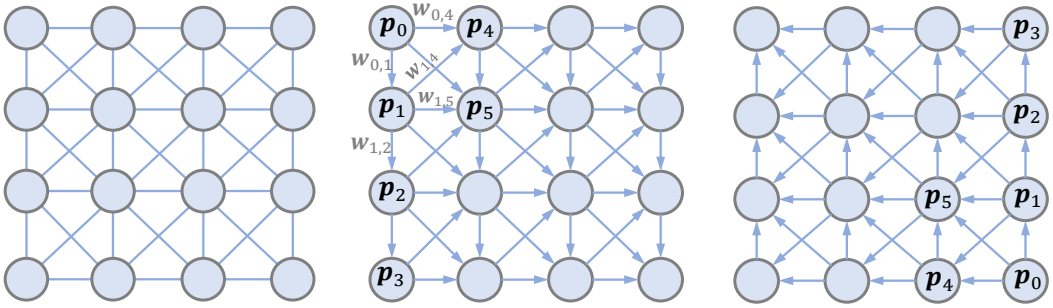
Supplementary Materials for "DSMNet"



(a) 8-connected graph (b) directed graph G_1 (c) directed graph G_2

Figure 1: Illustration of the graph construction. The 8-way connected graph is separated into two directed graphs G_1 and G_2.

1. Proof of Footnote 1

The proposed non-local filter is defined as:

$$\begin{aligned} C_i^A(\mathbf{p}) &= \frac{\sum_{\mathbf{q}\in G_i} W(\mathbf{q},\mathbf{p})\cdot C(\mathbf{q})}{\sum_{\mathbf{q}\in G_i} W(\mathbf{q},\mathbf{p})}, \\ W(\mathbf{q},\mathbf{p}) &= \sum_{l_{\mathbf{q},\mathbf{p}}\in G_i} \prod_{e\in l_{\mathbf{q},\mathbf{p}}} \omega_e. \end{aligned} \tag{1}$$

Following all the variable definitions in the paper, here, we prove that

$$\sum_{\mathbf{q}\in G_i} W(\mathbf{q},\mathbf{p}) = 1, \quad \text{if} \sum_{\mathbf{q}\in N_\mathbf{p}} \omega_{e(\mathbf{q},\mathbf{p})} = 1. \tag{2}$$

Since any path which reaches node $\mathbf{p}$ must pass through its neighborhoods $\mathbf{q}$, we can expand $W(\mathbf{q},\mathbf{p})$ to get that

$$\sum_{\mathbf{q}\in G_i} W(\mathbf{q},\mathbf{p}) = \omega_{e(\mathbf{p},\mathbf{p})} + \sum_{\mathbf{p}'\in N_\mathbf{p}, \mathbf{p}'\neq\mathbf{p}} \omega_{e(\mathbf{p}',\mathbf{p})} \sum_{\mathbf{q}\in G_i} W(\mathbf{q},\mathbf{p}')$$

Following the order of $\mathbf{p}_0, \mathbf{p}_1 ... \mathbf{p}_n ... \mathbf{p}_N$ (Fig. 1), we can prove Eq. (2) by mathematical induction:
When $n=0$, for $\mathbf{p}_0$, $\sum_{\mathbf{q}\in G_i} W(\mathbf{q},\mathbf{p}_0) = W(\mathbf{p}_0,\mathbf{p}_0) = \omega_{e(\mathbf{p}_0,\mathbf{p}_0)} = 1$
Assume when $n \le t$, $\sum_{\mathbf{q}\in G_i} W(\mathbf{q},\mathbf{p}_n) = 1$.
We can get that for $n = t+1$:

$$\begin{aligned} \sum_{\mathbf{q}\in G_i} W(\mathbf{q},\mathbf{p}_{t+1}) &= \omega_{e(\mathbf{p}_{t+1},\mathbf{p}_{t+1})} + \sum_{\mathbf{p}_k\in N_{\mathbf{p}_{t+1}}, \mathbf{p}_k\neq\mathbf{p}_{t+1}} \omega_{e(\mathbf{p}_k,\mathbf{p}_{t+1})} \sum_{\mathbf{q}\in G_i} W(\mathbf{q},\mathbf{p}_k) \\ &= \omega_{e(\mathbf{p}_{t+1},\mathbf{p}_{t+1})} + \sum_{\mathbf{p}_k\in N_{\mathbf{p}_{t+1}}, \mathbf{p}_k\neq\mathbf{p}_{t+1}} \omega_{e(\mathbf{p}_k,\mathbf{p}_{t+1})} \cdot 1 \\ &= \sum_{\mathbf{p}_k\in N_{\mathbf{p}_{t+1}}} \omega_{e(\mathbf{p}_k,\mathbf{p}_{t+1})} \\ &= 1. \end{aligned}$$

Here, $k \le t$, since $\mathbf{p}_k \in N_{\mathbf{p}_{t+1}}$.
This yields the equivalence of Eq. (2).

