restricted by the sensing range of LiDAR sensors.

Figure 4: Sample high-resolution rectified image pairs as well as ground-truth 3D reconstruction in HR-RS.

deceleration factor $a_{dec} = 0.30$, dry condition friction coefficient $\mu = 0.67$, and plugging in $v = \{25, 45\}mph$, we get $d = \{26.36, 78.01\}m$. If assuming a moderate deceleration factor $a_{dec} = 0.24$, we get $d = \{31.76, 95.20\}m$. If assuming a

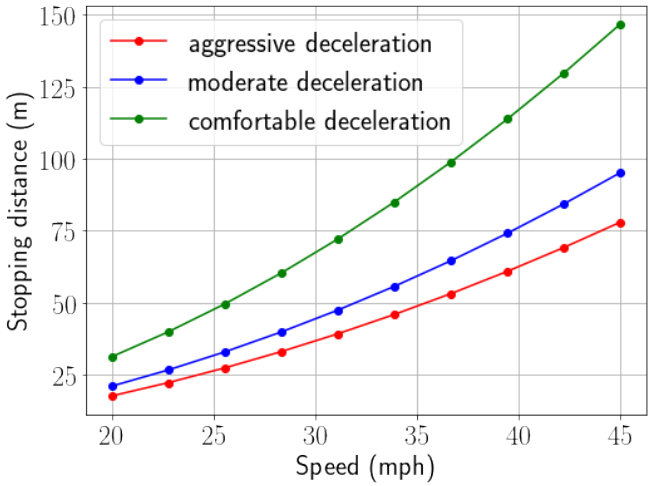

Figure 5: Profile of stopping distance versus speed. To achieve a comfortable deceleration at 45 mph, an ideal sensing range is over 145 meters, which poses challenge on current LiDAR/stereo based depth sensing systems.

comfortable deceleration factor $a_{dec} = 0.15m$, we get $d = \{47.58, 146.76\}m$.

B. Method Details

B.1. Network architecture

We give a detailed network architecture plot in Fig. 6.

B.2. Train & test details

We implemented the HSM network using Pytorch. We train the model using Adam optimizer with batch size of 24 on a machine with 4 Titan X Pascal GPUs, while setting the initial learning rate to 0.001 and betas to $(0.9, 0.999)$. We train for 9 epochs and then shrink the learning rate by 10. During training, we augment Middlebury, KITTI-15, ETH3D [13, 12, 11] and HR-VS to the same size as Sceneflow, resulting in around 170k training samples. We perform both symmetric and asymmetric augmentations on the fly. Asymmetric chromatic augmentations include randomly applying different brightness ($[0.5, 2]$), gamma ($[0.8, 1.2]$) and contrast ($[0.8, 1.2]$) two target and inference images. We apply y-disparity augmentation by uniformly sample a delta y-translation ($[0, 2]$px) and rotation($[0, 0.1]$ deg) at a chance of 0.5. We also apply asymmetric masking at a chance of 0.5 by uniformly sample the width ($[50, 150]$px) and height ($[50, 150]$px) of the mask and randomly placing it on the image. Symmetric augmentations include scaling ($[0.9, 2.4]$ for low-res images and $[0.225, 1.2]$ for high-res images) and randomly cropping to fix-sized (576×768) patches. We set the number of disparities for searching as 768px in training time. In test time, we make predictions at 3 scales from coarse to fine. When testing on Middlebury14 images, we set disparity search range according to maximum disparities in the calibration files, while for upscaled KITTI15 test images, we set disparity search range as 384. For HR-RS images, we set disparity search range as 512.

C. Additional Results

C.1. Results on non-occluded pixels of Middlbury14

Comparison to SOTAs on Middlebury14 **non-occluded** pixels is shown in Table 1. Compared to MC-CNN-acrt on non-occluded pixels, we reduced avgerr by 45.8%, rms by 51.6% and bad-4.0 by 1.6% while running 294.1 times faster. Compared to CBMV_ROB, we reduced avgerr by 27.9%, rms by 35.6% and bad-4.0 by 8.9% while running 7737.3 times faster. We runs 0.31 times slower than iResNet_ROB but reduced avgerr by 54.1%, rms by 25.9% and bad-4.0 by 69.4%.

