Domain-invariant Stereo Matching Networks

Feihu Zhang[1] Xiaojuan Qi[1] Ruigang Yang[2] Victor Prisacariu[1] Benjamin Wah[3] Philip Torr[1]
[1]University of Oxford [2]Baidu Research [3]CUHK

Abstract

State-of-the-art stereo matching networks have difficulties in generalizing to new unseen environments due to significant domain differences, such as color, illumination, contrast, and texture. In this paper, we aim at designing a domain-invariant stereo matching network (DSMNet) that generalizes well to unseen scenes. To achieve this goal, we propose i) a novel "domain normalization" approach that regularizes the distribution of learned representations to allow them to be invariant to domain differences, and ii) a trainable non-local graph-based filter for extracting robust structural and geometric representations that can further enhance domain-invariant generalizations. When trained on synthetic data and generalized to real test sets, our model performs significantly better than all state-of-the-art models. It even outperforms some deep learning models (e.g. *MC-CNN [54]) fine-tuned with test-domain data. The code and dataset will be avialable at* https://github.com/feihuzhang/DSMNet.

1. Introduction

Stereo reconstruction is a fundamental problem in computer vision, robotics and autonomous driving. It aims to estimate 3D geometry by computing disparities between matching pixels in a stereo image pair. Recently, many end-to-end deep neural network models (*e.g.* [4, 17, 56]) have been developed for stereo matching that achieve impressive accuracy on several datasets or benchmarks.

However, state-of-the-art stereo matching networks (supervised [4,17,56] and unsupervised [45,59]) cannot generalize well to unseen data without fine-tuning or adaptation. Their difficulties lie in the large domain differences (such as color, illumination, contrast and texture) between stereo images in various datasets. As illustrated in Fig. 1, the pre-trained models on one specific dataset produce poor results on other real and unseen scenes.

Domain adaptation and transfer learning methods (*e.g.* [3, 11, 45]) attempt to transfer or adapt from one source domain to another new domain. Typically, a large number of stereo images from the new domain are required for the adaptation. However, these cannot be easily obtained in many real scenarios. And, in this case, we still need a good method for disparity estimation even without data from the new domain for adaptation.

Thus, it is desirable to design a model that can generalize well to unseen data without re-training or adaptation. The difficulties for developing such a domain invariant stereo matching network (DSMNet) come from the significant domain differences between stereo images in various scenes/datasets (*e.g.* Fig. 1(a) and 1(b)). Such differences make the learned features unstable, distorted and noisy, leading to many wrong matching results.

Fig. 1 visualizes the features learned by some state-of-the-art stereo matching models [4, 53, 56]. Due to the limited effective receptive field of convolutional neural networks [28], they capture the domain-sensitive local patterns (*e.g.* local contrast, edge and texture) when constructing matching features, which, however, break down and produce a lot of artifacts (*e.g.* noises) in the feature maps when applied to the novel test data (Fig. 1(c)). The artifacts and distortions in the features inhibit robust matching, leading to wrong matching results (Fig. 1(e)).

In this paper, we propose two novel neural network layers for constructing the robust deep stereo matching network for cross-domain generalization without further fine-tuning or adaptation. Firstly, to reduce the domain shifts/differences between different datasets/scenes, we propose a novel domain normalization layer that fully regulates the feature's distribution in both the spatial (height and width) and the channel dimensions. Secondly, to eliminate the artifacts and distortions in the features, we propose a learnable non-local graph-based filtering layer that can capture more robust structural and geometric representations (*e.g.* shape and structure, as illustrated in Fig. 1(d)) for domain-invariant stereo matching.

We formulate our method as an end-to-end deep neural network model and train it only with synthetic data. In our experiments, without any fine-tuning or adaptation on the real test datasets, our DSMNet far outperforms: 1) almost all state-of-the-art stereo matching models (*e.g.* GANet [56]) trained on the same synthetic dataset, 2) most of the traditional methods (*e.g.* Cosfter filter, SGM [13]

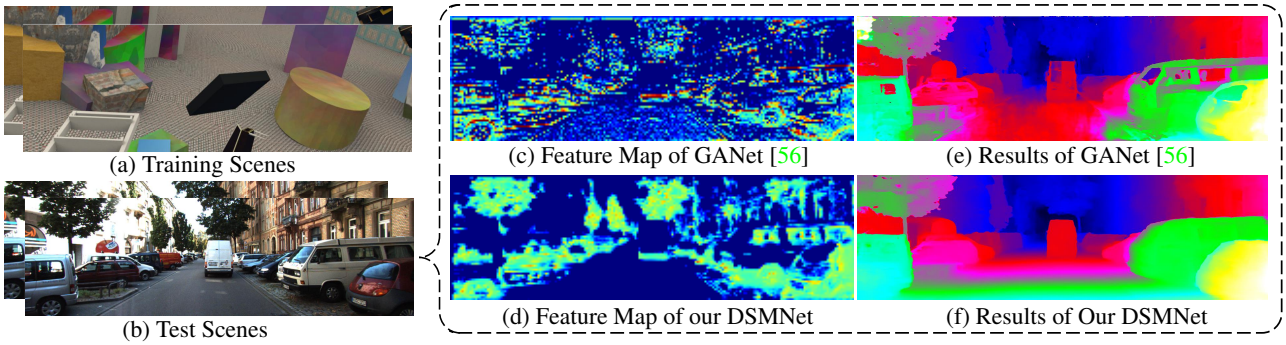


Figure 1: Visualization of the feature maps and disparity results. The state-of-the-art GANet [56] is used for comparisons. Models are trained on synthetic data (Sceneflow [30]) and tested on novel real scenes (KITTI [31]). The feature maps from GANet has many artifacts (*e.g.* noises). Our DSMNet mainly captures the structure and shape information as robust features, and there is no distortions or artifacts in the feature map. It can produce accurate disparity estimations in the novel test scenes. The same observations are shown by more models (*e.g.* PSMNet [4], HD^3 [53]) and datasets in the supplementary material.

et al.), 3) most of the unsupervised/self-supervised models trained on the target test domains. Our model even surpasses some of the fine-tuned (on the target domains) supervised deep neural network models (*e.g.* MC-CNN [54], content-CNN [29], DispNetC [30] *et al.*).

2. Related Work

2.1. Deep Neural Networks for Stereo Matching

In recent years, deep neural networks have seen great success in the task of stereo matching [4, 17, 56]. These models can be categorized into three types: 1) learning better features for traditional stereo matching algorithms, 2) correlation-based end-to-end deep neural networks, 3) cost-volume based stereo matching networks.

In the first category, deep neural networks have been used to compute patch-wise similarity scores as the matching costs [54, 57]. The costs are then fed into the traditional cost aggregation and disparity computation/refinement methods [13] to get the final disparity maps. The models are, however, limited by the traditional matching cost aggregation step and often produce wrong predictions in occluded regions, large textureless/reflective regions and around object edges.

DispNetC [30], a typical method in the second category, computes the correlations by warping between stereo views and attempts to predict the per-pixel disparity by minimizing a regression training loss. Many other sate-of-the-art methods, including iResNet [25], CRL [36], SegStereo [51], EdgeStereo [42], HD^3 [53], and MADNet [45], are all based on color or feature correlations between the left and right views for disparity estimation.

The recently developed cost-volume based models explicitly learn feature extraction, cost volume, and regularization function all end to end. Examples include GC-Net [17], PSM-Net [4] , StereoNet [18], AnyNet [49], GANet [56] and EMCUA [34]. They all utilize a similarity cost as the third dimension to build the 4D cost volume in which the real geometric context is maintained.

There are also others that combine the correlation and cost volume strategies (*e.g.* [12]).

The common feature of these models is that they all require a large number of training samples with ground truth depth/disparity. More importantly, a model trained on one specific domain cannot generalize well to new scenes without fine-tuning or retraining.

2.2. Adaptation and Self-supervised Learning

Self-supervised Learning: A recent trend of training stereo matching networks in an unsupervised manner relies on image reconstruction losses that are achieved by warping left and right views [58, 59]. However, they cannot solve the occlusions and reflective regions where there is no correspondence between the left and the right views. Also, they cannot generalize well to other new domains.

Domain Adaptation: Some methods pre-train the models on synthetic data and then explore the cross-domain knowledge to adapt [11, 37] for a new domain. Others focus on the online or offline adaptations [39, 43–45]. For example, MADNet [45] is proposed to adapt the pre-trained model online and in real time. But, it has poor accuracy even after the adaptation. Moreover, the domain adaptation approaches require a large number of stereo images from the target domain for adaptations. However, these cannot be easily obtained in many real scenarios. And, in this case, we still need a good method for disparity estimation even without data from the new domain for adaptation.

