BETTER STEREO MATCHING FROM SIMPLE YET EFFECTIVE WRANGLING OF DEEP FEATURES

Lei Chen[1], *Zongqing Lu*[1,*], *Qingmin Liao*[1], *Jing-Hao Xue*[2]

[1]ShenZhen International Graduate School/Department of Electronic Engineering,
Tsinghua University, China
[2] Department of Statistical Science, University College London, UK
chen-l18@mails.tsinghua.edu.cn, luzq@sz.tsinghua.edu.cn,
liaoqm@tsinghua.edu.cn, jinghao.xue@ucl.ac.uk

ABSTRACT

Cost volume plays a pivotal role in stereo matching. Most recent works focused on deep feature extraction and cost refinement for a more accurate cost volume. Unlike them, we probe from a different perspective: feature wrangling. We find that simple wrangling of deep features can effectively improve the construction of cost volume and thus the performance of stereo matching. Specifically, we develop two simple yet effective wrangling techniques of deep features, spatially a differentiable feature transformation and channel-wise a memory-economical feature expansion, for better cost construction. Exploiting the local ordering information provided by a differentiable rank transform, we achieve an enhancement of the search for correspondence; with the help of disparity division, our feature expansion allows for more features into the cost volume with no extra memory required. Equipped with these two feature wrangling techniques, our simple network can perform outstandingly on the widely used KITTI and Sceneflow datasets.

Index Terms— Stereo matching, deep learning, feature wrangling, cost construction

1. INTRODUCTION

Stereo matching is a popular task in computer vision. It aims to find the horizontal offset, namely disparity d, between corresponding points in rectified stereo pairs. With the disparity d, baseline distance B and focal length f, the depth Z can be derived via $Z = fB/d$. This ability, obtaining 3D depth data from 2D images, enables stereo matching to enjoy wide applications, such as 3D reconstruction, autonomous driving and robotics navigation.

Traditionally, stereo matching can be accomplished through four steps [1]: matching cost computation, cost aggregation, disparity computation and disparity refinement. The first two steps are most crucial, constructing and then refining a pivotal object - cost volume. Cost volume describes the similarity between each pixel in the left view and its right-view corresponding points determined by a range of disparities. Many studies have been devoted to generating an optimal cost volume and achieved remarkable results [2–5].

With the arising of deep learning, powerful stereo feature descriptors can be leveraged to compute the matching cost [4, 6]. Given stereo feature pairs $\mathbf{f}_l, \mathbf{f}_r \in \mathbb{R}^{c\times h\times w}$, with w, h and c are the width, height and number of channels of feature maps, the cost construction can be divided into two types: correlation volume and 4D feature volume. The correlation operation proposed in [7] generated the cost volume by conducting the inner product of $\mathbf{f}_l$ and the shifted $\mathbf{f}_r$. Unlike it, others [5, 8, 9] directly concatenated $\mathbf{f}_l$ and the shifted $\mathbf{f}_r$ along the feature dimension and constructed a 4D feature volume (short as feature volume hereafter). GwcNet [9] and AMNet [10] explored the combination of both types; [11, 12] managed to reduce the feasible disparity range required for the cost construction.

Different from the above schemes and following the "garbage in, garbage out" principle, our focus is on a different perspective: how about making some feasible adjustments on the feature application, i.e. feature wrangling, to boost the stereo matching performance. Specifically, we develop two simple yet effective feature wrangling techniques: spatially a differentiable feature transformation and channel-wise a memory-economical feature expansion, as shown in Figure 1. Feature transformation extracts the local ordering information of pixels via a rank transform [2] and excludes unreasonable candidate corresponding pixels, yielding a better matching result. Feature expansion enables more feature descriptors to contribute to the cost construction with no extra memory required, facilitating an effective and economical construction.