C.2. Qualitative results

More qualitative results on Middlebury14 benchmark and additional images is shown in Fig. 7 and Fig. 8. We provide all reconstruction videos on Middlebury14 benchmark here. We also provide coarse-to-fine outputs of HSM Network on

Figure 6: Detailed network architecture. 2D/3D Convolutions are defined by (in_channels,out_channels,stride), 2D features are defined by (channels, height,width) and 3D features are defined by (channels, disparity,height,width). All convolution kernels are 3×3 by default. **Left:** Feature pyramid encoder. To extract features with different levels of details, while maintaining the coarse-scale information, we adopt a encoder-decoder architecture with skip connections. Encoder part consists of 3 convolutions extracting low-level features, followed by a pooling, 4 residual blocks with a stride of 2, and an SPP [16] module integrating contextual information. We fuse features before and after SPP by concatenation and apply up-convs (convolution after up-sampling) to gradually goes from coarse to fine with increased feature map scale. To control the feature volume size, we project the features to a lower dimensional subspace. **Right:** Feature volume decoder. We first construct feature volumes at 4 different scales by comparing features of target and reference views at potential matching positions. Then we filter feature volumes from coarse to fine by 4 decoder blocks as described in Section 3.1. The first and second block contains a VPP layer to aggregate contextual information. We also apply 3D up-convolutions (convolution after tri-linear up-sampling) at the end of decoder blocks, the outputs of which are fused with the finer-scale raw feature volumes to gradually increase spatial and disparity dimension resolution. Finally, cost volume predictions are made at each scale and disparity is calculated by differentiable soft argmin [7].

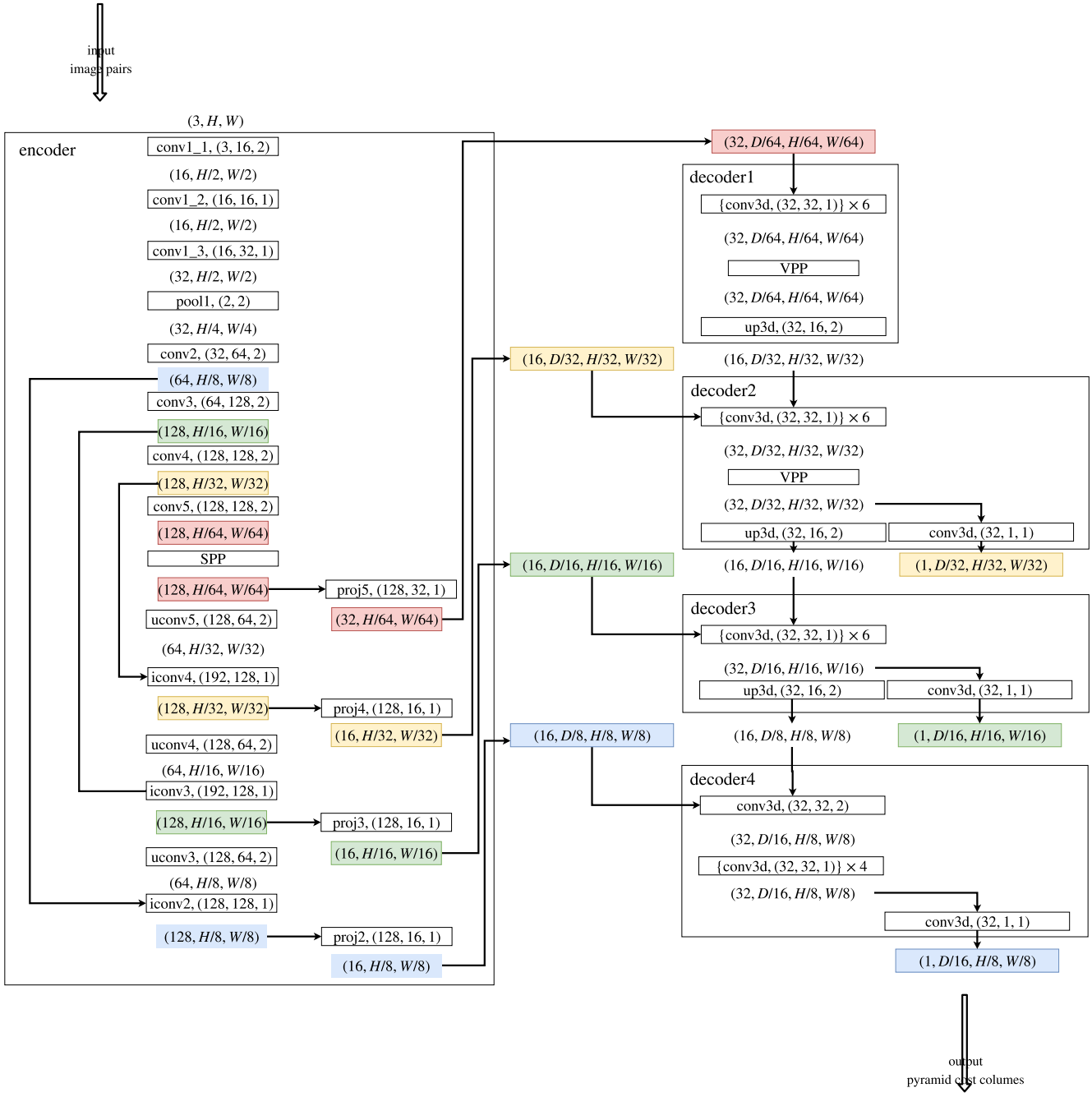

Figure 7: Reconstruction of HSM-F estimation on image Crusade-perfect by projecting disparity to world coordinates.