2.3. Cross-Domain Generalization

Different to domain adaptation, domain generalization is a much harder problem that assumes no access to target information for adaptation or fine-tuning. There are many approaches that explore the idea of domain-invariant feature

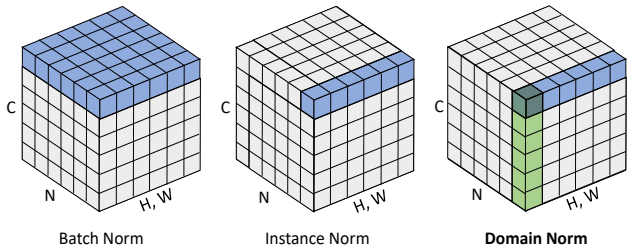
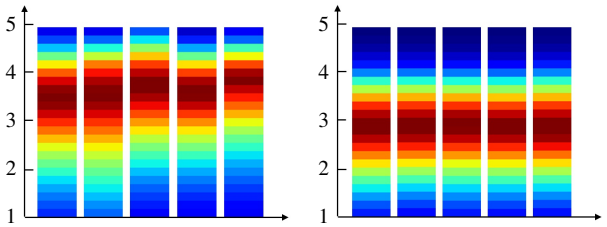


Figure 2: Normalization methods. Each subplot shows a feature map tensor, with N as the batch axis, C as the channel axis, and (H, W) as the spatial axes. The blue elements in set S are normalized by the same mean and variance. The proposed domain normalization consists of image-level normalization (blue, Eq. (1)) and pixel-level normalization of each C-channel feature vector (green, Eq. (3)).



(a) Instance Norm (b) Domain Norm

Figure 3: Norm distributions of the features of different datasets (from left to right: synthetic SceneFlow, KITTI, Middlebury, CityScapes and ETH 3D). We choose the output feature of the feature extraction network for our study. The norm of the C-channel feature vector of each pixel is counted for the distribution. Instance normalization can only reduce the image-level differences, but does not normalize the C-channel feature vectors at pixel level.

learning. Previous approaches focus on developing data-driven strategies to learn invariant features from different source domains [10, 20, 32]. Some recent methods utilize meta-learning that takes variations in multiple source domains to generalize to novel test distributions [1, 21]. Other approaches [22, 23] employ an invariant adversarial network to learn domain-invariant representation/features for image recognition. Choy *et al*. [6] develop a universal feature learning framework for visual correspondences using deep metric learning.

In contrast to the above approaches, there are methods that try to improve the batch or instance normalization in order to improve the generalization and robustness for style transfer or image recognition [24, 33, 35].

In summary, for stereo matching, work is seldom done to improve the generalization ability of the end-to-end deep neural network models, especially when developing the domain-invariant stereo matching networks.

3. Proposed DSMNet

To overcome the challenges in cross-domain generalization, we develop in the following sections our domain-invariant stereo matching networks. These include domain normalization to remove the influence of the domain shifts (*e.g*. color, style, illuminance), as well as non-local graph-based filtering and aggregation to capture the non-local structural and geometric context as robust features for domain-invariant stereo reconstruction.

3.1. Domain Normalization

Batch normalization (BN) has become the default feature normalization operation for constructing end-to-end deep stereo matching networks [4, 17, 30, 42, 45, 56]. Although it can reduce the internal covariate shift effects in training deep networks, it is domain-dependent and has negative influence on the model's cross-domain generalization ability.

BN normalizes the features as follows:

$$\hat{x}_i = \frac{1}{\sigma}(x_i - \mu_i). \tag{1}$$

Here x and $\hat{x}$ are the input and output features, respectively, and i indexes elements in a tensor (*i.e*. feature maps, as illustrated in Fig. 2) of size $N \times C \times H \times W$ (N: batch size, C: channels, H: spatial height, W: spatial width). μ_i and σ_i are the corresponding channel-wise mean and standard deviation (std) and are computed by:

$$\mu_i = \frac{1}{m}\sum_{k\in S_i} x_k, \quad \sigma_i = \sqrt{\frac{1}{m}\sum_{k\in S_i}(x_k - \mu_i)^2 + \varepsilon}, \tag{2}$$

where S_i is the set of elements in the same channel as element i (Fig. 2), and ε is a small constant to avoid dividing by zeros.

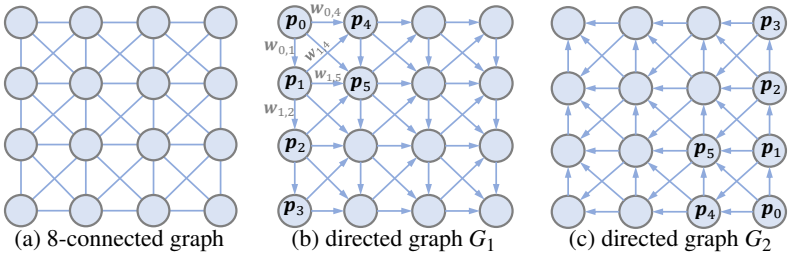
Mean μ and standard deviation σ are computed per batch in the training phase, and the accumulated values of the training set are utilized for inference. However, different domains may have different μ and σ caused by color shifts, contrast, and illumination (Fig. 1(a) and 1(b)). Thus μ and σ computed for one dataset are not transferable to others.

Instance normalization (IN) [33, 38] overcomes the dependency on data-set statistics by normalizing each sample separately, where elements in S_i are confined to be from the same sample as illustrated in Fig. 2. In theory, IN is domain-invariant, and normalization across the spatial dimensions (H, W) reduces image-level appearance/style variations.

However, matching of stereo views is realized at the pixel level by finding an accurate correspondence for each pixel using its C-channel feature vector. Any inconsistence of the feature norm and scaling will significantly influence the matching cost and similarity measurements.

Fig. 3 illustrates that IN cannot regulate the norm distribution of pixel-wise feature vectors that vary in datasets/domains.

We propose in Fig. 2 our **domain-invariant normalization (DN)**. Our method normalizes features along the spatial axis (H, W) to induce style-invariant representations similar



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

Figure 4: Illustration of the graph construction. The 8-way connected graph is separated into two directed graphs G_1 and G_2.

to IN as well as along the channel dimension (C) to enhance the local invariance.

Our DN is realized as follows:

$$\hat{x}'_i = \frac{\hat{x}_i}{\sqrt{\sum_{i \in S'_i} |\hat{x}_i|^2 + \varepsilon}}, \tag{3}$$

where S'_i (green region in Fig. 2) includes C elements from the same example (N axis) and the same spatial location (H, W axis). $\hat{x}_i$ is computed as Eq. (1) and (2) with elements in S_i from the same channel and sample (blue region in Fig. 2). In DN, besides normalization across spatial dimension, we also employ L_2 normalization to normalize features along the channel axis. They collaborate with each other to address the address the sensitivity to domain shift as well as stress noises and extreme values in feature vectors. As illustrated in Fig. 3, it helps regulate the norm distribution of the features in different datasets and improves the robustness to local domain shifts (*e.g.* texture pattern, noise, contrast).

Finally, the trainable per-channel scale γ and shift β are added to enhance the discriminative representation ability as BN and IN. The final formulation is as follows:

$$y_i = \gamma_i \, \hat{x}'_i + \beta_i. \tag{4}$$

3.2. Non-local Aggregation

We propose a graph-based filter that robustly exploits non-local contextual information and reduces the dependence on local patterns (see Fig. 1(c)) for domain-invariant stereo matching.

3.2.1 Formulation

Our inspiration comes from traditional graph-based filters that are remarkably effective in employing non-local structural information for structure-preserving texture and detail removing/smoothing [55], denoising [5, 55], as well as depth-aware estimation and enhancement [26, 52].

For a 2D image/feature map I, we construct an 8-connected graph by connecting pixel $\mathbf{p}$ to its eight neighbors (see Fig. 4). To avoid loops and achieve fast non-local information aggregation over the graph, we split it into two reverse directed graphs G_1, G_2 (see Fig. 4(b) and 4(c)).

We assign weight ω_e to each edge $e \in G$, and a feature (or color) vector $C(\mathbf{p})$ to each node $\mathbf{p} \in G$. We also allow $\mathbf{p}$ to propagate information to itself with weight $\omega_e(\mathbf{p},\mathbf{p})$. For graph G_i ($i = 0, 1$), our non-local filter is defined as follows:

$$\begin{aligned} C_i^A(\mathbf{p}) &= \frac{\sum_{\mathbf{q} \in G_i} W(\mathbf{q},\mathbf{p}) \cdot C(\mathbf{q})}{\sum_{\mathbf{q} \in G_i} W(\mathbf{q},\mathbf{p})}, \\ W(\mathbf{q},\mathbf{p}) &= \sum_{l_{\mathbf{q},\mathbf{p}} \in G_i} \prod_{e \in l_{\mathbf{q},\mathbf{p}}} \omega_e. \end{aligned} \tag{5}$$

Here, $l_{\mathbf{q},\mathbf{p}}$ is a feasible path from $\mathbf{q}$ to $\mathbf{p}$. Note that $e(\mathbf{q},\mathbf{q})$ is included in the path and counts for the start node $\mathbf{q}$. Unlike traditional geodesic filters, we consider all valid paths from source node $\mathbf{q}$ to target node $\mathbf{p}$. The propagation weight along path $l_{\mathbf{q},\mathbf{p}}$ is the product of all edge weights ω_e along the path. Here weight $W(\mathbf{q},\mathbf{p})$ is defined as the sum of the weights of all feasible paths from $\mathbf{q}$ to $\mathbf{p}$, which determines how much information is diffused to $\mathbf{p}$ from $\mathbf{q}$.