We summarize our contributions as: 1) We make a new differentiable rank transform of deep features, to fully leverage spatially local ordering information for better stereo

*Corresponding Author. This work was supported by the Special Foundation for the Development of Strategic Emerging Industries of Shenzhen (JCYJ20170817161056260).

978-1-6654-3864-3/21/$31.00 ©2021 IEEE

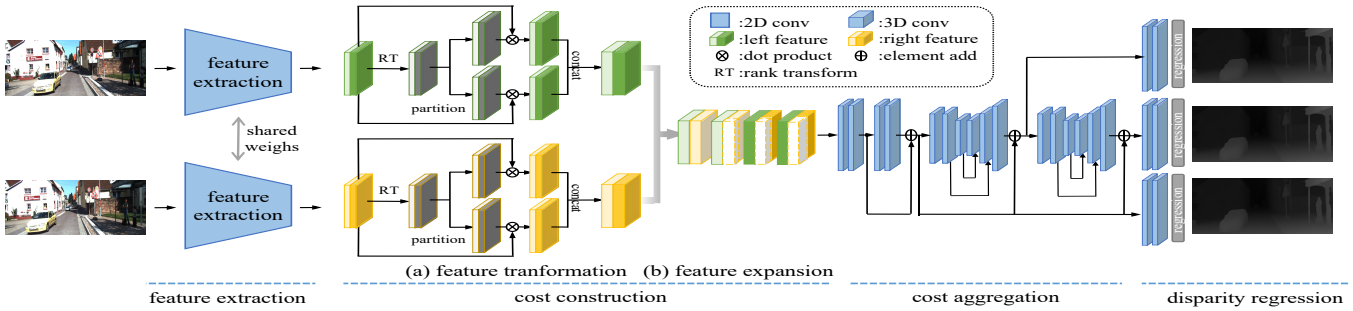


Fig. 1. Diagram of our proposed network. Our network architecture consists of four modules: feature extraction, cost construction, cost aggregation and disparity regression. In the cost construction, we propose two new feature wrangling ideas: (a) spatially a differentiable feature transformation; (b) channel-wise a memory-economical feature expansion. Moreover, in the cost aggregation, we also revamp the connections between 3D convolutions.

matching; 2) We propose a new channel-wise feature expansion idea, to encode more feature maps into a more informative cost volume without requiring any extra memory; 3) Equipped with the above two feature wrangling techniques, our simple network can achieve the state-of-the-art performance on the KITTI and Sceneflow datasets.

2. METHOD

2.1. Feature Transformation

Although deep neural networks often extract powerful features, [13] indicates that performing appropriate spatial transformation on features can help the network learn a more effective data pattern. Inspired by this, we propose a differentiable rank transform to wrangle stereo feature pairs into a more effective data pattern for stereo matching, as detailed below.

Unlike other visual tasks, stereo matching relies not only on the learning of the scene, but also on the stereo consistency between the two views. Therefore, a feature transform facilitating the stereo correspondence search is desired, for example, by alleviating inconsistent candidate pairs. Following this intuition, we partition the pixels into multiple subsets in each view, and conduct the correspondence search only in corresponding pairs of subsets, to effectively reduce the matching errors. That is, to boost the matching performance, we spatially decompose the matching task into multiple smaller, simpler and more accurate sub-tasks.

As for the partition strategy, in order to satisfy the local consistency between the transforms on the left and right views, we resort to a classical local feature transform - rank transform. Rank transform can extract the local ordering information from the data for stereo matching and achieve more robust results.