Figure 8: Reconstruction of HSM-F estimation on image Plants by projecting disparity to world coordinates.

Method	time (s)	avgerr	rms	bad-4.0	bad-2.0	bad-1.0	A99
HSM-F (Ours)	0.51 (0.61)	**2.07**$_2$	**10.3**$_1$	**4.83**$_{15}$	**10.2**$_{23}$	**24.6**$_{33}$	**39.2**$_1$
SGM_ROB [5]	0.32	5.66	25.6	11.6	18.4	31.1	127
iResNet_ROB [10]	0.34 (0.42)	4.51	13.9	15.8	24.8	39.7	65.6
ELAS_ROB [4]	0.48	9.52	28.2	18.9	27.3	45.9	134
IDR [8]	0.49	6.35	21.8	12.7	18.1	29.7	102
PSMNet_ROB [2]	0.64	6.68	19.4	23.5	42.1	63.9	84.5
DN-CSS_ROB [6]	0.66	4.04	13.9	14.7	22.8	36.0	58.8
LPS-H [14]	9.52	14.4	32.1	15.5	19.2	30.9	123
SPS [9]	25.5	11.6	43.0	15.0	19.6	29.1	244
LPS-F [14]	25.8	13.7	31.9	16.6	20.3	27.6	126
MC-CNN-acrt [15]	150	3.82	21.3	4.91	8.08	17.1	92.9
CBMV_ROB [1]	3946	2.87	16.0	5.30	$\underline{7.65}_{14}$	$\underline{15.6}_9$	70.1

Table 1: Results on Middlebury14 benchmark where non-occluded pixels are evaluated. Best results over the "fast" group are bolded, and best results overall are underlined. The subscript number shows the absolute rank among the whole benchmark. While running at 510ms/image on 6 megapixel images, our method out-performed all algorithms running faster under 1000ms/image by a large margin on all metrics. Even over the whole benchmark, we achieved the first place on rms as well as A99, and the second place on avgerr.

Method	avgerr	bad-1.0	bad-2.0*	time (ms)
Full-method	4.01	**46.93**	**26.88**	97
- masking	4.05	48.89	28.20	97
- VPP	**3.88**	47.99	27.73	93

Table 2: Additional diagnostic table for removing individual components.

Middlebury and HR-RS here in videos.

D. More Diagnostics

D.1. Detailed setup

We follow the same training protocol as described in the experiment setup (section 4.1), but train different models with the same pre-trained encoder weights. We train on a single Tesla V100 GPU with a batch size of 8 for 60k iterations. The learning rate is down-scaled by 10 for the last 10k iterations. We then test on half-resolution additional Middlebury images except for "Shopvac", resulting in 24 test images in total.

D.2. More diagnostics

We provide results on removing "asymmetric masking" and "Volumetric Pyramid Pooling (VPP)" as listed in Tab. 2. After removing "asymmetric masking", bad-2.0 error increases by 4.9%. Also, removing VPP results in an increase of bad-2.0 error by 3.2%. Both components are helpful in that they contribute to accuracy without heavily slowing down the speed.

D.3. Qualitative effect of asymmetric augmentations

Qualitative effect of adding asymmetric augmentations are shown in Fig. 9, Fig. 10 and Fig. 11, where we find that after applying asymmetric augmentations, our model becomes more robust to real-world calibration imperfectness, change of imaging conditions and hard-to-match regions.

D.4. Resilience to calibration error

We first compare our HSM-H model to ELAS-H [4] in terms of robustness to calibration error as shown in Tab. 3. We find our model is more robust to calibration errors: When the camera becomes imperfectly calibrated, the average pixel error

