GA-Net: Guided Aggregation Net for End-to-end Stereo Matching

Feihu Zhang[1]* Victor Prisacariu[1] Ruigang Yang[2] Philip H.S. Torr[1]
[1] University of Oxford [2] Baidu Research, Baidu Inc.

立体匹配 匹配成本融合非常重要 提出两个层 可以捕捉全局和局部依赖性 前者是半全局融合层是可微分的 后者是局部导向融合层 后边跟着传统的成本滤波策略

Abstract

In the stereo matching task, matching cost aggregation is crucial in both traditional methods and deep neural network models in order to accurately estimate disparities. We propose two novel neural net layers, aimed at capturing local and the whole-image cost dependencies respectively. The first is a semi-global aggregation layer which is a differentiable approximation of the semi-global matching, the second is the local guided aggregation layer which follows a traditional cost filtering strategy to refine thin structures.

These two layers can be used to replace the widely used 3D convolutional layer which is computationally costly and memory-consuming as it has cubic computational/memory complexity. In the experiments, we show that nets with a two-layer guided aggregation block easily outperform the state-of-the-art GC-Net which has nineteen 3D convolutional layers. We also train a deep guided aggregation network (GA-Net) which gets better accuracies than state-of-the-art methods on both Scene Flow dataset and KITTI benchmarks. Code will be available at https://github.com/feihuzhang/GANet.

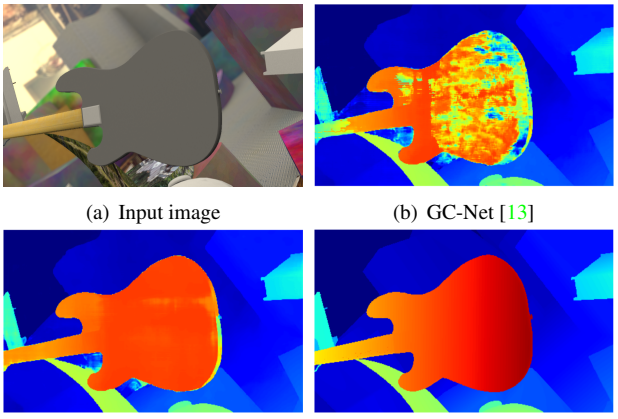
1. Introduction

Stereo reconstruction is a major research topic in computer vision, robotics and autonomous driving. It aims to estimate 3D geometry by computing disparities between matching pixels in a stereo image pair. It is challenging due to a variety of real-world problems, such as occlusions, large textureless areas (*e.g.* sky, walls *etc.*), reflective surfaces (*e.g.* windows), thin structures and repetitive textures.

Traditionally, stereo reconstruction is decomposed into three important steps: feature extraction (for matching cost computation), matching cost aggregation and disparity prediction [9, 21]. Feature-based matching is often ambiguous, with wrong matches having a lower cost than the correct ones, due to occlusions, smoothness, reflections, noise etc. Therefore, cost aggregation is a key step needed to obtain accurate disparity estimations in challenging regions.

Deep neural networks have been used for matching cost computation in, e.g, [30, 33], with (i) cost aggregation based on traditional approaches, such as cost filtering [10] and semi-global matching (SGM) [9] and (ii) disparity computation with a separate step. Such methods considerably improve over traditional pixel matching, but still struggle to produce accurate disparity results in textureless, reflective and occluded regions. End-to-end approaches that link matching with disparity estimation were developed in e.g. DispNet [15], but it was not until GC-Net [13] that cost aggregation, through the use of 3D convolutions, was incorporated in the training pipeline. The more recent work of [3], PSMNet, further improves accuracy by implementing the stacked hourglass backbone [17] and considerably increasing the number of 3D convolutional layers for cost aggregation. The large memory and computation cost incurred by using 3D convolutions is reduced by down-sampling and up-sampling frequently, but this leads to a loss of precision in the disparity map.

Among these approaches, traditional semi-global matching (SGM) [9] and cost filtering [10] are all robust and efficient cost aggregation methods which have been widely



(a) Input image (b) GC-Net [13] 当前最好方法

(c) Our GA-Net-2 (d) Ground truth 我们的结果和gt

Figure 1: Performance illustrations. (a) a challenging input image. (b) Result of the state-of-the-art method GC-Net [13] which has nineteen 3D convolutional layers for matching cost aggregation. (c) Result of our GA-Net-2, which only uses two proposed GA layers and two 3D convolutional layers. It aggregates the matching information into the large textureless region and is an order of magnitude faster than GC-Net. (d) Ground truth.