For a pixel $I(i,j)$, we take an $r \times r$ window centered at it as its local region. Then the rank transform of $I(i,j)$ can be

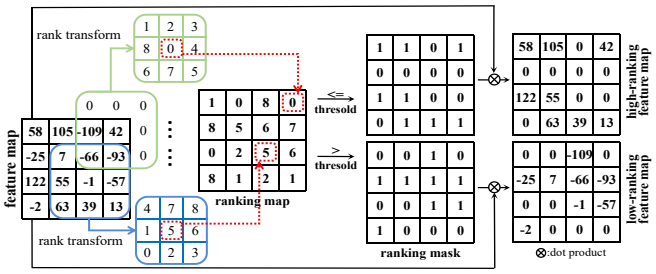


Fig. 2. The process of feature transformation. We illustrate the feature transformation with the window size 3×3 and threshold $t = 4$ for example.

written as

$$RT(i,j) = \sum_{(\Delta i, \Delta j)} H(I(i+\Delta i, j+\Delta j) - I(i,j)), \quad (1)$$

where $I(i+\Delta i, j+\Delta j)$ is a pixel within the $r \times r$ neighborhood of $I(i,j)$ and $H(\cdot)$ is the 0-1 Heaviside function. In our implementation, we take a differentiable approximation of H, denoted by H_ε, as follows:

$$H_\varepsilon(x) = \frac{1}{1+e^{-\varepsilon x}}. \quad (2)$$

With ε in (2) large enough, the result of (1) is close to the actual local ranking of $I(i,j)$. In this way, the rank transform can be modeled into our end-to-end network.

For a feature map F, rank transform, as shown in the green/blue box of Figure 2, transfers the pixel intensity into their local ranking. Comparing the ranking map with a given threshold t produces two mask maps, which can be then used to divide the pixels of original feature map into two feature maps, namely high-ranking and low-ranking feature maps, as illustrated in Figures 2 and 3. For example, our expressions to derive the high-ranking mask and map are

$$M_h(i,j) = \max\left(-\frac{RT(i,j)-t}{|RT(i,j)-t|}, 0\right), \quad (3)$$

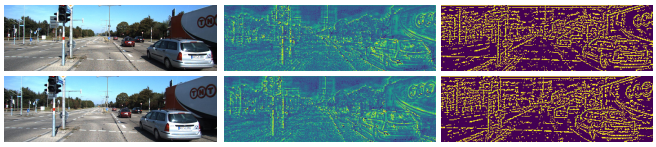


Fig. 3. From left to right: stereo image pairs (upper: left view; lower: right view), high-ranking and low-ranking feature pairs. For better view, we scaled the high-ranking and low-ranking feature maps differently.

$$F_h(i,j) = F(i,j) \cdot M_h(i,j). \tag{4}$$

High- and low-ranking feature maps can be regarded as attention maps focusing on specific pixel subsets. And the mask maps is regulated by the gradient back-propagation to achieve better pixel partition. This "attention" mechanism is locally adaptive and simpler to implement.

The threshold t can regulate the proportion of valid pixels in the high- and low-ranking feature maps. We found that, for the feature transform using the $r \times r$ window, the proportion in high- and low-ranking feature map are close to $t : r \times r$ and $r \times r - t : r \times r$, respectively. In Figure 3, we visualize the high- and low-ranking feature pairs, where the window size of rank transform is 5×5 [1] and $t = 18$.

2.2. Feature Expansion

In deep stereo matching, better performance can be expected by incorporating more features, but at the price of dramatically increasing cache requirement. To tackle this problem, we propose a feature expansion scheme that can effective plant more feature maps into the cost volume without requiring extra memory.

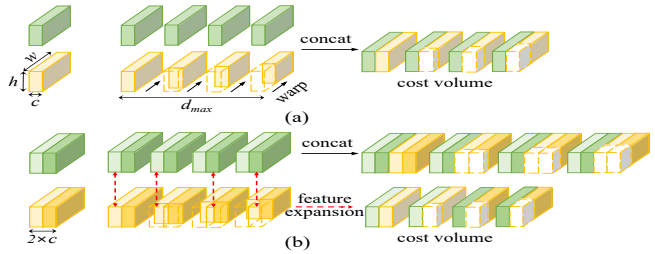


Fig. 4. (a) Conventional feature volume construction by concatenating cubes of c features from the left (green) and shifted right (yellow) views. (b) (lower) Our feature expansion vs. (upper) concatenation of cubes of kc features (divided into k subsets of c features) for feature volume construction. $k = 2$ and $d_{max} = 4$ are used in illustration.