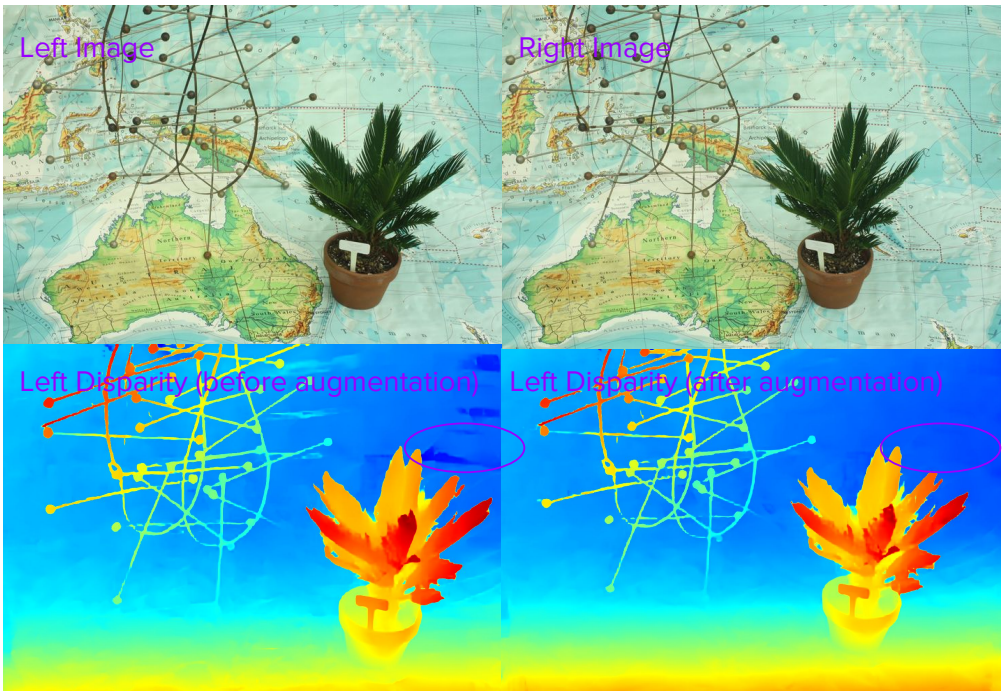

Figure 9: Effect of asymmetric y-disparity augmentation on Middlebury14 test image Australia. After applying asymmetric y-disparity augmentation, the model's ability to handle calibration error get improved in the purple-circled regions.

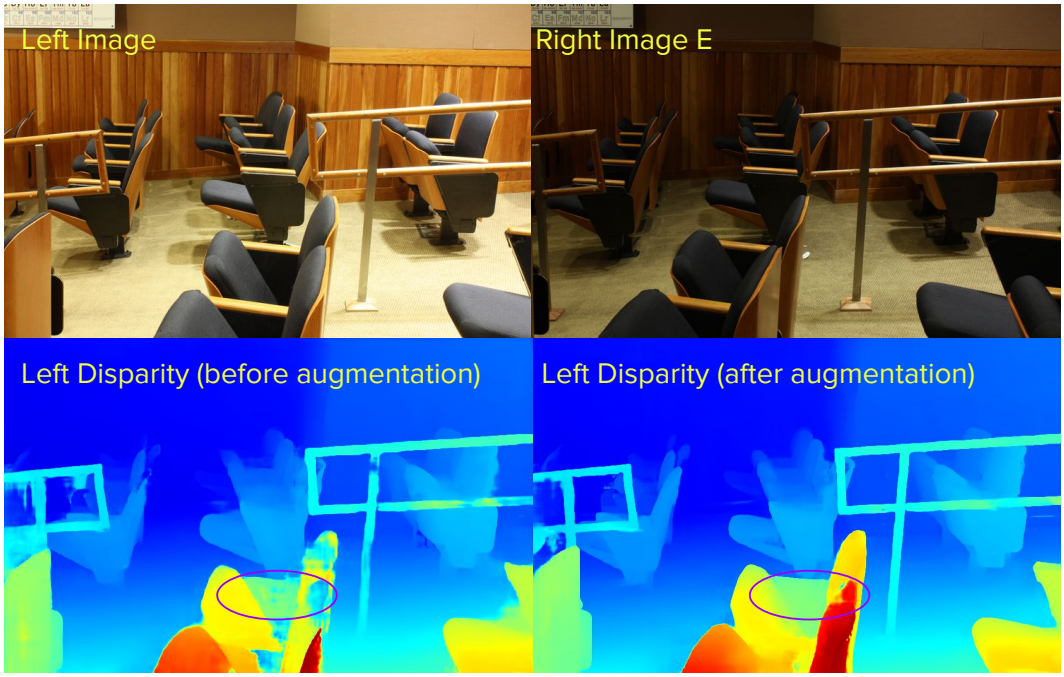

Figure 10: Effect of asymmetric chromatic augmentation on Middlebury14 test image ClassroomE. "E" stands for changed exposure between views. After applying asymmetric chromatic augmentation during training time, the model's ability to handle exposure change get improved in the purple-circled region.

rate of HSM-H only increases 5.9%, while that of ELAS-H increases by 29.6%. The same conclusion can be made in terms