For the edge weight $\omega_{(\mathbf{q},\mathbf{p})}$, we define it in a self-regularized manner as follows:

$$\omega_e(\mathbf{q},\mathbf{p}) = \frac{\mathbf{x_p}^T \mathbf{x_q}}{\|\mathbf{x_p}\|_2 \cdot \|\mathbf{x_q}\|_2}, \tag{6}$$

where $\mathbf{x_p}$ and $\mathbf{x_q}$ represent the feature vectors of $\mathbf{p}$ and $\mathbf{q}$, respectively. This definition does not introduce new parameters and thus is more robust to cross-domain generalization.

Compared to other local filters, such as Gaussian filter, median filter, and mean filter that can only propagate information in a local region determined by the filter kernel size, our proposed non-local filter allows the propagation of long-range information with weights as a spatial accumulation along all feasible paths in a graph.

For stable training and to avoid extreme values, we further add a normalization constraint to the weights associated with $\mathbf{p}$ in the graph G_i as:

$$\sum_{\mathbf{q} \in N_\mathbf{p}} \omega_{e(\mathbf{q},\mathbf{p})} = 1. \tag{7}$$

Here, $N_\mathbf{p}$ is the set of the connected neighbors of $\mathbf{p}$ (including itself), and $e(\mathbf{q},\mathbf{p})$ is the directed edge connecting $\mathbf{q}$ and $\mathbf{p}$. For example, in Fig. 4(b), for node $\mathbf{p}_0$, $\omega_{e(\mathbf{p}_0,\mathbf{p}_0)} = 1$; and for node $\mathbf{p}_4$, $\omega_{0,4} + \omega_{1,4} + \omega_{e(\mathbf{p}_4,\mathbf{p}_4)} = 1$.

If Eq. (7) holds, we can further derive $\sum_{\mathbf{q} \in G_i} W(\mathbf{q},\mathbf{p}) =$

1*. Eq. (5) can then be simplified as follows:

$$\begin{aligned} C_i^A(\mathbf{p}) &= \sum_{\mathbf{q}\in G_i} W(\mathbf{q},\mathbf{p})\cdot C(\mathbf{q}), \\ W(\mathbf{q},\mathbf{p}) &= \sum_{l_{\mathbf{q},\mathbf{p}}\in G_i} \prod_{e\in l_{\mathbf{q},\mathbf{p}}} \omega_e. \end{aligned} \tag{8}$$

Such a transformation not only increases the robustness in training but also reduces the computational costs.

3.2.2 Linear Implementation

Eq. (8) can be realized as an iterative linear aggregation, where the node representation is sequentially updated following the direction of the graph (*e.g.* from top to bottom, then left to right in G_1). In each step, $\mathbf{p}$ is updated as:

$$\begin{aligned} C_i^A(\mathbf{p}) &= \omega_{e(\mathbf{p},\mathbf{p})}\cdot C(\mathbf{p}) + \sum_{\mathbf{q}\in N_\mathbf{p},\mathbf{q}\neq\mathbf{p}} \omega_{e(\mathbf{q},\mathbf{p})}\cdot C_i^A(\mathbf{q}) \\ s.t. &\quad \sum_{\mathbf{q}\in N_\mathbf{p}} \omega_{e(\mathbf{q},\mathbf{p})} = 1. \end{aligned} \tag{9}$$

Finally, we repeat the aggregation process for both G_1 and G_2 where the updated representation with G_1 is used as the input for aggregation with G_2 (similar to patchmatch stereo [2]). The aggregation of Eq. (9) is a linear process with time complexity of $O(n)$ (with n nodes in the graph). During training, backpropagation can be realized by reversing the propagation equation which is also a linear process (*available in the supplementary material*).

3.2.3 Relations to Existing Approaches

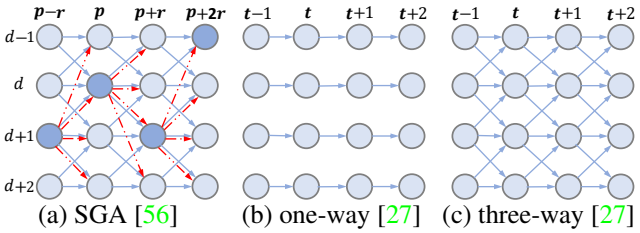
We show that the recently proposed semi-global aggregation (SGA) layer [56] and affinity-based propagation approach [27] are special cases of our graph-based non-local filter (Eq. (8)). In addition, we compare it with non-local neural networks [48, 50] and the attention mechanism [15].

Semi-global Aggregation (SGA) [56] is proposed as a differentiable approximation of SGM [13] and can be presented as follows:

$$\begin{aligned} C_\mathbf{r}^A(\mathbf{p},d) &= \text{sum}\begin{cases} \omega_0(\mathbf{p},\mathbf{r})\cdot C(\mathbf{p},d) \\ \omega_1(\mathbf{p},\mathbf{r})\cdot C_\mathbf{r}^A(\mathbf{p}-\mathbf{r},d) \\ \omega_2(\mathbf{p},\mathbf{r})\cdot C_\mathbf{r}^A(\mathbf{p}-\mathbf{r},d-1) \\ \omega_3(\mathbf{p},\mathbf{r})\cdot C_\mathbf{r}^A(\mathbf{p}-\mathbf{r},d+1) \\ \omega_4(\mathbf{p},\mathbf{r})\cdot \max_i C_\mathbf{r}^A(\mathbf{p}-\mathbf{r},i). \end{cases} \\ s.t. &\quad \sum_{i=0,1,2,3,4} \omega_i(\mathbf{p},\mathbf{r}) = 1 \end{aligned} \tag{10}$$

The aggregations are done in four directions, namely $\mathbf{r} = \{(0,1),(0,-1),(1,0),(-1,0)\}$. Taking the right to left propagation ($\mathbf{r} = (0,1)$) as an example, we can construct a propagation graph in Fig. 5(a). The y-coordinate represents disparity d, and the x-coordinate represents the indexes of the pixels/nodes. Compared to our non-local graph in Fig. 4(b), edges connecting top and bottom nodes are removed, and the maximum of each column is densely connected to every node of the next column (red edges). The SGA layer can then be realized by our proposed non-local filter in Eq. (8). Here, $(\mathbf{p}-\mathbf{r}, d\pm 1)$ are the neighborhood nodes of $\mathbf{p}$, and $\omega_{0,\ldots,4}$ are the corresponding edge weights.

*The proof is available in the supplementary material.



(a) SGA [56] (b) one-way [27] (c) three-way [27]

Figure 5: Special cases of our non-local filter. (a) Semi-global aggregation (SGA) layer [56]. The dark blue node represents the maximum of each column. (b) and (c) are the affinity-based spatial propagations [27]. They aggregate from column t to $t+1$.

The **Affinity-based Spatial Propagation** in [27] can be achieved as:

$$\begin{aligned} C^A(\mathbf{p},d) &= \left(1 - \sum_{\mathbf{q}\in N_\mathbf{p},\mathbf{q}\neq\mathbf{p}} \omega_{e(\mathbf{q},\mathbf{p})}\right) C(\mathbf{p}) \\ &+ \sum_{\mathbf{q}\in N_\mathbf{p},\mathbf{q}\neq\mathbf{p}} \omega_{e(\mathbf{q},\mathbf{p})} C^A(\mathbf{q}), \end{aligned} \tag{11}$$

where $\omega_{e(\mathbf{q},\mathbf{p})}$ are the learned affinities. $1-\sum_{\mathbf{q}\in N_\mathbf{p}} \omega_{e(\mathbf{q},\mathbf{p})}$ is equal to our weight $\omega_{\mathbf{e}(\mathbf{p},\mathbf{p})}$ for $\mathbf{p}$. The graphs for filtering can be constructed as in Fig. 5(b) and 5(c) for the one-way and three-way propagations [27], respectively.

The **Non-local Neural Networks and Attentions** [15, 48, 50] are implemented without spatial and structural awareness. The similarity definition between two pixels only considers the feature differences without considering their spatial distances. Therefore, they will easily smooth out depth edges and thin structures (as illustrated in the supplementary material). Our non-local filter spatially aggregates the message along the paths in the graph which can avoid over smoothness and better preserve the structure of the disparity maps.