As shown in Figure 4(a), given two c-channel feature pairs $\mathbf{f}_l, \mathbf{f}_r \in \mathbb{R}^{c \times h \times w}$, we can construct the feature volume (of size $c \times d_{max} \times h \times w$ with d_{max} the disparity range) in the conventional way [5]. However, when c increases to kc, it will require extra memory of $(kc - c) \times d_{max} \times h \times w$ (see the upper branch of Figure 4(b)), plus higher computation in subsequent networks as well. To circumvent this issue, as illustrated in the lower branch of Figure 4(b), we propose a simple feature expansion scheme. Specifically, for the stereo feature pairs $\mathbf{f}_l'$, $\mathbf{f}_r'$ with kc feature channels, we divide them into k feature subsets $(\{\mathbf{f}_{l,0}', \ldots, \mathbf{f}_{l,k-1}'\}, \{\mathbf{f}_{r,0}', \ldots, \mathbf{f}_{r,k-1}'\})$ with each of c feature channels. Accordingly, the disparity range $[0, d_{max} - 1]$ is also divided into k intervals $[0, \frac{d_{max}-1}{k}], \ldots, [\frac{(d_{max}-1)\times(k-1)}{k}, d_{max}-1]$. Then, each feature subset pair is responsible to construct the sub-volume in its corresponding disparity interval, and during training, gradually change from the equal initial states to the one most suitable for its corresponding disparity intervals. Finally, these k sub-volumes are concatenated to form the final feature volume.

The advantages of our proposed scheme are: first, it plants more feature maps into the final feature volume without requiring extra memory (Figure 4(a) vs. 4(b)). Secondly, different feature maps only need to be responsible for the matching under different disparity intervals, which simplifies the matching. Thirdly, it is very simple to implement, with only k the expansion factor determining how many more features to be planted.

2.3. Network Architecture

Our network architecture is illustrated in Figure 1, where the modules of feature extraction, cost construction and cost aggregation revamp the corresponding modules of PSMNet [8]. We remove spatial pyramid pooling from the feature extraction module, and add our proposed feature transformation and feature expansion into the cost construction module.

Moreover, in PSMNet, the cost aggregation module includes a basic 3D convolution combination and three 3D hourglasses [8]. We revamp the residual connection between 3D hourglasses and the identity connection within each 3D hourglass, as shown in Figure 1. Note that our cost aggregation just has two 3D hourglasses. The outputs of the basic 3D convolution combination and the two 3D hourglasses are used to supervise the training of the network.

In our network, we adopt the disparity regression [5] to estimate the disparity map. With c_d for $d = 0, \ldots, d_{max} - 1$ as the cost distribution, the disparity regression first performs the softmax operation on c_d to attain the probability distribution p_d, and then uses p_d to estimate the disparity $\hat{d}$:

$$p_d = \frac{e^{c_d}}{\sum_{i=0}^{d_{max}-1} e^{c_i}}; \quad \hat{d} = \sum_{d=0}^{d_{max}-1} p_d \times d\,. \tag{5}$$

We apply the smooth L_1 loss as the regression loss to train the network. In order to balance the losses of the three outputs, we adopt the weighted average strategy, and the weights are set to [0.5,0.7,1.0].

[1] In no special case, we use $r = 5$ for feature transformation.