2. Backpropagation

The proposed structure-preserving graph-based filter (SGF) can be realized as an iterative linear aggregation as:

$$C_i^A(\mathbf{p}) \;\; = \omega_{e(\mathbf{p},\mathbf{p})} \cdot C(\mathbf{p}) + \sum_{\mathbf{q}\in N_\mathbf{p}, \mathbf{q}\neq\mathbf{p}} \omega_{e(\mathbf{q},\mathbf{p})} \cdot C_i^A(\mathbf{q}) \tag{3}$$

The backpropagation for ω_e and $C(\mathbf{p})$ can be computed inversely. Assume the gradient from next layer is $\frac{\partial E}{\partial C_i^A}$. The backpropagation can be implemented as:

$$\begin{aligned}
\frac{\partial E}{\partial C(\mathbf{p})} &= \frac{\partial E}{\partial C_i^b(\mathbf{p})} \cdot \omega_{e(\mathbf{p},\mathbf{p})}, \\
\frac{\partial E}{\partial \omega_{e(\mathbf{p},\mathbf{p})}} &= \frac{\partial E}{\partial C_i^b(\mathbf{p})} \cdot C(\mathbf{p}), \\
\frac{\partial E}{\partial \omega_{e(\mathbf{q},\mathbf{p})}} &= \frac{\partial E}{\partial C_i^b(\mathbf{p})} \cdot C_i^A(\mathbf{q}), \;\; \mathbf{q} \in N_\mathbf{p} \;\&\; \mathbf{q} \neq \mathbf{p}
\end{aligned} \tag{4}$$

where, $\frac{\partial E}{\partial C_\mathbf{i}^b}$ is a temporary gradient variable which can be calculated iteratively (similar to Eq. (3)):

$$\frac{\partial E}{\partial C_i^b(\mathbf{p})} = \frac{\partial E}{\partial C_i^A(\mathbf{p})} + \sum_{\mathbf{q}\in N_\mathbf{p}, \mathbf{q}\neq\mathbf{p}} \frac{\partial E}{\partial C_i^b(\mathbf{q})} \cdot \omega_e(\mathbf{q},\mathbf{p}) \tag{5}$$

The propagation of Eq. (5) is an inverse process and in an order of $\mathbf{p}_N, \mathbf{p}_{N-1}, ...\mathbf{p}_0$

3. Details of the Architecture

Table 3 presents the details of the parameters of the DSMNet. It has seven SGF layers which are used in feature extraction and cost aggregation. The proposed Domain Normalization layer is used to replace Batch Normalization after each 2D convolutional layer in the feature extraction and guidance networks.

4. Efficiency and Parameters

As shown in Table 1, our proposed SGF is a linear process that can be realized efficiently. The inference time is increased by about 5$\sim$10% compared with the baseline. Moreover, no any new parameters are introduced for the proposed domain normalization and SGF layers.

We also compare the memory requirements of state-of-the-art stereo matching models in Table 1 (test phase with KITTI resolution: 1242×375). The memory requirements are *PSMNet (4.6) vs. PSMNet-DSMNet (4.9)* and *GANet (6.4) vs. DSMNet (5.8)*. DSMNet consumes less memory than GANet. It uses no LGA layers [11]. Compared with other non-local strategies [4, 8, 9], our SGF is realized by iterative linear propagation and has a lower complexity in memory requirement.

Table 1: Comparisons of Memory, Elapsed Time and Number of Parameter

Methods	Elapsed Time	Parameter Number	Memory (Test Phase, GB)
GANet-deep [11]	1.8s	60M	6.4
Baseline	1.4s	48M	5.5
Our DSMNet	1.5s	48M	5.8
PSMNet [1]	0.4s	52M	4.6
DSMNet (PSMNet)	0.42s	52M	4.9

5. Comparison with BN

In Fig. 3, we compare the batch normalization (BN) and our proposed domain normalization (DN). *Mean* and *Variance* of the 32-channel features are computed using five different datasets. Different normalization strategies are implemented and all other settings are kept the same. The two models (BN or DN) are trained on the same synthetic dataset and test on five different datasets. The output of the last convolutional layer (with ReLU) in the feature extraction network is used to calculate the mean and variance. We can find that, for BN, different datasets have different mean and variances in each of the 32 feature channels. This will significantly influence the domain generalization abilities. As a comparison, our DN can remove the mean and variance shifts between different datasets.