3.3. Network Architecture

As illustrated in Fig. 6, we utilize the backbone of GANet as the baseline architecture. The local guided aggregation layer in [56] is removed since it's domain-dependent and captures a lot of local patterns that are very sensitive to local domain shifts.

We replace the original batch normalization layer by our proposed domain normalization layer for feature extraction. For the feature extraction network, we utilize a total of seven proposed filtering layers. For 3D cost aggregation of the cost volume, two non-local filters are further added for cost volume filtering in each channel/depth. All the details of the network architecture are presented in Table I in the supplementary material.

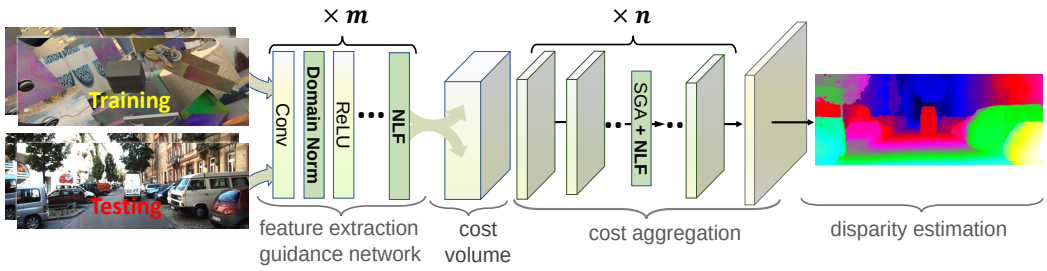


Figure 6: Overview of the network architecture. Synthetic data are used for training, while using data from other new domains (*e.g.* real KITTI dataset) for testing. The backbone of the state-of-the-art GANet [56] is used as the baseline. The proposed domain normalization is used after each convolutional layer in the feature extraction and guidance network. Several non-local filter layers are implemented for both feature extraction and cost aggregation.

4. Experimental Results

In our experiments, we train our method only with synthetic data and test it on four real datasets to evaluate its domain generalization ability. During training, we use disparity regression [17] for disparity prediction, and the smooth L_1 loss to compute the errors for back-propagation (the same as in [4,56]). All the models are optimized with Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$). We train with a batch size of 8 on four GPUs using 288×624 random crops from the input images. The maximum of the disparity is set as 192. We train the model on the synthetic dataset for 10 epochs with a constant learning rate of 0.001. All other training settings are kept the same as those in [56].

4.1. Datasets

KITTI stereo 2012 [9] and 2015 [31] datasets provide about 400 image pairs of outdoor driving scenes for training, where the disparity labels are transformed from Velodyne LiDAR points. The **Cityscapes** dataset [7] provides a large amount of high-resolution ($1k \times 2k$) stereo images collected from out-door city driving scenes. The disparity labels are pre-computed by SGM [13] which is not accurate enough for training deep neural network models. The **Middlebury** stereo dataset [40] is designed for indoor scenes with higher resolution (up to $2k \times 3k$). But it provides no more than 50 image pairs that are not enough to train robust deep neural networks. In addition, **ETH 3D** dataset [41] provides 27 pairs of gray images for training.

These existing real datasets are all limited by their small quantity or poor ground-truth labels, making them insufficient for training deep learning models. Hence, we just use them as test sets for evaluating our models' cross-domain generalization ability.

We mainly use synthetic data to train our domain-invariant models. The existing Scene Flow synthetic dataset [30] contains 35k training image pairs with a resolution of 540×960. This dataset has a limited number of the out-door driving scenes that provide stereo pairs with a few settings of the camera baselines and image resolutions. We use CARLA [8] to generate a new supplementary synthetic dataset (with 20k stereo pairs) with more diverse settings, including two kinds of image resolutions (720×1080 and 1080×1920), three different focal lengths, and five different camera baselines (in a range of 0.2-1.5m). This supplementary dataset can significantly improve the diversity of the training set (*which will be published with the paper*).

The two advantages in using synthetic data are that it can avoid all the difficulties of labeling a large amount of real data, and that it can eliminate the negative influence of wrong depth values in real datasets.

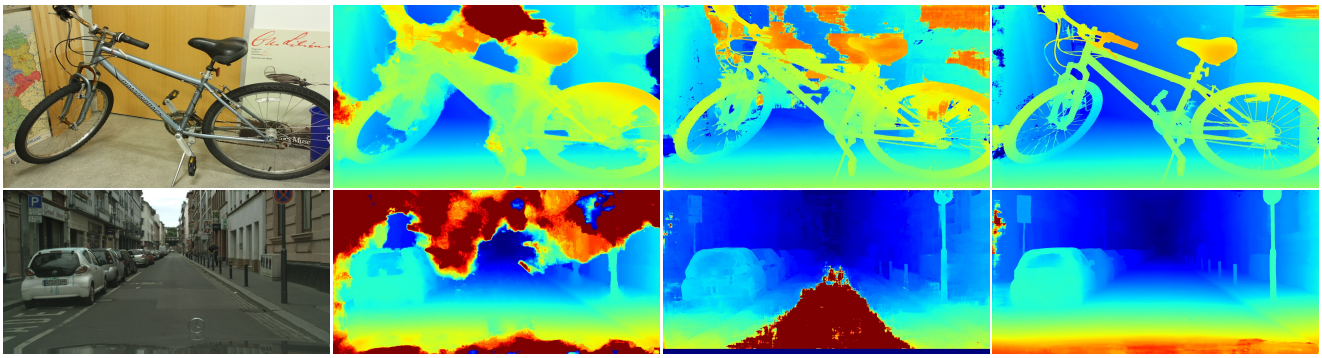
Table 1: Ablation study. Models are trained on synthetic data (SceneFlow). Threshold error rates (%) are used for evaluations.

Normlize	Non-local Filter		Backbone	Midd	KITTI
	feature	cost volume		3-pixel	2-pixel
BN			ours	30.3	9.4
DN			ours	27.1	7.9
DN	+3		ours	24.2	7.1
DN	+7		ours	22.9	6.8
DN	+9		ours	22.4	6.8
DN	+7	+2	ours	**21.8**	**6.5**
BN			PSMNet	39.5	16.3
BN			GANet	32.2	11.7
DN	+7	+2	PSMNet	26.1	8.5
DN	+7	+2	GANet	23.7	7.3

4.2. Ablation Study

We evaluate the performance of our DSMNet with numerous settings, including different architectures, normalization strategies and numbers (0-9) of the proposed non-local filter (NLF) layers. As listed in Table 1, the full-setting DSMNet far outperforms the baseline in accuracy by 3% on the KITTI and 8% on the Middlebury datasets. Our proposed domain normalization improves the accuracy by about 1.5%, and the NLF layers contribute another 1.4% on the KITTI dataset.

Moreover, our proposed layers are generic and could be seamlessly integrated into other deep stereo matching models. Here, we replace our backbone model with GANet [56] and PSMNet [4]. The accuracies are improved by 4∼8% on KIITTI dataset and 8∼13% on Middlebury dataset for coss-domain evaluations compared with the original PSMNet and GANet.



(a) Input view (b) HD^3 [53] (c) PSMNet [4] (d) Our DSMNet

Figure 7: Comparisons with state-of-the-art models. Models are trained on synthetic data and evaluated on high-resolution real datasets (Middlebury and CityScapes). Our DSMNet can produce much more accurate disparity estimation. (See supplementary for more results.

Table 2: Comparisons with Existing Normalization and Filtering/Attention Strategies

Models	Middlebury (full)	KITTI
Batch Norm	29.1	7.3
Instance Norm	27.1	6.4
Adaptive Norm [33]	28.2	6.8
Attention [15]	25.2	5.9
Feature Denoising [50]	25.9	6.1
Affinity [27]	23.1	5.2
DSMNet (full setting)	20.1	4.1

4.3. Component Analysis and Comparisons

To further validate the superiorities of the proposed layers , we compare each of them with other related normalization and non-local strategies.

Normalization Strategies. Table 2 compares our domain normalization with batch normalization [16], instance normalization [47], and the recently proposed adaptive batch-instance normalization [33]. We keep all other settings the same as our DSMNet and only replace the normalization method for training and evaluation. Our domain normalization is superior to others for domain-invariant stereo matching because it can fully regulate the feature vectors' distribution and remove both image-level and local contrast differences for cross-domain generalization.

Non-local Approaches. Finally, we compare our graph-based non-local filter with other related strategies, including affinity-based propagation [27], non-local neural network denoising [50], and non-local attention [15] (in Table 2). 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 [50] and non-local attention [15] do not have spatial constraints that usually lead to over smoothness of the depth edges (as shown in the supplementary material). Affinity-based propagations [27] are special cases of our proposed filtering strategy and are not as effective in feature and cost volume aggregations for stereo matching.

Table 3: Evaluations on the KITTI, Middlebury, and ETH 3D validation datasets. Threshold error rates (%) are used.