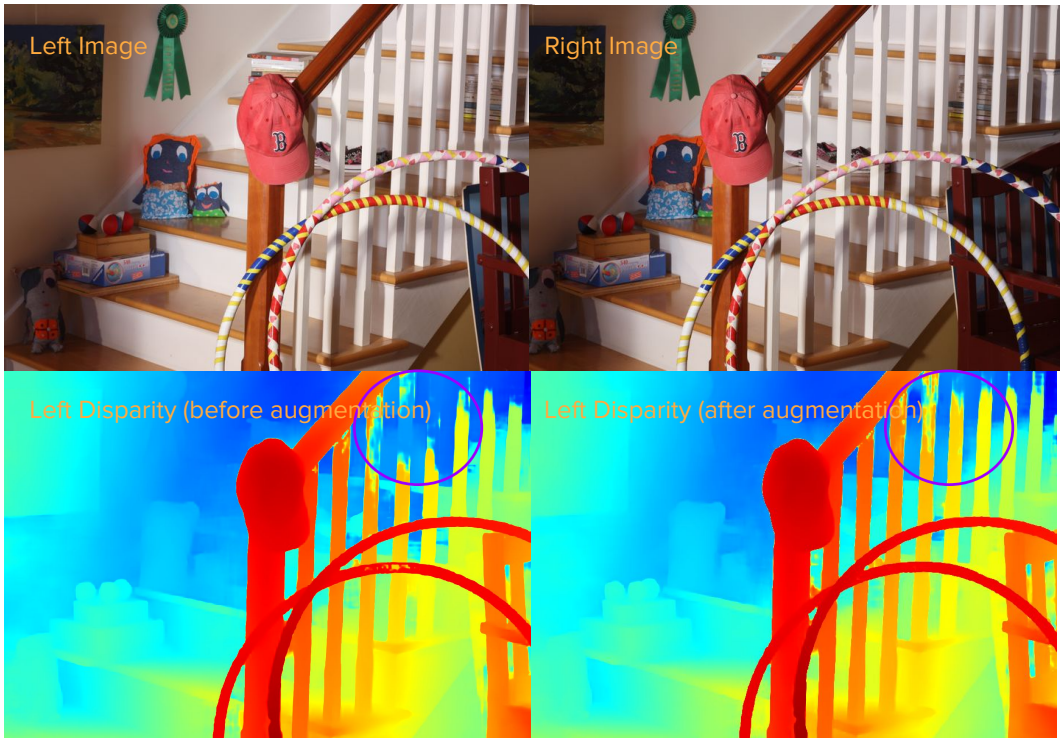

Figure 11: Effect of asymmetric masking augmentation on Middlebury14 test image Australia. After applying asymmetric masking augmentation during training time, the model's ability to handle calibration error get improved in the purple-circled region.

	avgerr (px)			bad-2.0 (%)		
Method	perfect	imperfect	increase (%)	perfect	imperfect	increase (%)
HSM-H	**3.9**	**4.1**	**5.9**	**26.2**	**27.5**	**5.0**
ELAS-H [4]	8.3	10.8	29.6	30.4	34.6	14.1

Table 3: Resilience to calibration error on Middlebury14 additional images. For each of the 12 image pairs, we evaluate separately on both its perfect and imperfect-calibrated version. We also calculate the increase of error rate when the calibration becomes imperfect (the lower the better).

of bad-2.0 metric. The robustness of our model is due to the coarse-to-fine matching process where the calibration error is suppressed in the coarse scale. Our y-disparity augmentation is also helpful in that it forces the network to learn features robust to y-disparities.

We then test our model's sensitivity to calibration error as follows: we select images with perfect calibration (12 in total) and 1) rotate the target view images around center for $\theta \in \{0, 0.05, 0.1, \ldots, 0.4\}$ degrees, as well as 2) translate along y-axis for $y_d \in \{0, 0.5, 1, \ldots, 4\}$ pixels, while keeping the reference view unchanged. The resulting avgerr to is shown in Fig. 12. We find that under a small rotation of 0.05 degrees (resulting 1.07 px y-disparity around the image border), our model's error rate only increases by 5.3%, while under a large rotation of 0.4 degrees (resulting 8.6 px y-disparity around the image border), our model's error increases by 105%, while still better than the results of ELAS-H on perfectly calibrated images (8.0 versus 8.3). Also under a small translation of 1px, our model's error rate almost does not increase (only rise by 1.0%), while under a large translation of 8 px, our model's error increases by 51.5%.