*Part of the work was done when working in Baidu Research.

used in many industrial products. But, they are not differentiable and cannot be easily trained in an end-to-end manner.

In this work, we propose two novel cost aggregation layers for end-to-end stereo reconstruction to replace the use of 3D convolutions. Our solution considerably increases accuracy, while decreasing both memory and computation costs.

First, we introduce a semi-global guided aggregation layer (SGA) which implements a differentiable approximation of semi-global matching (SGM) [9] and aggregates the matching cost in different directions over the whole image. This enables accurate estimations in occluded regions or large textureless/reflective regions.

Second, we introduce a local guided aggregation layer (LGA) to cope with thin structures and object edges in order to recover the loss of details caused by down-sampling and up-sampling layers.

As illustrated in Fig. 1, a cost aggregation block with only two GA layers and two 3D convolutional layers easily outperforms the state-of-the-art GC-Net [13], which has nineteen 3D convolutional layers. More importantly, one GA layer has only 1/100 computational complexity in terms of FLOPs (floating-point operations) as that of a 3D convolution. This allows us to build a real-time GA-Net model, which achieves better accuracy compared with other existing real-time algorithms and runs at a speed of 15$\sim$20 fps.

We further increase the accuracy by improving the network architectures used for feature extraction and matching cost aggregation. The full model, which we call "GA-Net", achieves the state-of-the-art accuracy on both the Scene Flow dataset [15] and the KITTI benchmarks [7, 16].

2. Related Work

Feature based matching cost is often ambiguous, as wrong matches can easily have a lower cost than correct ones, due to occlusions, smoothness, reflections, noise etc. To deal with this, many cost aggregation approaches have been developed to refine the cost volume and achieve better estimations. This section briefly introduces related work in the application of deep neural networks in stereo reconstruction with a focus on the existing matching cost aggregation strategies, and briefly reviews approaches for traditional local and semi-global cost aggregations.

2.1. Deep Neural Networks for Stereo Matching

Deep neural networks were used to compute patch-wise similarity scores in [4, 6, 29, 33], with traditional cost aggregation and disparity computation/refinement methods [9, 10] used to get the final disparity maps. These approaches achieved state-of-the-art accuracy, but, limited by the traditional matching cost aggregation step, often produced wrong predictions in occluded regions, large textureless/reflective regions and around object edges. Some other methods looked to improve the performance of traditional cost aggregation, with, e.g. SGM-Nets [23] predicting the penalty-parameters for SGM [9] using a neural net, whereas Schönberger *et al*. [22] learned to fuse proposals by optimization in stereo matching and Yang *et al*. proposed to aggregate costs using a minimum spanning tree [28].

Recently, end-to-end deep neural network models have become popular. Mayer *et al*. created a large synthetic dataset to train end-to-end deep neural network for disparity estimation (*e.g*. DispNet) [15]. Pang *et al*. [19] built a two-stage convolutional neural network to first estimate and then refine the disparity maps. Tulyakov *et al*. proposed end-to-end deep stereo models for practical applications [26]. GC-Net [13] incorporated the feature extraction, matching cost aggregation and disparity estimation into a single end-to-end deep neural model to get state-of-the-art accuracy on several benchmarks. PSMNet [3] used pyramid feature extraction and a stacked hourglass block [18] with twenty-five 3D convolutional layers to further improve the accuracy.

2.2. Cost Aggregation

Traditional stereo matching algorithms [1,9,27] added an additional constraint to enforce smoothness by penalizing changes of neighboring disparities. This can be both local and (semi-)global, as described below.