6. Carla Dataset

Since Sceneflow dataset only has limited number of stereo pairs for diving scenes, we use the Carla [3] platform to produce the stereo pairs for outdoor driving scenes. As shown in Table 2, the new carla supplementary dataset has more diverse settings, including two kinds of image resolutions (720×1080 and 1080×1920), three different focal lengths, and six different camera baselines (in a range of 0.2-1.5m). This supplementary dataset can significantly improve the diversity of the training set. As shown in Fig. 2, the Carla data still have significant domain differences (*e.g.* color, textures) compared with the real scenes (*e.g.* KITTI, CityScapes), but, our DSMNet focus on extract shape and structure information for robust stereo matching. These can be better transferred to the real scenes and produce more accurate disparity estimation.

Table 2: Statistics of the Carla Stereo Dataset

dataset	number of pairs	focal length	baseline settings	resolutions
SceneFlow	34,000	450, 1050	0.54	960×540
Carla stereo	20,000	640, 670, 720	0.2, 0.3, 0.5, 1.0, 1.2, 1.5	1280×720, 1920×1080

(a) left view (b) right view (c) disparity map

Figure 2: Example of the Carla stereo data.

7. More Results

7.1. Feature Visualization

As compared in Fig. 4, the features of the state-of-the-art models are mainly local patterns which can have a lot of artifacts (*e.g.* noises) when suffering from domain shifts. Our DSMNet mainly captures the non-local structure and shape information, which are robust for cross-domain generalization. There is no artifacts in the feature maps of our DSMNet.

7.2. Disparity Results on Different Datasets

More results and comparisons are provided in Fig. 5. All the models are trained on the synthetic dataset and tested on the real KITTI, Middlebury, ETH3D and Cityscapes datasets.

7.3. Training with Other Datasets

Training on "Flyingthings3D": We also tried to train only with "Flyingthings3D" dataset (without synthetic dirving scenes) and evaluate on the KITTI 2015 real driving scenes. Without synthetic driving scenes for training, error rates (%) are: *PSMNet (25.1) vs. GANet (19.5) vs. DSMNet (9.8)*. DSMNet outperforms others by 9$\sim$15%.

Indoor and outdoor domains: We test the cross-domain generalizations between KITTI (outdoor) and Middlebury (indoor) scenes:

i) From KITTI to Middlebury, error rates (%) are *PSMNet (33.6) vs. GANet (29.1) vs. DSMNet (20.5)*. DSMNet outperforms the state of the arts by 8$\sim$13% in accuracy.

ii) From Middlebury to KITTI, error rates (%) are *PSMNet (15.0) vs. GANet (11.2) vs. DSMNet (6.3)*. Our DSMNet again outperforms the state of the arts by 5$\sim$9%.

7.4. Comparisons with Non-local Networks and Attentions

Our graph-based filtering strategy is better for capturing the structural and geometric context for robust domain-invariant stereo matching. The non-local neural network denoising [9] and non-local attention [4] do not have spatial constraints that usually lead to over smoothness of the depth edges and thin structures (as shown in Fig. 6).

References

[1] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5410–5418, 2018.

[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)*, pages 3213–3223, 2016.

[3] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. *arXiv preprint arXiv:1711.03938*, 2017.

[4] Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. Ccnet: Criss-cross attention for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 603–612, 2019.

[5] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4040–4048, 2016.

[6] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3061–3070, 2015.

[7] Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nešić, Xi Wang, and Porter Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *German conference on pattern recognition*, pages 31–42. Springer, 2014.

[8] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7794–7803, 2018.

[9] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 501–509, 2019.

[10] Zhichao Yin, Trevor Darrell, and Fisher Yu. Hierarchical discrete distribution decomposition for match density estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6044–6053, 2019.

[11] Feihu Zhang, Victor Prisacariu, Ruigang Yang, and Philip HS Torr. Ga-net: Guided aggregation net for end-to-end stereo matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 185–194, 2019.