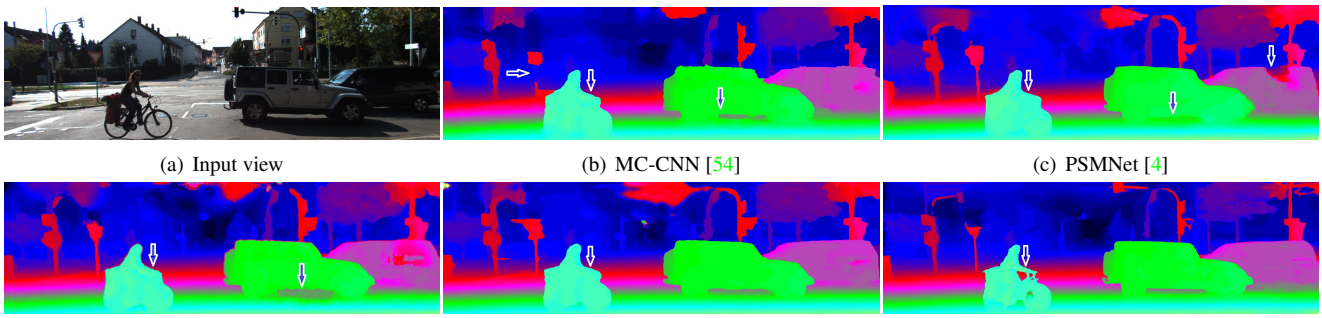
Models	KITTI		Middlebury			ETH3D	Carla
	2012	2015	full	half	quarter		
CostFilter [14]	21.7	18.9	57.2	40.5	17.6	31.1	41.1
PatchMatch [2]	20.1	17.2	50.2	38.6	16.1	24.1	30.1
SGM [13]	7.1	7.6	38.1	25.2	10.7	12.9	20.2
Training set	SceneFlow						
HD^3 [53]	23.6	26.5	50.3	37.9	20.3	54.2	35.7
gwcnet [12]	20.2	22.7	47.1	34.2	18.1	30.1	33.2
PSMNet [4]	15.1	16.3	39.5	25.1	14.2	23.8	25.9
GANet [56]	10.1	11.7	32.2	20.3	11.2	14.1	18.8
Our DSMNet	**6.2**	**6.5**	**21.8**	**13.8**	**8.1**	**6.2**	**9.8**
Training set	SceneFlow + Carla						
HD^3 [53]	19.1	19.5	47.3	35.2	19.5	45.2	–
gwcnet [12]	17.2	18.1	45.2	31.8	17.2	29.4	–
PSMNet [4]	10.3	11.0	35.5	23.7	13.8	20.3	–
GANet [56]	7.2	7.6	31.9	19.7	11.4	13.5	–
Our DSMNet	**3.9**	**4.1**	**20.1**	**13.6**	**8.2**	**6.0**	–

4.4. Cross-Domain Evaluations

In this section, we compare our proposed DSMNet with state-of-the-art stereo matching models by training with synthetic data and evaluating on real test sets.

Comparisons with State-of-the-Art Models. In Table 3 and Fig. 7, we compare our DSMNet with other state-of-the-art deep neural network models on the four real datasets. All the models are trained on synthetic data (either SceneFlow or a mixture of SceneFlow and Carla). We find that DSMNet far outperforms the state-of-the-art models by 3$\sim$30% in error rates on all these datasets. It is also far better than traditional stereo matching algorithms, like SGM [13], costfilter [14] and patchmatch [2].

Evaluation on the KITTI Benchmark. Table 4 presents the performance of our DSMNet on the KITTI benchmark [31]. Our model far outperforms most of the unsupervised/self-supervised models trained on the KITTI domain. It is even better than supervised stereo matching networks (including, MC-CNN [54], content-CNN [29],



(a) Input view (b) MC-CNN [54] (c) PSMNet [4]

(d) HD^3 [53] (e) GANet-deep [56] (f) Our DSMNet-synthetic

Figure 8: Comparisons with the fine-tuned state-of-the-art models. Our model is trained only with synthetic data. All others are fine-tuned on the KITTI target scenes. As pointed by arrows, our DSMNet can produce more accurate object boundaries.

Table 4: Evaluation on KITTI 2015 Benchmark

Models	Training Set	Error Rates (%)
Our DSMNet	**Synthetic**	**3.71**
MC-CNN-acrt [54]	Kitti-gt	3.89
DispNetC [30]	Kitti-gt	4.34
Content-CNN [29]	Kitti-gt	4.54
MADNet-finetune [45]	Kitti-gt	4.66
Weak Supervise [46]	Kitti-gt	4.97
MADNet [45]	Kitti (no gt)	8.23
OASM-Net [19]	Kitti (no gt)	8.98
Unsupervised [59]	Kitti (no gt)	9.91

Table 5: Evaluation on the KITTI 2015 Benchmark (Fine-tuning)

Models	Non-Occluded	All Area
GANet + **Our NLF**	**1.58**	**1.77**
GANet-deep [56]	1.63	1.81
DSMNet-finetune	1.71	1.90
GANet-15 [56]	1.73	1.93
HD^3 [53]	1.87	2.02
gwcnet-g [12]	1.92	2.11
PSMNet [4]	2.14	2.32
GCNet [17]	2.61	2.87

and DispNetC [30]) trained or fine-tuned on the KITTI dataset. When compared with other fine-tuned state-of-the-art models (*e.g.* PSMNet [4], HD^3 [53], GANet-deep [56]), our DSMNet (without fine-tuning) produces more accurate object boundaries (Fig. 8).

4.5. Fine-tuning

In this section, we show DSMNet's best performance when fine-tuned on the target domain. We fine-tune the model pre-trained on synthetic data for a further 700 epochs using the KITTI 2015 training set. The learning rate for fine-tuning begins at 0.001 for the first 300 epochs and decreases to 0.0001 for the rest. The results are submitted to the KITTI benchmarks for evaluations.

Table 5 compares the results of the fine-tuned DSMNet and those of other state-of-the-art DNN models. We find that DSMNet outperforms most of the recent models (including PSMNet [4], HD^3 [53], GwcNet [12] and GANet-15 [56]) by a noteworthy margin. This implies that DSMNet can achieve the same accuracy by fine-tuning on one specific dataset, without sacrificing accuracy to improve its cross-domain generalization ability.

We also separately test the effectiveness of our non-local filtering strategy. Using the current best "GANet-deep" [56] (including the Local Guided Aggregation layer) as the baseline, we add five filtering layers for feature extraction. All other settings are kept the same as the original GANet. After training on synthetic data and fine-tuning on the KITTI training dataset, the model gets a new state-of-the-art accuracy (1.77%) on KITTI 2015 benchmark. This shows that our graph-based filter can improve not only cross-domain generalization but also the accuracy on the test domains.

4.6. Efficiency and Parameters

Our proposed non-local filtering is a linear process that can be realized efficiently. The inference time is increased slightly by no more than 5% compared with the baseline. Moreover, no any new parameter is introduced for the proposed domain normalization and non-local filtering layers. *Detailed comparisons are available in the supplementary material.*

5. Conclusion

In this paper, we have proposed two end-to-end trainable neural network layers for our domain-invariant stereo matching network. Our novel domain normalization can fully regulate the distribution of learned features to address significant domain shifts, and our non-local graph-based filter can capture more robust non-local structural and geometric features for accurate disparity estimation in cross-domain situations. We have verified our model on four real datasets and have shown its superior accuracy when compared to other state-of-the-art stereo matching networks in cross-domain generalization.

References

[1] Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. Metareg: Towards domain generalization using meta-regularization. In *Advances in Neural Information Processing Systems (NIPS)*, pages 998–1008, 2018. 3

[2] Michael Bleyer, Christoph Rhemann, and Carsten Rother. Patchmatch stereo-stereo matching with slanted support windows. In *British Machine Vision Conference (BMVC)*, pages 1–11, 2011. 5, 7

[3] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 3722–3731, 2017. 1

[4] 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. 1, 2, 3, 6, 7, 8, 13, 14, 16

[5] Xiaogang Chen, Sing Bing Kang, Jie Yang, and Jingyi Yu. Fast patch-based denoising using approximated patch geodesic paths. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1211–1218, 2013. 4

[6] Christopher B Choy, JunYoung Gwak, Silvio Savarese, and Manmohan Chandraker. Universal correspondence network. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2414–2422, 2016. 3

[7] 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. 6, 14

[8] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. *arXiv preprint arXiv:1711.03938*, 2017. 6, 13

[9] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361. IEEE, 2012. 6

[10] Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. Domain generalization for object recognition with multi-task autoencoders. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, pages 2551–2559, 2015. 3

[11] Xiaoyang Guo, Hongsheng Li, Shuai Yi, Jimmy Ren, and Xiaogang Wang. Learning monocular depth by distilling cross-domain stereo networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 484–500, 2018. 1, 2

[12] Xiaoyang Guo, Kai Yang, Wukui Yang, Xiaogang Wang, and Hongsheng Li. Group-wise correlation stereo network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3273–3282, 2019. 2, 7, 8