Table 1. Ablation study of feature transformation on the Sceneflow test set and the KITTI2015 validation set. EPE is the main evaluation metric of Sceneflow, and $D1$ for KITTI, as marked in bold. The best strategies are marked in red and the most optimized schemes in each threshold are in blue.

model	Sceneflow		KITTI2015	
	EPE	$D1$	EPE	**D1**
original	1.0751	5.5209	0.6733	2.0089
ft_6	1.0007	5.1540	0.6554	1.7873
ft_6^h	1.0625	5.5100	0.6655	1.9521
ft_6^l	1.0125	5.2988	0.6595	1.7979
ft_{12}	1.0152	5.2215	0.6697	1.8508
ft_{12}^h	1.0319	5.3304	0.6742	1.8802
ft_{12}^l	1.0193	5.2862	0.6524	1.9030
ft_{18}	0.9818	5.1075	0.6623	1.7633
ft_{18}^h	1.0035	5.2588	0.6570	1.7788
ft_{18}^l	1.0449	5.4641	0.6693	1.9052
ft_{18}&fe_3	0.9388	5.0672	0.6485	1.7828

3. EXPERIMENTS

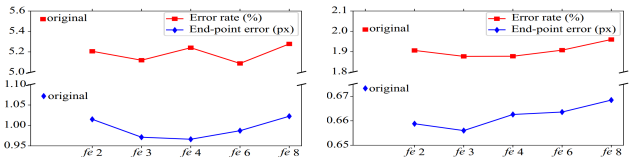
3.1. Implementation Details

Our network is implemented with PyTorch and trained with Adam (β_1 =0.9,β_2 =0.999). The disparity range is set to $[0, D-1]$, where $D = 192$.

DataSets. Sceneflow [7] is a large synthetic dataset with three subsets, namely Flyingthings3D, Monkaa and Driving. We perform experiments on Flyingthings3D, which contains 22,290 training image pairs and 4,370 test image pairs with dense ground-truth disparity maps. KITTI2012 [14] and KITTI2015 [15] are real scene datasets. KITTI2015 has 200 training image pairs with sparse ground truth and KITTI2012 has 194. It does not provide the ground truth for 200 test pairs of KITTI2015 and 195 of KITTI2012. The results of test set can be assessed by submitting to the evaluation server. Middlebury [16] is a high-resolution dataset of real indoor scenes, but it only provides no more than 50 image pairs for training.

Training strategy. In the ablation study, we use Sceneflow and KITTI2015 to evaluate our proposals and search for the best settings. On Sceneflow, the network is trained from scratch for 10 epochs at the learning rate of 10^{-3}. The best model is selected to be further finetuned on KITTI2015. The learning rate is set to 10^{-3} for first 200 epochs and reduced to 10^{-4} for the remaining 100 epochs. We also validate the applicability of our feature wrangling strategies on top networks (PSMNet, GA-Net). All the experiments are performed on two 1080Ti GPU with three batches on each GPU.

Performance metrics. End-point error (EPE) is the average value of disparity errors (the absolute error between the estimated disparity and the true value); N-px represents the percentage of pixels with disparity error beyond N; similar to N-px, $D1$ is the percentage of pixels with the disparity error beyond 3 or exceeding 5% of the true disparity.



(a) (b)

Fig. 5. Results of models with different feature expansion factor k on (a) Sceneflow and (b) KITTI, with EPE (end-point error) and $D1$ (error rate) as the evaluation metrics.

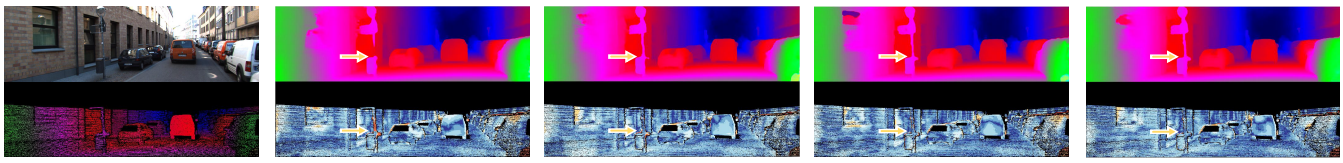
3.2. Ablation Studies

In this section, we conduct multiple experiments on Sceneflow and KITTI2015 to evaluate the effectiveness of our two proposals. For simplicity, we call the model employing feature transformation ft_t (t : threshold), and the model applying feature expansion fe_k (k : expansion factor).