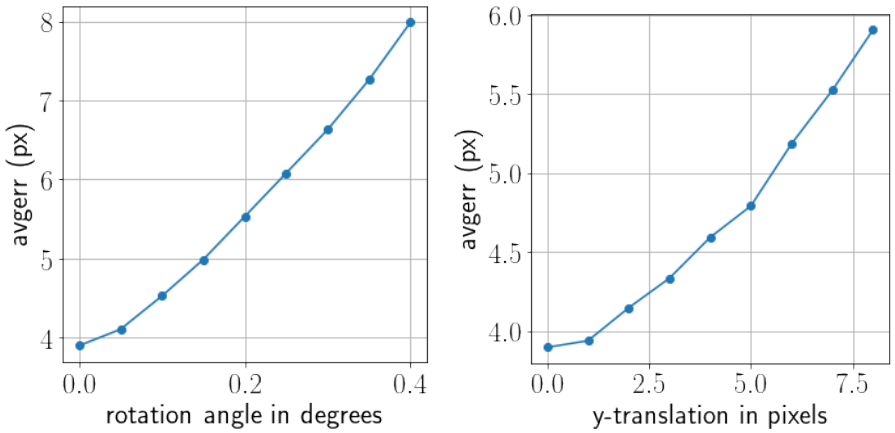

Figure 12: Resilience to calibration error.

	avgerr (px)			bad-2.0 (%)		
Method	normal	E	increase (%)	normal	E	increase (%)
HSM-H	**4.0**	**4.3**	**8.2**	**26.9**	**30.0**	**11.5**
ELAS-H [4]	9.5	13.7	44.2	32.5	41.0	26.2

Table 4: Resilience to exposure change on Middlebury14 additional images. For each of the 24 image pairs, we evaluate separately on both its normal and changed exposure (E) version, and calculate the increase of error rate when the exposure changes (the lower the better).

	avgerr (px)			bad-2.0 (%)		
Method	normal	L	increase (%)	normal	L	increase (%)
HSM-H	**4.0**	**9.3**	**131.1**	**26.9**	**51.2**	90.3
ELAS-H [4]	9.5	23.6	148.4	32.5	57.8	**77.8**

Table 5: Resilience to lighting change on Middlebury14 additional images. For each of the 24 image pairs, we evaluate separately on both its normal and changed lighting (L) version, and calculate the increase of error rate when the exposure changes (the lower the better).

D.5. Robustness to change of imaging conditions

We first compare our HSM-H model to ELAS-H in terms of robustness to exposure change and robustness to lighting changes as shown in Tab. 4, where we found our model to be more robust: when exposure changes, the average error of HSM-H only increases 8.2%, while that of ELAS-H increases by 44.2%. The robustness of our model also shows the effectiveness of asymmetric chromatic augmentation, which forces the model to learn features robust to chromatic changes and therefore helps to find correct correspondences.

We then compare HSM-H with ELAS-H in terms of robustness to lighting changes as shown in Tab. 5. We found that when lighting changes, the average error of HSM-H increases 131.1%, higher than that of ELAS-H (148.4%). Although in terms of bad-2.0, the error of HSM-H increases more than ELAS-H (90.3 versus 78.8), HSM-H still performs better than ELAS-H (51.2 versus 57.8). We also include a failure hard case for both HSM/ELAS as in Fig 13. The proposed asymmetric chromatic augmentation does not address the problem of changed lighting between views. However, it is possible to take advantage of synthetic data generated under different lighting conditions to train a model robust to such variants.