[13] Heiko Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):328–341, 2008. 1, 2, 5, 6, 7

[14] Asmaa Hosni, Christoph Rhemann, Michael Bleyer, Carsten Rother, and Margrit Gelautz. Fast cost-volume filtering for visual correspondence and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(2):504–511, 2013. 7

[15] 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, 7, 13, 17

[16] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 7

[17] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-end learning of geometry and context for deep stereo regression. *CoRR, vol. abs/1703.04309*, 2017. 1, 2, 3, 6, 8

[18] Sameh Khamis, Sean Ryan Fanello, Christoph Rhemann, Adarsh Kowdle, Julien P. C. Valentin, and Shahram Izadi. Stereonet: Guided hierarchical refinement for real-time edge-aware depth prediction. *CoRR*, abs/1807.08865, 2018. 2

[19] Ang Li and Zejian Yuan. Occlusion aware stereo matching via cooperative unsupervised learning. In *Asian Conference on Computer Vision*, pages 197–213. Springer, 2018. 8

[20] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 5542–5550, 2017. 3

[21] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 3

[22] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5400–5409, 2018. 3

[23] Ya Li, Xinmei Tian, Mingming Gong, Yajing Liu, Tongliang Liu, Kun Zhang, and Dacheng Tao. Deep domain generalization via conditional invariant adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 624–639, 2018. 3

[24] Yanghao Li, Naiyan Wang, Jianping Shi, Xiaodi Hou, and Jiaying Liu. Adaptive batch normalization for practical domain adaptation. *Pattern Recognition*, 80:109–117, 2018. 3

[25] Zhengfa Liang, Yiliu Feng, Yulan Guo, Hengzhu Liu, Wei Chen, Linbo Qiao, Li Zhou, and Jianfeng Zhang. Learning for disparity estimation through feature constancy. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2811–2820, 2018. 2

[26] Ming-Yu Liu, Oncel Tuzel, and Yuichi Taguchi. Joint geodesic upsampling of depth images. In *Proceedings of the*

IEEE conference on computer vision and pattern recognition (CVPR), pages 169–176, 2013. 4

[27] Sifei Liu, Shalini De Mello, Jinwei Gu, Guangyu Zhong, Ming-Hsuan Yang, and Jan Kautz. Learning affinity via spatial propagation networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1520–1530, 2017. 5, 7

[28] Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. Understanding the effective receptive field in deep convolutional neural networks. In *Advances in neural information processing systems (NIPS)*, pages 4898–4906, 2016. 1

[29] Wenjie Luo, Alexander G Schwing, and Raquel Urtasun. Efficient deep learning for stereo matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5695–5703, 2016. 2, 7, 8

[30] 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. 2, 3, 6, 8, 13, 14

[31] 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. 2, 6, 7, 14

[32] Saeid Motiian, Marco Piccirilli, Donald A Adjeroh, and Gianfranco Doretto. Unified deep supervised domain adaptation and generalization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 5715–5725, 2017. 3

[33] Hyeonseob Nam and Hyo-Eun Kim. Batch-instance normalization for adaptively style-invariant neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2558–2567, 2018. 3, 7

[34] Guang-Yu Nie, Ming-Ming Cheng, Yun Liu, Zhengfa Liang, Deng-Ping Fan, Yue Liu, and Yongtian Wang. Multi-level context ultra-aggregation for stereo matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3283–3291, 2019. 2

[35] Xingang Pan, Ping Luo, Jianping Shi, and Xiaoou Tang. Two at once: Enhancing learning and generalization capacities via ibn-net. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 464–479, 2018. 3

[36] Jiahao Pang, Wenxiu Sun, Jimmy SJ. Ren, Chengxi Yang, and Qiong Yan. Cascade residual learning: A two-stage convolutional neural network for stereo matching. *IEEE International Conference on Computer Vision Workshops (ICCVW)*, 2017. 2

[37] Jiahao Pang, Wenxiu Sun, Chengxi Yang, Jimmy Ren, Ruichao Xiao, Jin Zeng, and Liang Lin. Zoom and learn: Generalizing deep stereo matching to novel domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2070–2079, 2018. 2

[38] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2337–2346, 2019. 3

[39] Matteo Poggi, Davide Pallotti, Fabio Tosi, and Stefano Mattoccia. Guided stereo matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 979–988, 2019. 2

[40] 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. 6, 14

[41] Thomas Schops, Johannes L Schonberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3260–3269, 2017. 6

[42] Xiao Song, Xu Zhao, Liangji Fang, and Hanwen Hu. Edgestereo: An effective multi-task learning network for stereo matching and edge detection. *arXiv preprint arXiv:1903.01700*, 2019. 2, 3

[43] Alessio Tonioni, Matteo Poggi, Stefano Mattoccia, and Luigi Di Stefano. Unsupervised adaptation for deep stereo. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 2

[44] Alessio Tonioni, Oscar Rahnama, Thomas Joy, Luigi Di Stefano, Thalaiyasingam Ajanthan, and Philip HS Torr. Learning to adapt for stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9661–9670, 2019. 2

[45] Alessio Tonioni, Fabio Tosi, Matteo Poggi, Stefano Mattoccia, and Luigi Di Stefano. Real-time self-adaptive deep stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 195–204, 2019. 1, 2, 3, 8

[46] Stepan Tulyakov, Anton Ivanov, and Francois Fleuret. Weakly supervised learning of deep metrics for stereo reconstruction. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1339–1348, 2017. 8

[47] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 7

[48] 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. 5

[49] Yan Wang, Zihang Lai, Gao Huang, Brian H. Wang, Laurens Van Der Maaten, Mark Campbell, and Kilian Q Weinberger. Anytime stereo image depth estimation on mobile devices. *arXiv preprint arXiv:1810.11408*, 2018. 2

[50] 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. 5, 7, 13, 17

[51] Guorun Yang, Hengshuang Zhao, Jianping Shi, Zhidong Deng, and Jiaya Jia. Segstereo: Exploiting semantic information for disparity estimation. *arXiv preprint arXiv:1807.11699*, 2018. 2

[52] Qingxiong Yang. A non-local cost aggregation method for stereo matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1402–1409. IEEE, 2012. 4

[53] 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. 1, 2, 7, 8, 14, 16

[54] Jure Zbontar and Yann LeCun. Computing the stereo matching cost with a convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1592–1599, 2015. 1, 2, 7, 8

[55] Feihu Zhang, Longquan Dai, Shiming Xiang, and Xiaopeng Zhang. Segment graph based image filtering: fast structure-preserving smoothing. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 361–369, 2015. 4

[56] 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. 1, 2, 3, 5, 6, 7, 8, 13, 14

[57] Feihu Zhang and Benjamin W Wah. Fundamental principles on learning new features for effective dense matching. *IEEE Transactions on Image Processing*, 27(2):822–836, 2018. 2

[58] Yiran Zhong, Yuchao Dai, and Hongdong Li. Self-supervised learning for stereo matching with self-improving ability. *arXiv preprint arXiv:1709.00930*, 2017. 2

[59] Chao Zhou, Hong Zhang, Xiaoyong Shen, and Jiaya Jia. Unsupervised learning of stereo matching. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1567–1575, 2017. 1, 2, 8

Supplementary Material

A. Proof of Footnote 1

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

$$\sum_{\mathbf{q}\in G_i} W(\mathbf{q},\mathbf{p}) = 1, \quad \text{if} \sum_{\mathbf{q}\in N_\mathbf{p}} \omega_{e(\mathbf{q},\mathbf{p})} = 1. \tag{12}$$

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. 4), we can prove Eq. (12) 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 $n\leq t$, $\sum_{\mathbf{q}\in G_i} W(\mathbf{q},\mathbf{p}_n) = 1$.
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, $k\leq t$, since $\mathbf{p}_k \in N_{\mathbf{p}_{t+1}}$.
This yields the equivalence of Eq. (12).