Feature transformation. Different thresholds t represent different strategies to leverage the rank information in the feature transformation. We mainly compare the cases of $t = 6, 12$ and 18. For each threshold, we further compare the feature transformation scheme applying only high-ranking (ft_t^h) or low-ranking features (ft_t^l) with that applying both. In Table 1, we can observe the following patterns. First, all the ft_t outperform the original model without feature transformation, even for the models (ft_6^h, ft_{18}^l) having fewer valid pixels and lacking sufficient semantic information. This verifies the benefit from our feature transformation to the search for stereo correspondence. Secondly, the performance of the models with much valid pixels, such as ft_6^l and ft_{18}^h, is significantly improved, inferring that the pixel partition enhances the local variant of features and benefits to matching. Thirdly, models using high-ranking or low-ranking feature alone perform worse than models that applying both. ft_{18} works the best and its $D1$ result on KITTI2015 is 15% lower than the original model, owing to the necessary semantic information and the advantage of correspondence search.

Feature expansion. From Figure 5, we can observe the follows. First, all models with feature expansion have a performance improvement from the original model. On Sceneflow, the EPE of fe_4 is 0.9662, reducing nearly 10% from the original model (1.0751), indicating the benefit from using a more informative cost volume due to feature expansion. Secondly, there can be an optimal expansion factor k (e.g. $k = 3, 4$ for these two datasets), as with the increase of expansion factor k, the performance of fe_k does not monotonically increase. A very large k may challenge the feature extraction module to converge to an optimal solution.

In addition, we compare the two feature volume construction strategies in Figure 4(b). The EPE of the network with-



(a) (b) (c) (d) (e)

Fig. 6. (a) Left view (upper panel) and ground truth (lower panel). (b-e) The estimation results (upper panels) and error maps (lower panels) of the original model, ft_{18}, fe_3 and $ft_{18}\&fe_3$, respectively. The yellow arrows in (b-e) point to the estimation differences of these models in a specific region, and the warmer color in error maps means the larger error.

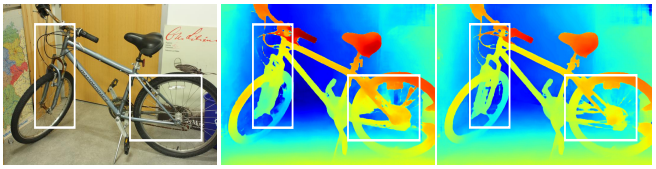


Fig. 7. Results on Middleburry test set. From left to right: left view, the result of our network without feature wrangling, and the result with feature wrangling, respectively.

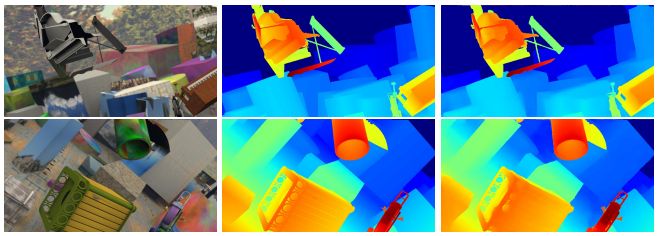


Fig. 8. Results of our model on the Sceneflow test set. From left to right: left view, ground truth, and our estimation.

out feature expansion on Sceneflow is 0.98, which achieves a subtle improvement comparing with the 1.01 of fe_2. However, taking account of parameter and prediction speed (7.2M vs. 4.6M and 0.62s vs. 0.42s), the feature expansion strategy shows significant performance.