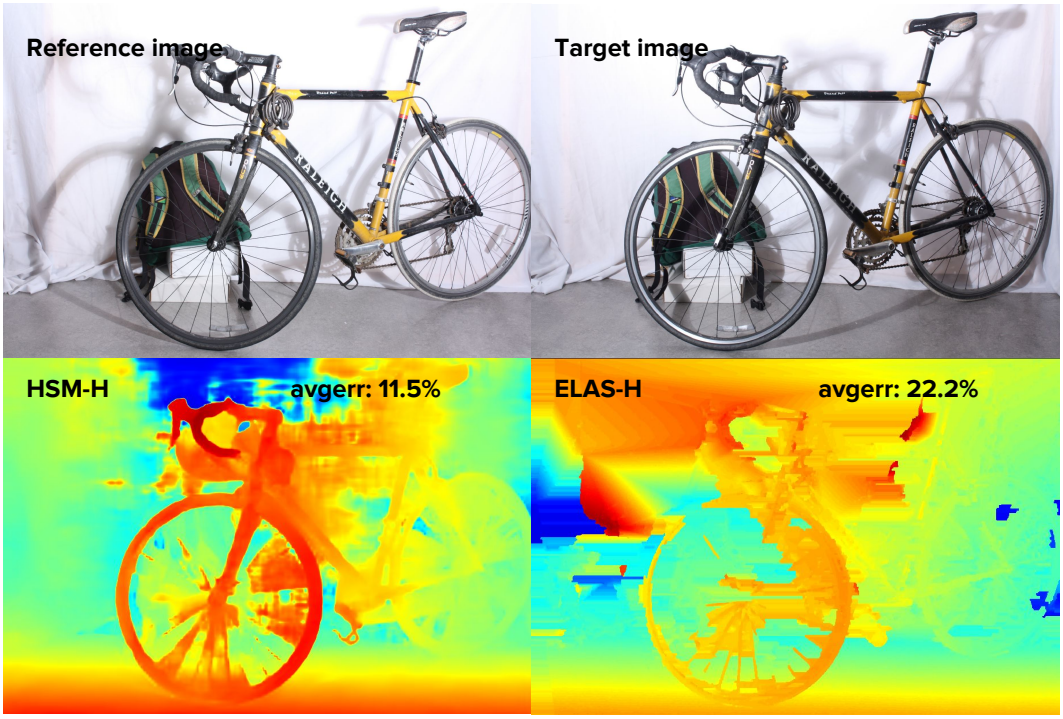

Figure 13: A failure case of our model on "Bicycle1" with changed lighting condition. Notice that change of lighting condition introduces different shading to reference and target images, and both HSM and ELAS cannot handle that.

D.6. Robustness to camera occlusion

In real-world autonomous driving scenarios, abnormal cases may occur when one camera is occluded/blurred by raindrops or snowflakes, which introduces challenges to the robustness of stereo matching algorithms. To study the effect of camera occlusion, we place a rectangle at target image center filled with the RGB means while keeping the reference view unchanged. Results are shown in Fig. 14. We find that with the increased number of occluded pixels, the average error grows slowly at the beginning (0.1% from 0 to with 5625 occluded pixels), and faster towards the end. We also find that our model is capable of handling small camera occlusions, by inferring disparities based on monocular cues and context cues.

E. High-resolution Real Stereo (HR-RS) benchmark

We include more results in the benchmark evaluation as shown in Tab. 6. ELAS [4] is taken from the Robust Vision Challenge official package, iResNet [10] is taken from their Github repository, and we implemented two-pass SGBM2 [5] using OpenCV (with SAD window size = 3, truncation value for pre-filter = 63, p1 = 216, p2 = 864, uniqueness ratio = 10, speckle window size = 100, speckle range = 32). The results from SGBM2 is also post-processed using weighted least square filter with default parameters. "HSM-FH" means the model operates on full resolution images and make predictions at the $\frac{1}{2}$ scale (2nd last scale).

We find that our method (HSM-F) processes high-resolution images efficiently and out-performs the baselines by a large margin at all distance ranges. Interestingly, we find that with an increased resolution, our model performs better on all distance ranges. This is possibly because high-resolution not only helps to find tiny objects far-away, but also thin objects near-by.