B. Backpropagation

The backpropagation for ω_e and $C(\mathbf{p})$ in Eq. (9) can be computed inversely. Assume the gradient from next layer is $\frac{\partial E}{\partial C_i^A}$. The backpropagation can be implemented as:

$$\begin{aligned}
\frac{\partial E}{\partial C(\mathbf{p})} &= \frac{\partial E}{\partial C_i^b(\mathbf{p})} \cdot \omega_{e(\mathbf{p},\mathbf{p})}, \\
\frac{\partial E}{\partial \omega_{e(\mathbf{p},\mathbf{p})}} &= \frac{\partial E}{\partial C_i^b(\mathbf{p})} \cdot C(\mathbf{p}), \\
\frac{\partial E}{\partial \omega_{e(\mathbf{q},\mathbf{p})}} &= \frac{\partial E}{\partial C_i^b(\mathbf{p})} \cdot C_i^A(\mathbf{q}), \quad \mathbf{q}\in N_\mathbf{p} \;\&\; \mathbf{q}\neq\mathbf{p}
\end{aligned} \tag{13}$$

where, $\frac{\partial E}{\partial C_\mathbf{i}^b}$ is a temporary gradient variable which can be calculated iteratively (similar to Eq. (9)):

$$\frac{\partial E}{\partial C_i^b(\mathbf{p})} = \frac{\partial E}{\partial C_i^A(\mathbf{p})} + \sum_{\mathbf{q}\in N_\mathbf{p},\mathbf{q}\neq\mathbf{p}} \frac{\partial E}{\partial C_i^b(\mathbf{q})} \cdot \omega_e(\mathbf{q},\mathbf{p}) \tag{14}$$

The propagation of Eq. (14) is an inverse process and in an order of $\mathbf{p}_N,\mathbf{p}_{N-1},...\mathbf{p}_0$

C. Details of the Architecture

Table 8 presents the details of the parameters of the DSMNet. It has seven non-local filtering 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.

D. Efficiency and Parameters

As shown in Table 6, our proposed non-local filtering is a linear process that can be realized efficiently. The inference time is increased by about 5% compared with the baseline. Moreover, no any new parameters are introduced for the proposed domain normalization and non-local filtering layers.

Table 6: Efficiency (Elapsed Time) and Number of Parameter

Methods	Elapsed Time	Parameter Number
GANet-deep [56]	1.8s	60M
Baseline	1.4s	48M
Our DSMNet	1.5s	48M
PSMNet [4]	0.4s	52M
DSMNet (PSMNet)	0.42s	52M

E. Carla Dataset

Since the synthetic Sceneflow dataset [30] only has limited number about 7,000 of stereo pairs for diving scenes, we use the Carla [8] platform to produce the stereo pairs for outdoor driving scenes. As shown in Table 7, 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. 9, the Carla scenes still have significant domain differences (*e.g.* color, textures) compared with the real scenes (*e.g.* KITTI, CityScapes), but, our DSMNet can extract shape and structure information for robust stereo matching. These can be better transferred to the real scenes and produce more accurate disparity estimation.

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

Figure 9: Example of the Carla stereo data.

F. More Results

F.1. Feature Visualization

As compared in Fig. 10, 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.

F.2. Disparity Results on Different Datasets

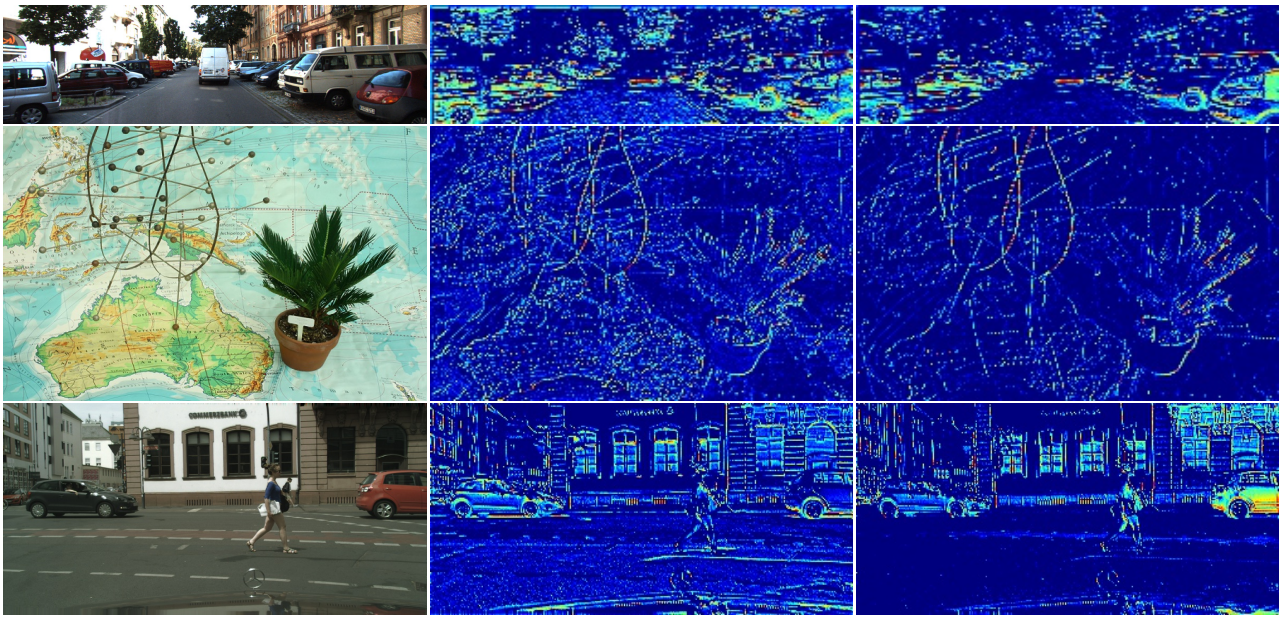
More results and comparisons are provided in Fig. 11. All the models are trained on the synthetic dataset and tested on the real KITTI, Middlebury, ETH3D and Cityscapes datasets.

F.3. Comparisons with Other Non-local Strategies

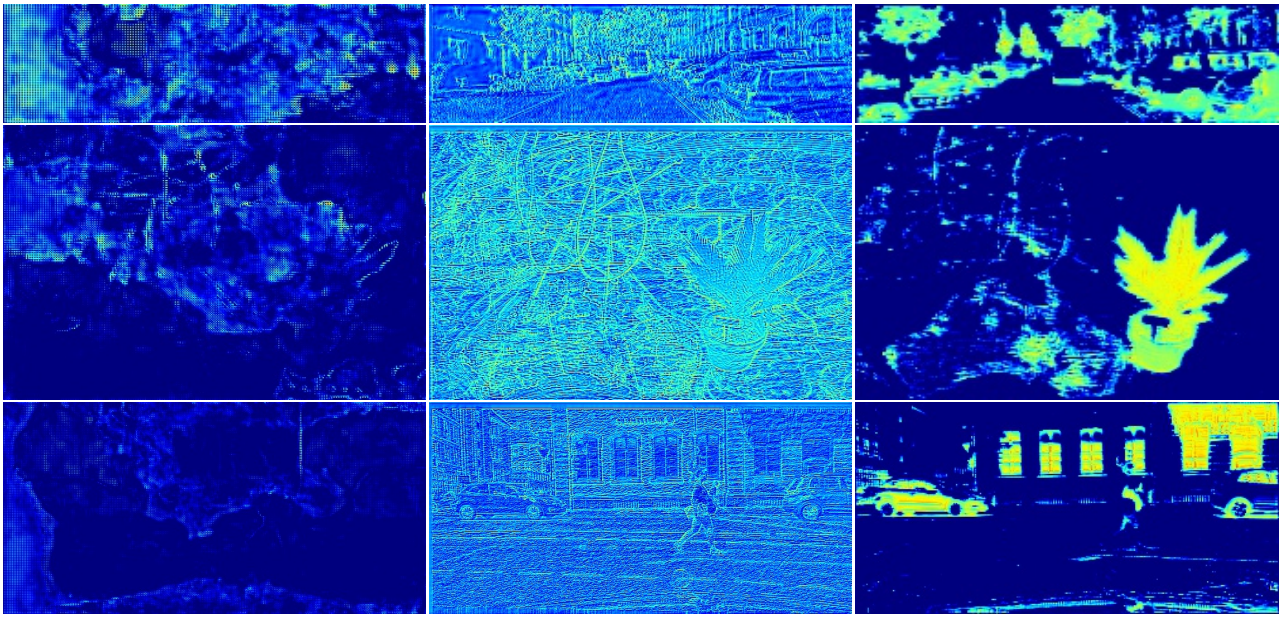
Our graph-based filtering strategy is better for capturing the structural and geometric context for robust domain-invariant stereo matching. The non-local neural network denoising [50] and non-local attention [15] do not have spatial constraints that usually lead to over smoothness of the depth edges and thin structures (as shown in Fig. 12).