Effectiveness of two feature wranglings. In Table 1, we also list the results of the model $ft_{18}\&fe_3$, combining the two schemes under their respective optimal hyper-parameters. In Table 1, $ft_{18}\&fe_3$ offers the best performance on Sceneflow, where EPE is reduced by 12.68% compared with the original model, and $D1$ is reduced by 8.22%. We also observe that, although the 3px (1.7828) of $ft_{18}\&fe_3$ on KITTI2015 is slightly higher than the 1.7633 of ft_{18}, $ft_{18}\&fe_3$ outperforms ft_{18} in all other scenarios, especially for the EPE of Sceneflow, which is improved by 4.38% from ft_{18}. The superiority of two strategies and their combination is reflected in the regions highlighted in Figure 6. Figure 7 shows us that the feature wrangling strategies promotes the network's estimation at some front and back scene boundaries (marked by white boxes).

Table 2. Results on the Sceneflow, KITTI2015 and KITTI2012 test sets. The key metric and best results are in bold. * denotes the model equipped our proposals.

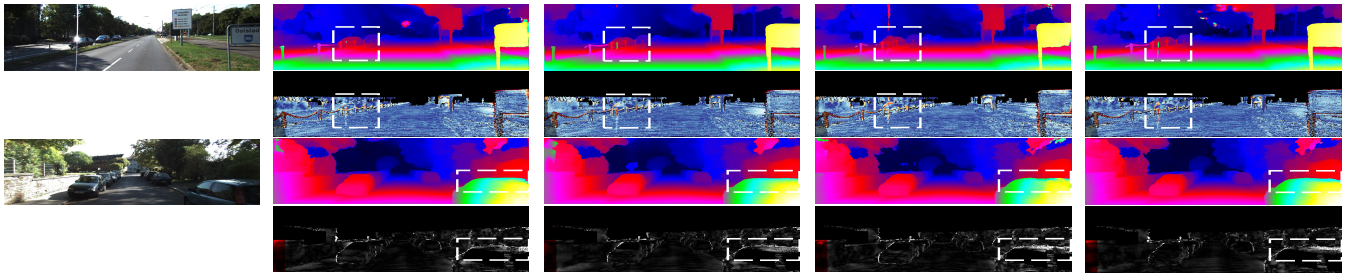
Method	Sceneflow EPE(px)	KITTI2015 D1(%)		KITTI2012 3px(%)	
		All	Noc	All	**Noc**
SegStereo [17]	1.45	2.25	2.08	2.03	1.68
EdgeStereo [18]	1.11	2.16	2.00	1.83	1.46
GC-Net [5]	2.51	2.87	2.45	2.30	1.77
SSPVC-Net [19]	0.87	2.11	1.91	1.90	1.47
DeepPruner [11]	0.86	3.56	2.15	-	-
PSMNet [8]	1.09	2.32	2.14	1.89	1.49
GA-Net [20]	0.84	1.81	1.63	**1.60**	1.19
PSMNet*	0.97	2.03	1.82	1.80	1.31
GA-Net*	**0.81**	**1.68**	**1.50**	1.69	**1.15**
ours	**0.81**	1.97	1.80	1.69	1.29

3.3. Comparisons with the state-of-the-art methods

In this section, we compare our best model with some state-of-the-art algorithms.

In Table 2, we show the EPEs of ours and some classic networks on the Sceneflow test set. Among them, our model and GA-Net* achieves the best performance, proving the validity of our proposal. Tables 2 also presents official evaluation data of top networks on the KITTI2015 and KITTI2012 test sets, respectively. On KITTI2015, our network performs outstandingly on multiple metrics and the performance promotion of GA-Net* and PSMNet* from their original network strongly confirms the effectiveness and applicability of our schemes. In fact, our network works better than PSMNet on both KITTI2012 and KITTI2015 test sets, showing the effectiveness of our network structure improvement.

Figure 8 reveals the excellent performance of our network on some detailed structures. The qualitative results on the KITTI2015 and KITTI2012 test sets are presented in Figure 9, which can be compared to highlight the advantages of our approach in some error-prone regions (marked in the white dashed boxes).