Table 3: Parameters of the network architecture of "DSMNet"

No.	Layer Description	Output Tensor
	Feature Extraction	
input	normalized image pair as input	H×W×3
1	3×3 conv, **DN**, ReLU	H×W×32
2	3×3 conv, stride 3, **DN**, ReLU	$1/3$H×$1/3$W×32
3	3×3 conv, **DN**, ReLU	$1/3$H×$1/3$W×32
4	**SGF**, **DN**, ReLU	$1/3$H×$1/3$W×32
5	3×3 conv, stride 2, **DN**, ReLU	$1/6$H×$1/6$W×48
6	**SGF**, **DN**, ReLU	$1/6$H×$1/6$W×48
7	3×3 conv, **DN**, ReLU	$1/6$H×$1/6$W×48
8-9	repeat 5,7	$1/12$H×$1/12$W×64
10-11	repeat 8-9	$1/24$H×$1/24$W×96
12-13	repeat 8-9	$1/48$H×$1/48$W×128
14	3×3 deconv, stride 2, **DN**, ReLU	$1/24$H×$1/24$W×96
15	3×3 conv, **DN**, ReLU	$1/24$H×$1/24$W×96
16-17	repeat 14-15	$1/12$H×$1/12$W×64
18-19	repeat 14-15	$1/6$H×$1/6$W×48
20	**SGF**, **DN**, ReLU	$1/6$H×$1/6$W×48
21-22	repeat 14-15	$1/3$H×$1/3$W×32
23-41	repeat 4-22	$1/3$H×$1/3$W×32
42	**SGF**, **DN**, ReLU	$1/3$H×$1/3$W×32
concat	(11,14), (9,16), (7,18), (4,21), (20,23), (17,25), (15,27), (13,33), (18,25)	
connection	(25,28), (23,30) (21,35), (19,37) (23, 40)	
cost volume	by feature concatenation	$1/3$H×$1/3$W×64×32
	Guidance Branch	
input	concat 1 and up-sampled 35 as input	H×W×64
(1)	3×3 conv, **DN**, ReLU	H×W×16
(2)	3×3 conv, stride 3, **DN**, ReLU	$1/3$H×$1/3$W×32
(3)	3×3 conv, **DN**, ReLU	$1/3$H×$1/3$W×32
(4)	3×3 conv (no bn & relu)	$1/3$H×$1/3$W×20
(5)	split, reshape, normalize	4× $1/3$H×$1/3$W×5
(6)-(8)	from (3), repeat (3)-(5)	4× $1/3$H×$1/3$W×5
(9)-(11)	from (6), repeat (6)-(8)	4× $1/3$H×$1/3$W×5
(12)	from (2), 3×3 conv, stride 2, **DN**, ReLU	$1/6$H×$1/6$W×32
(13)	3×3 conv, **DN**, ReLU	$1/6$H×$1/6$W×32
(14)	3×3 conv (no bn & relu)	$1/6$H×$1/6$W×20
(15)	split, reshape, normalize	4× $1/6$H×$1/6$W×5
(16)-(18)	from (13), repeat (13)-(15)	4× $1/6$H×$1/6$W×5
(19)-(21)	from (16), repeat (13)-(15)	4× $1/6$H×$1/6$W×5
(22)-(24)	from (19), repeat (13)-(15)	4× $1/6$H×$1/6$W×5
	Cost Aggregation	
input	4D cost volume	$1/3$H×$1/3$W×64×64
[1]	3×3×3, 3D conv	$1/3$H×$1/3$W×64×32
[2]	SGA layer: weight matrices from (5)	$1/3$H×$1/3$W×64×32
[3]	**SGF** layer	$1/3$H×$1/3$W×64×32
[4]	3×3×3, 3D conv	$1/3$H×$1/3$W×64×32
output	3×3×3, 3D to 2D conv, upsamping	H×W×193
	softmax, regression, loss weight: 0.2	H×W×1
[5]	3×3×3, 3D conv, stride 2	$1/6$H×$1/6$W×32×48
[6]	SGA layer: weight matrices from (15)	$1/6$H×$1/6$W×32×48
[7]	3×3×3, 3D conv, stride 2	$1/12$H×$1/12$W×16×64
[8]	3×3×3, 3D deconv, stride 2	$1/6$H×$1/6$W×32×48
[9]	3×3×3, 3D conv	$1/6$H×$1/6$W×32×48
[10]	SGA layer: weight matrices from (18)	$1/6$H×$1/6$W×32×48
[11]	3×3×3, 3D deconv, stride 2	$1/3$H×$1/3$W×64×32
[12]	3×3×3, 3D conv	$1/3$H×$1/3$W×64×32
[13]	SGA layer: weight matrices from (8)	$1/3$H×$1/3$W×64×32
[14]	**SGF** layer	$1/3$H×$1/3$W×64×32
output	3×3×3, 3D to 2D conv, upsamping	H×W×193
	softmax, regression, loss weight: 0.6	H×W×1
$[15-24]$	repeat $[5-14]$	$1/3$H×$1/3$W×64×32
final	3×3×3, 3D to 2D conv, upsamping	H×W×193
output	regression, loss weight: 1.0	H×W×1
connection	concat: (6,8), (4,11), (9,15), (7,17), (16,18), (14,20); add: (1,4)	