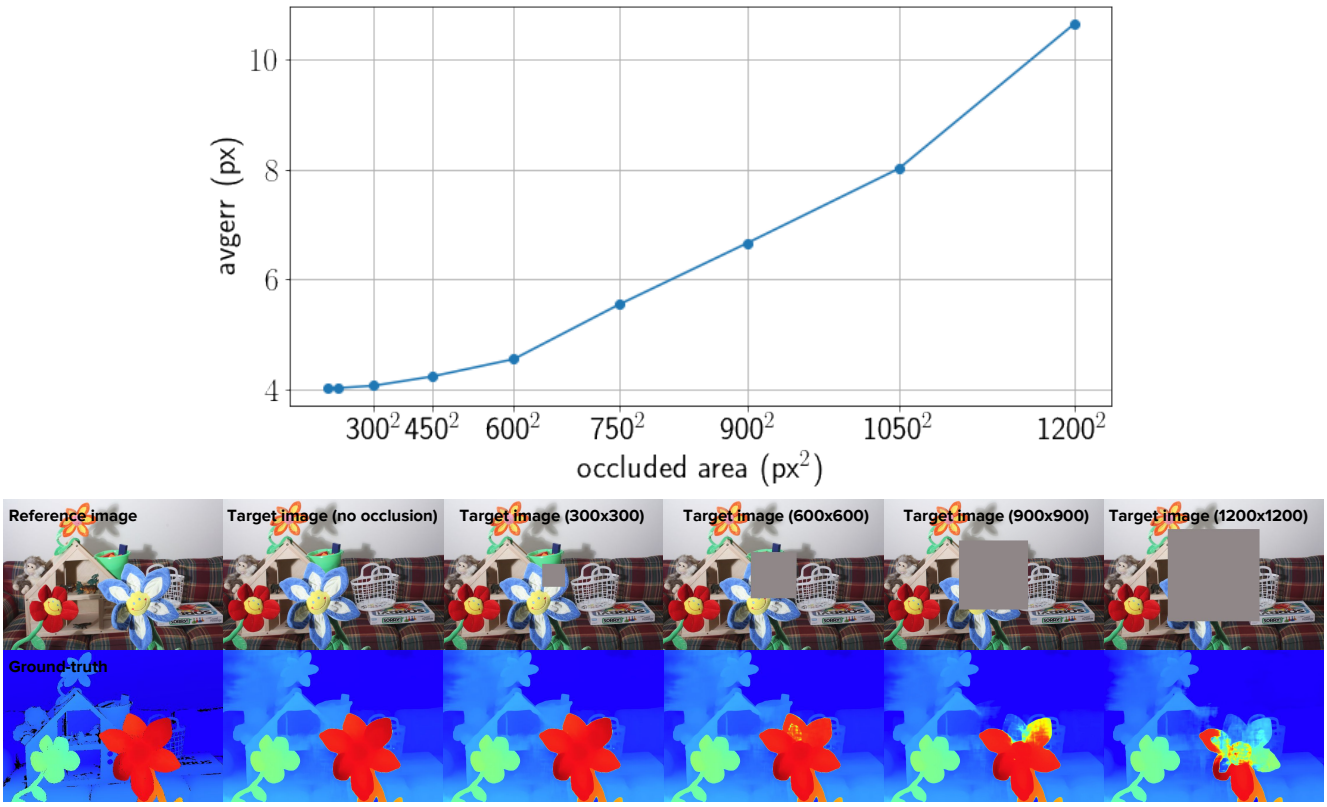

Figure 14: Our model's resilience to camera occlusion. **Top:** Average error grows with the increased number of occluded pixels. **Bottom:** Qualitative effect of camera occlusion to our model.

		avgerr (px)				bad-4.0 (%)			
method	time (ms)	S	M	L	All	S	M	L	All
HSM-FQ	91	6.22	4.93	3.87	5.08	42.3	43.5	33.7	40.9
HSM-FH	175	2.98	3.00	2.76	2.89	16.5	18.8	17.1	17.1
HSM-F	430	**2.96**	2.88	**2.71**	**2.82**	**15.7**	**16.7**	**14.9**	**15.5**
HSM-HH	42	3.64	3.42	3.80	3.46	25.9	27.6	34.4	26.9
HSM-H	74	3.04	**2.83**	3.01	2.86	18.9	18.9	19.7	18.2
HSM-Q	29	3.69	3.68	3.80	3.63	30.3	32.7	33.4	31.2
ELAS-H [4]	464	11.63	4.96	3.42	6.99	49.4	32.2	23.9	36.1
SGBM2-Q [5]	1321	12.90	3.87	3.11	6.60	50.8	27.3	19.5	32.8
iResNet-H [10]	410	4.90	3.71	3.49	4.00	32.1	24.4	22.8	25.8

Table 6: Results on HR-RS dataset. All pixels are evaluated at 3 ranges. S: 0-25m, M: 25-60m, L: 60-115m. HSM achieves significant improvement in all metrics compared to SOTA baselines.

References

[1] K. Batsos, C. Cai, and P. Mordohai. Cbmv: A coalesced bidirectional matching volume for disparity estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 6

[2] J.-R. Chang and Y.-S. Chen. Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5410–5418, 2018. 6

[3] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017. 1

[4] A. Geiger, M. Roser, and R. Urtasun. Efficient large-scale stereo matching. In *Computer Vision–ACCV 2010*, pages 25–38. Springer, 2010. 6, 8, 9, 10, 11

[5] H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on pattern analysis and machine intelligence*, 30(2):328–341, 2008. 6, 10, 11

[6] E. Ilg, T. Saikia, M. Keuper, and T. Brox. Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation. *arXiv preprint arXiv:1808.01838*, 2018. 6

[7] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach, and A. Bry. End-to-end learning of geometry and context for deep stereo regression. 4

[8] J. Kowalczuk, E. T. Psota, and L. C. Perez. Real-time stereo matching on cuda using an iterative refinement method for adaptive support-weight correspondences. *IEEE transactions on circuits and systems for video technology*, 23(1):94–104, 2013. 6

[9] C. LeGendre, K. Batsos, and P. Mordohai. High-resolution stereo matching based on sampled photoconsistency computation. In *British Machine Vision Conference*, 2017. 6

[10] Z. Liang, Y. Feng, Y. Guo, H. Liu, W. Chen, L. Qiao, L. Zhou, and J. Zhang. Learning for disparity estimation through feature constancy. 2018. 6, 10, 11

[11] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. arXiv:1512.02134. 3

[12] M. Menze and A. Geiger. Object scene flow for autonomous vehicles. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 3

[13] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, and P. Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *German Conference on Pattern Recognition*, pages 31–42. Springer, 2014. 3

[14] S. N. Sinha, D. Scharstein, and R. Szeliski. Efficient high-resolution stereo matching using local plane sweeps. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1582–1589, 2014. 6

[15] J. Zbontar and Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research*, 17(1-32):2, 2016. 6

[16] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2881–2890, 2017. 4