(a) Left (b) ours (c) GA-Net* (d) PSMNet (e) GA-Net

Fig. 9. Results of KITTI2015 (upper two rows) and KITTI2012 (lower two rows) test sets: (upper) disparity map and (lower) error map. For error maps, warmer or brighter color means larger error in KITTI2015 and KITTI2012, respectively. Significant improvements are highlighted by white boxes.

4. CONCLUSION

In this paper, we propose two feature wrangling strategies for the cost construction: One is a spatially differentiable feature transformation scheme amenable to stereo matching; and the other is a channel-wise memory-economical feature expansion scheme generating a information-richer cost volume. Both of them can effectively improve the performance of the network and help our simple network achieve the state-of-the-art results on multiple datasets. In future, we will further attempt other effective feature transformation to boost stereo matching performance.

5. REFERENCES

[1] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *IJCV*, vol. 47, no. 1-3, pp. 7–42, 2002.

[2] R. Zabih and J. Woodfill, "Non-parametric local transforms for computing visual correspondence," *ECCV*, pp. 151–158, 1994.

[3] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *PAMI*, vol. 30, no. 2, pp. 328–341, 2007.

[4] J. Zbontar and Y. LeCun, "Computing the stereo matching cost with a convolutional neural network," *CVPR*, pp. 1592–1599, 2015.

[5] 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," 2017, pp. 66–75.

[6] Z. Chen, X. Sun, L. Wang, Y. Yu, and C. Huang, "A deep visual correspondence embedding model for stereo matching costs," *ICCV*, pp. 972–980, 2015.

[7] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," *CVPR*, pp. 4040–4048, 2016.

[8] J. Chang and Y. Chen, "Pyramid stereo matching network," *ICCV*, pp. 5410–5418, 2018.

[9] X. Guo, K. Yang, W. Yang, X. Wang, and H. Li, "Group-wise correlation stereo network," *CVPR*, pp. 3273–3282, 2019.

[10] X. Du, M. El-Khamy, and J. Lee, "AMNet: Deep atrous multiscale stereo disparity estimation networks," *arXiv preprint arXiv:1904.09099*, 2019.

[11] S. Duggal, S. Wang, W. Ma, R. Hu, and R. Urtasun, "DeepPruner: learning efficient stereo matching via differentiable patchmatch," *ICCV*, pp. 4384–4393, 2019.

[12] X. Gu, Z. Fan, S. Zhu, Z. Dai, F. Tan, and P. Tan, "Cascade cost volume for high-resolution multi-view stereo and stereo matching.," *CVPR*, 2020.

[13] M. Jaderberg, K. Simonyan, A. Zisserman, and Kavukcuoglu K., "Spatial transformer networks," pp. 2017–2025, 2015.

[14] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," *CVPR*, pp. 3354–3361, 2012.

[15] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," *CVPR*, pp. 3061–3070, 2015.

[16] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, and P. Westling, "High-resolution stereo datasets with subpixel-accurate ground truth," *GCPR*, pp. 31–42, 2014.

[17] G. Yang, H. Zhao, J. Shi, Z. Deng, and J. Jia, "SegStereo: Exploiting semantic information for disparity estimation," *ECCV*, pp. 636–651, 2018.

[18] X. Song, X. Zhao, H. Hu, and L. Fang, "EdgeStereo: A context integrated residual pyramid network for stereo matching," *ACCV*, pp. 20–35, 2018.

[19] Z. Wu, X. Wu, X. Zhang, S. Wang, and L. Ju, "Semantic stereo matching with pyramid cost volumes," *CVPR*, pp. 7484–7493, 2019.

[20] F. Zhang, V. Prisacariu, R. Yang, and P. Torr, "GA-Net: Guided aggregation net for end-to-end stereo matching," *CVPR*, pp. 185–194, 2019.