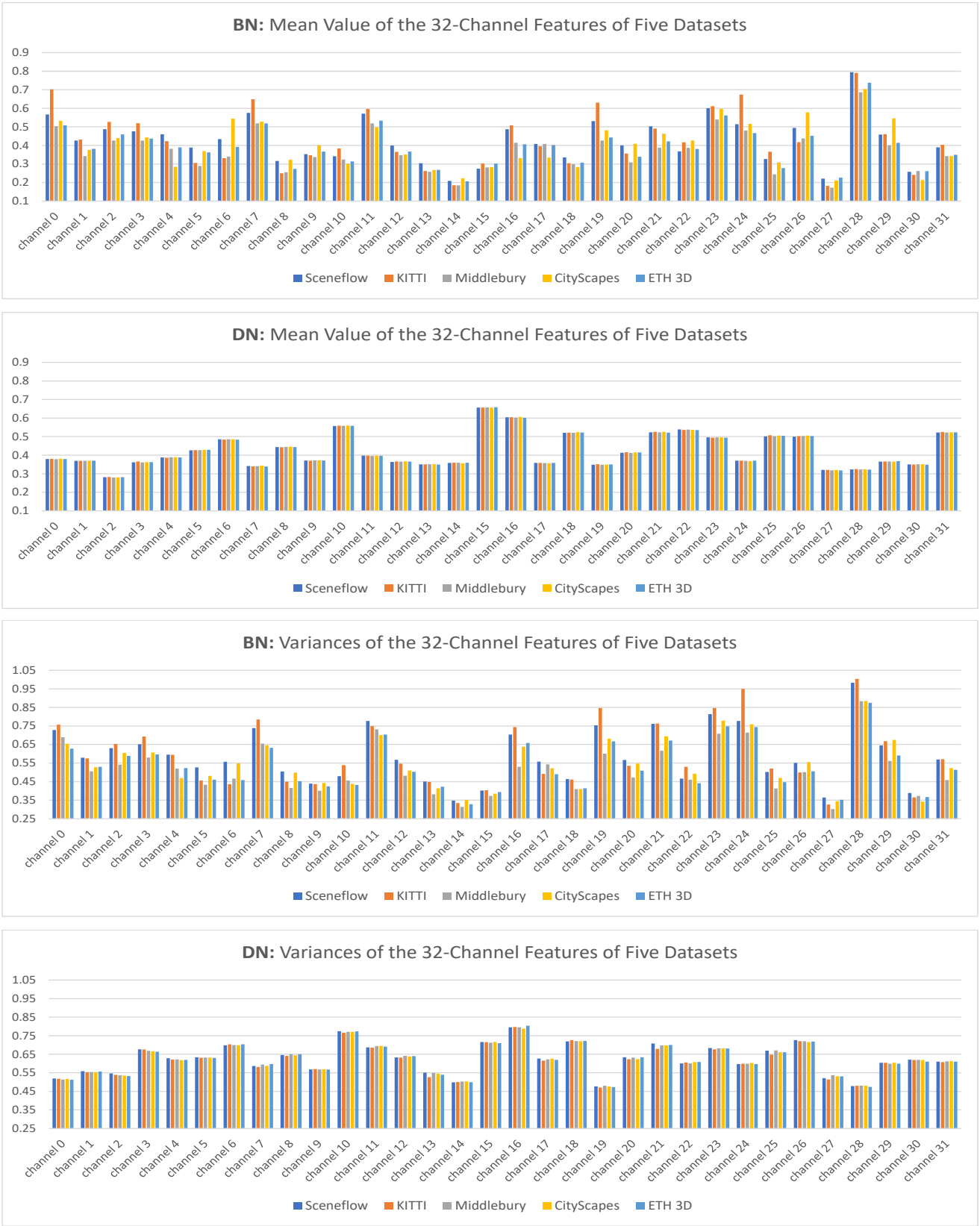
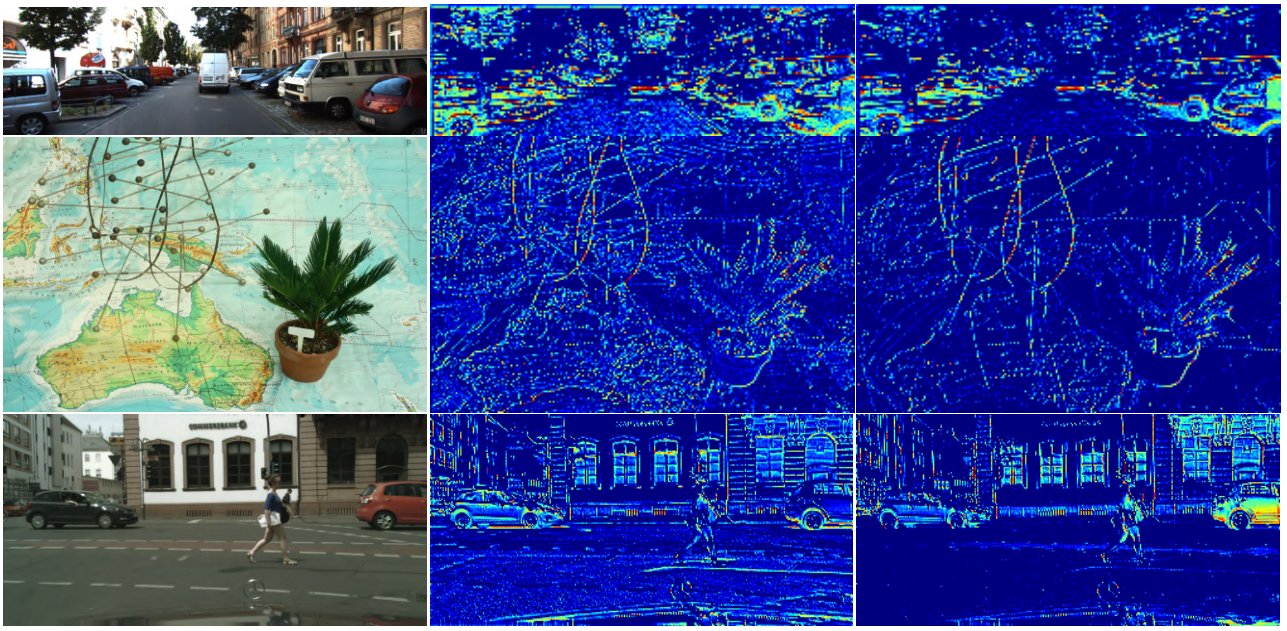