Table 7: 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) Input view (b) GANet-synthetic (c) GANet-finetune

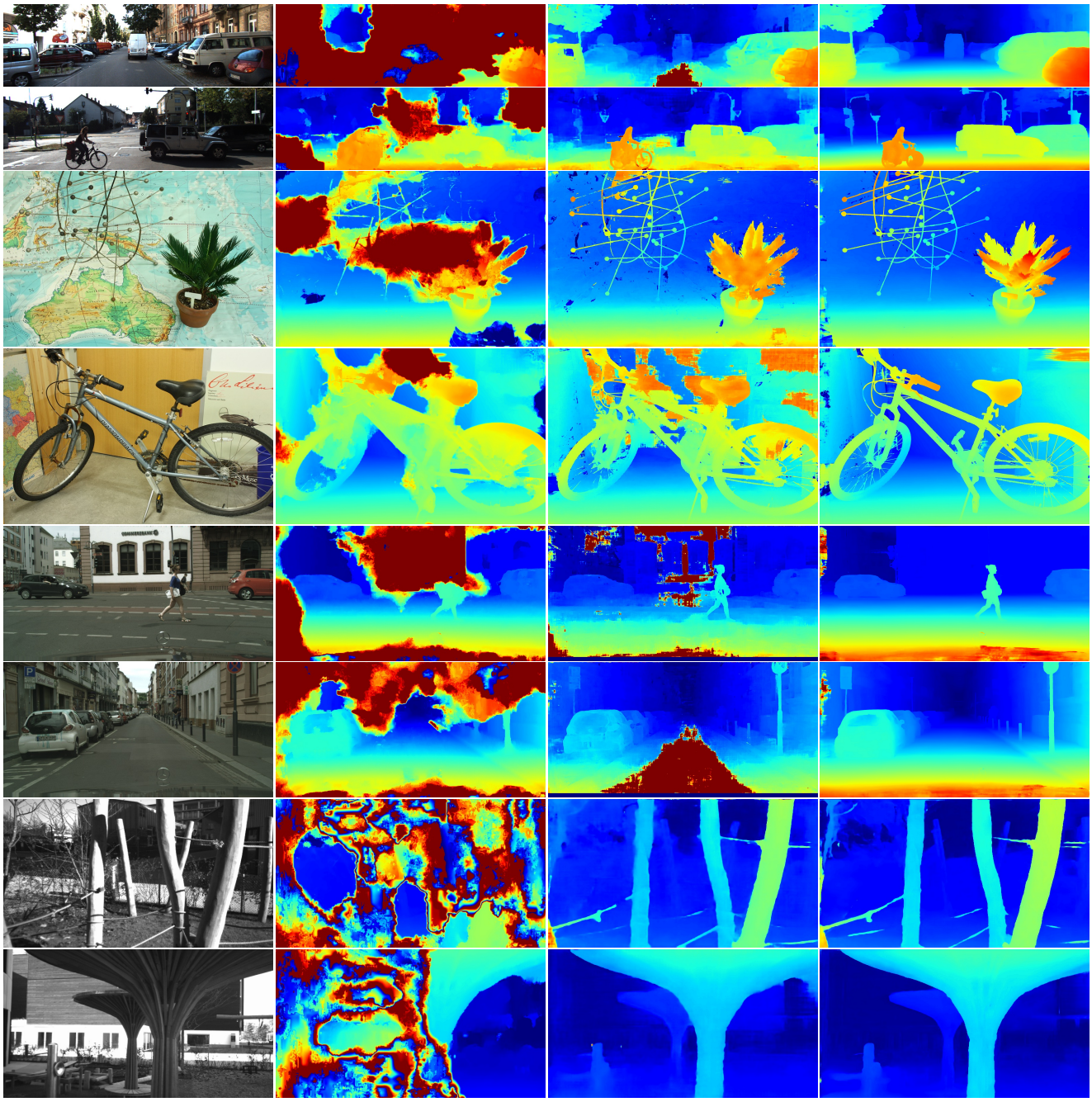


(d) HD^3-synthetic (e) PSMNet-synthetic (f) DSMNet-synthetic

Figure 10: Comparison and visualization of the feature maps for cross-domain test . (b) GANet [56], (d) HD^3 [53], (e) PSMNet [4] are trained on the synthetic dataset (Sceneflow [30]) and test on other real scenes/datasets (from top to bottom: Kitti [31], Middlebury [40] and CityScapes [7]). 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.

Table 8: 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	**NLF**, **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	**NLF**, **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	**NLF**, **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	**NLF**, **DN**, ReLU	$1/3$H×$1/3$W×32
concatenation	(11,14) (9,16) (7,18) (4,21) (20,24) (17,27) (15,29) (13,31) (18,25) (30,33) (28,35) (26,37) (23, 40)	
cost volume	by feature concatenation	$1/3$H×$1/3$W×64×32
Guidance Branch		
input	concate 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: weight matrices from (5)	$1/3$H×$1/3$W×64×32
[3]	**NLF**	$1/3$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]	3×3×3, 3D conv	$1/6$H×$1/6$W×32×48
[7]	SGA: weight matrices from (15)	$1/6$H×$1/6$W×32×48
[8]	3×3×3, 3D conv, stride 2	$1/12$H×$1/12$W×16×64
[9]	3×3×3, 3D deconv, stride 2	$1/6$H×$1/6$W×32×48
[10]	3×3×3, 3D conv	$1/6$H×$1/6$W×32×48
[11]	SGA: weight matrices from (18)	$1/6$H×$1/6$W×32×48
[12]	3×3×3, 3D deconv, stride 2	$1/3$H×$1/3$W×64×32
[13]	3×3×3, 3D conv	$1/3$H×$1/3$W×64×32
[14]	SGA: weight matrices from (8)	$1/3$H×$1/3$W×64×32
[15]	**NLF**	$1/3$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
[16 – 26]	repeat [5 – 15]	$1/3$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	concate: (4,12), (7,9), (8,19), (11,16), (15,23), (18,20); add: (1,4)	



(a) Input view (b) HD^3 [53] (c) PSMNet [4] (d) Our DSMNet

Figure 11: 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.

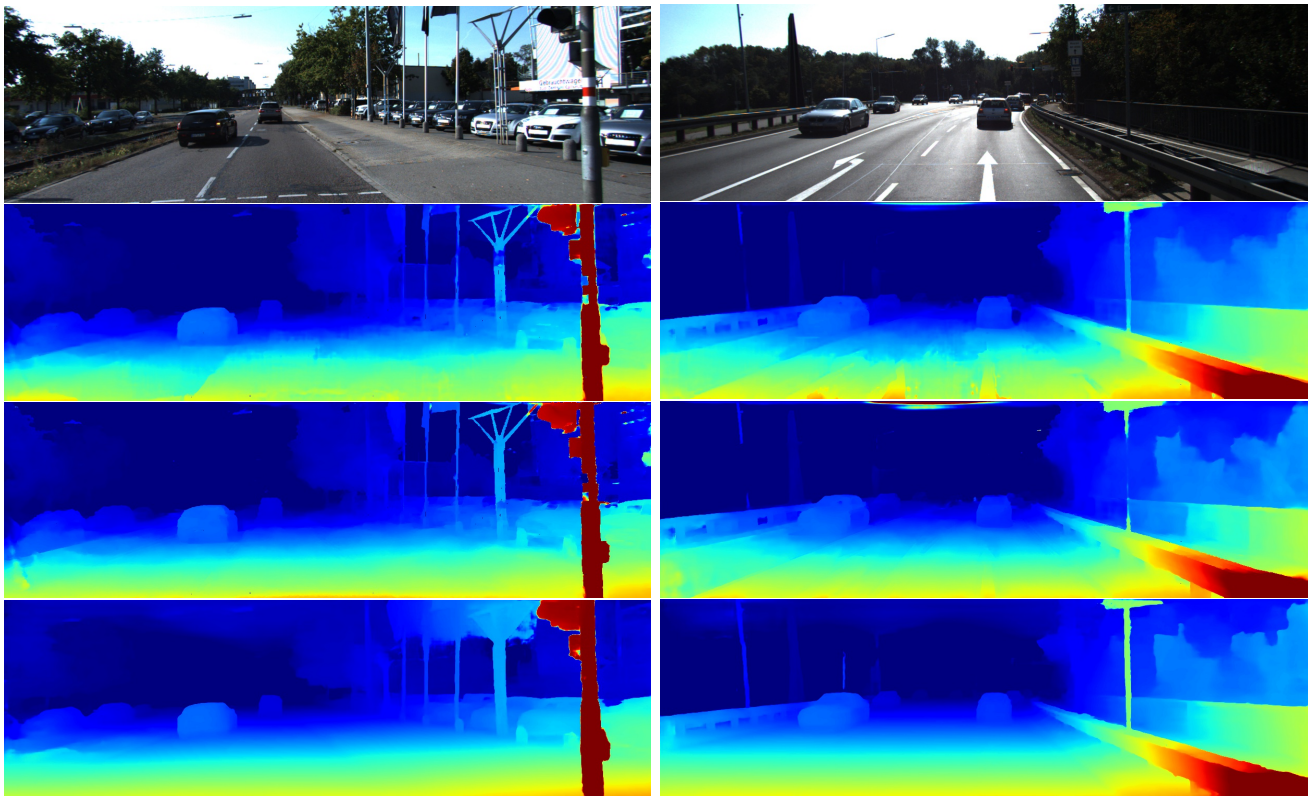


Figure 12: Comparisons with non-local attention mechanism [15] (*second row*) and non-local denoising [50] strategy (*third row*). When using these strategies, the thin structures (*e.g.* poles) are easily eroded by the background. These non-local strategies easily smooth out the disparity maps. As a comparison, our DSMNet (*last row*) can keep the thin structures of the disparity maps.