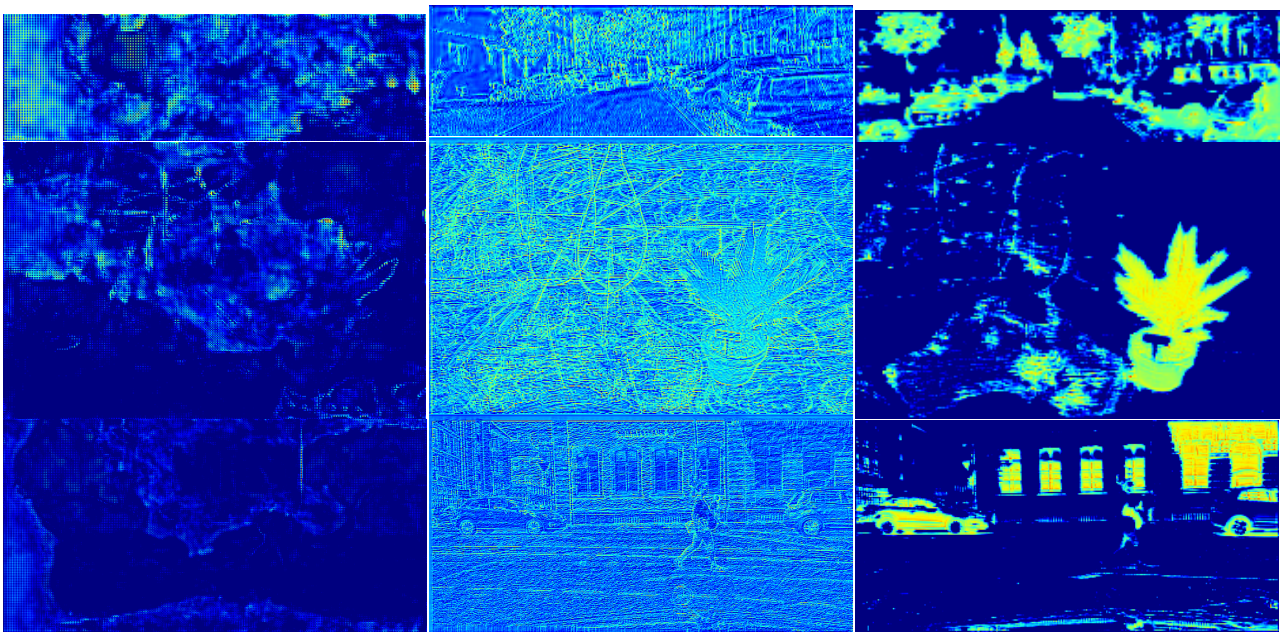


Figure 3: Comparisons of BN and DN. Mean and Variance of the 32-channel features are computed for five different datasets. The output of the feature extraction network is used to calculate the mean and variance.

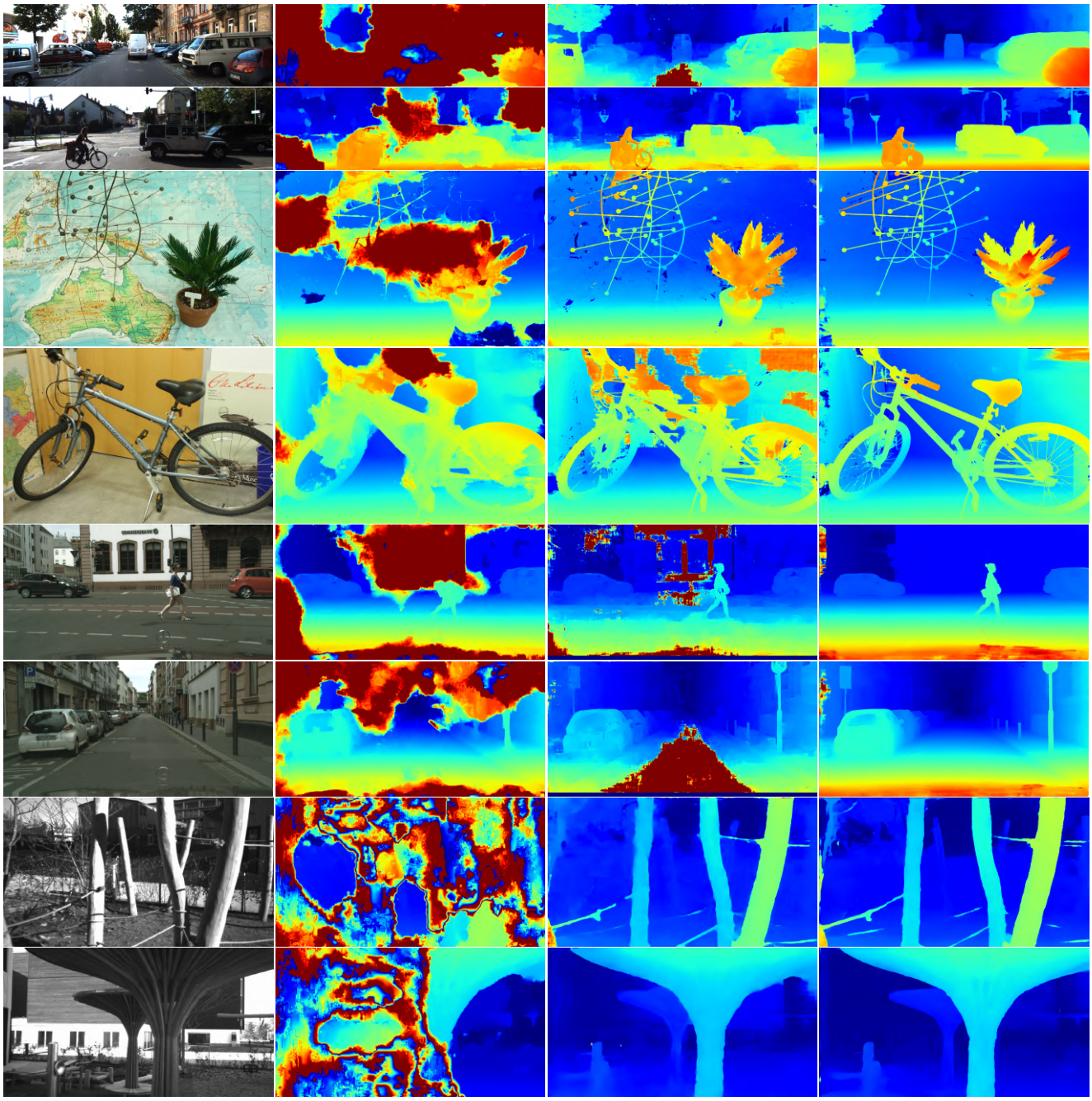


(a) Input view (b) GANet-synthetic (c) GANet-finetune



(d) HD^3-synthetic (e) PSMNet-synthetic (f) DSMNet-synthetic

Figure 4: Comparison and visualization of the feature maps for cross-domain test . (b) GANet [11], (d) HD^3 [10], (e) PSMNet [1] are trained on the synthetic dataset (Sceneflow [5]) and test on other real scenes/datasets (from top to bottom: Kitti [6], Middlebury [7] and CityScapes [2]). The features are mainly local patterns and produce a lot of artifacts (*e.g.* noises) when suffering from domain shifts. (c) GANet is finetuned on the test dataset for comparisons. The artifacts have been stressed after fine tuning. (f) Our DSMNet trained on the synthetic data. No distortions and artifacts are introduced on the feature maps. It mainly captures the non-local structure and shape information, which are more robust for cross-domain generalization.



(a) Input view (b) HD^3 [10] (c) PSMNet [1] (d) Our DSMNet

Figure 5: Comparisons with the state-of-the-art models on four real dataset (from top to bottom: KITTI, Middlebury, ETH3D and Cityscapes). All the models are trained on the synthetic dataset. Our DSMNet can produce accurate disparity estimation on other new datasets without fine-tuning.

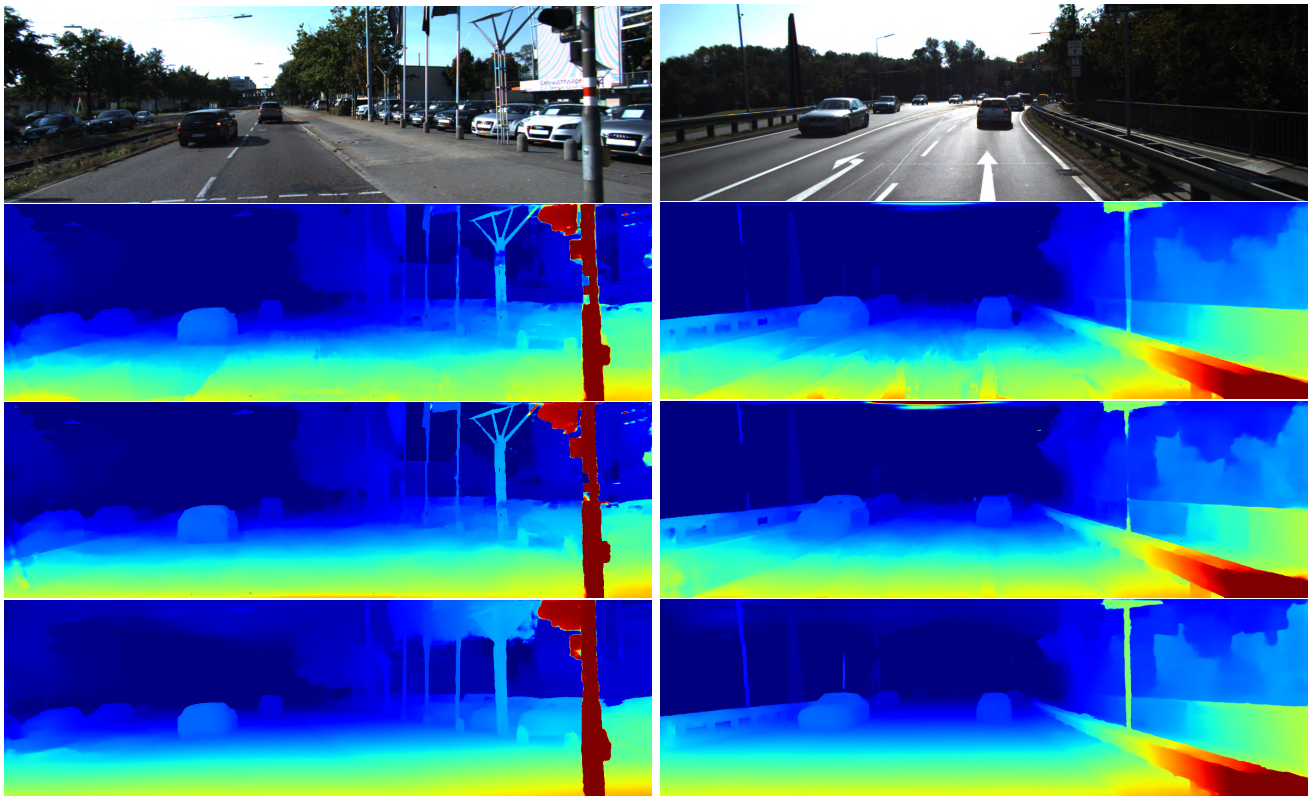


Figure 6: Comparisons with non-local attention mechanism [4] (*second row*) and non-local denoising [9] strategy (*third row*). When using these strategies, the thin structures (*e.g.* poles) are easily eroded by the background. These non-local strategies easily smooth out the disparity maps. As a comparison, our DSMNet (*last row*) can keep the thin structures of the disparity maps.