2.2.1 Local Cost Aggregation

The cost volume C is formed of matching costs at each pixel's location for each candidate disparity value d. It has a size of $H \times W \times D_{max}$ (with H: image height, W: image width, D_{max}: maximum of the disparities) and can be sliced into D_{max} slices for each candidate disparity d. An efficient cost aggregation method is the local cost filter framework [10, 31], where each slice of the cost volume $C(d)$ is filtered independently by a guided image filter [8, 25, 31]. The filtering for pixel's location $\mathbf{p} = (x, y)$ at disparity d is a weighted average of all neighborhoods $\mathbf{q} \in N_{\mathbf{p}}$ in the same slice $C(d)$:

$$C^A(\mathbf{p}, d) = \sum_{\mathbf{q} \in N_{\mathbf{p}}} \omega(\mathbf{p}, \mathbf{q}) \cdot C(\mathbf{q}, d) \tag{1}$$

Where $C(\mathbf{q}, d)$ means the matching cost at location $\mathbf{p}$ for candidate disparity d. $C^A(\mathbf{p}, d)$ represents the aggregated matching cost. Different image filters [8,25,31] can be used to produce the guided filter weights ω. Since these methods only aggregate the cost in a local region $N_{\mathbf{p}}$, they can run at fast speeds and reach real-time performance.

2.2.2 Semi-Global Matching

When enforcing (semi-)global aggregation, the matching cost and the smoothness constraints are formulated into one energy function $E(D)$ [9] with the disparity map of the input image as D. The problem of stereo matching can now be formulated as finding the best disparity map D^* that min-

imizes the energy $E(D)$:

$$E(D) = \sum_{\mathbf{p}}\{C_{\mathbf{p}}(D_{\mathbf{p}}) + \sum_{\mathbf{q}\in N_{\mathbf{p}}} P_1 \cdot \delta(|D_{\mathbf{p}} - D_{\mathbf{q}}| = 1) + \sum_{\mathbf{q}\in N_{\mathbf{p}}} P_2 \cdot \delta(|D_{\mathbf{p}} - D_{\mathbf{q}}| > 1)\}. \quad (2)$$

The first term $\sum_{\mathbf{p}} C_{\mathbf{p}}(D_{\mathbf{p}})$ is the sum of matching costs at all pixel locations $\mathbf{p}$ for disparity map D. The second term is a constant penalty P_1 for locations $\mathbf{q}$ in the neighborhood of $\mathbf{p}$ if they have small disparity discontinuities in disparity map D ($|D_{\mathbf{p}} - D_{\mathbf{q}}| = 1$). The last term adds a larger constant penalty P_2, for all larger disparity changes ($|D_{\mathbf{p}} - D_{\mathbf{q}}| > 1$).

Hirschmuller proposed to aggregate matching costs in 1D from sixteen directions to get a approximate solution with $O(KN)$ time complexity, which is well known as semi-global matching (SGM) [9]. The cost $C^A_{\mathbf{r}}(\mathbf{p},d)$ of a location $\mathbf{p}$ at disparity d aggregates along a path over the whole image in the direction $\mathbf{r}$, and is defined recursively as:

$$C^A_{\mathbf{r}}(\mathbf{p},d) = C(\mathbf{p},d) + \min \begin{cases} C^A_{\mathbf{r}}(\mathbf{p}-\mathbf{r},d), \\ C^A_{\mathbf{r}}(\mathbf{p}-\mathbf{r},d-1) + P_1, \\ C^A_{\mathbf{r}}(\mathbf{p}-\mathbf{r},d+1) + P_1, \\ \min\limits_i C^A_{\mathbf{r}}(\mathbf{p}-\mathbf{r},i) + P_2. \end{cases} \quad (3)$$

Where $\mathbf{r}$ is a unit direction vector. The same aggregation steps were used in MC-CNN [23, 30], and similar iterative steps were employed in [1, 2, 14].

In the following section, we detail our much more efficient guided aggregation (GA) strategies, which include a semi-global aggregation (SGA) layer and a local guided aggregation (LGA) layer. Both GA layers can be implemented with back propagation in end-to-end models to replace the low-efficient 3D convolutions and obtain higher accuracy.

3. Guided Aggregation Net

In this section, we describe our proposed guided aggregation network (GA-Net), including the guided aggregation (GA) layers and the improved network architecture.

3.1. Guided Aggregation Layers

State-of-the-art end-to-end stereo matching neural nets such as [3, 13] build a 4D matching cost volume (with size of $H \times W \times D_{max} \times F$, H: height, W: width, D_{max}: max disparity, F: feature size) by concatenating features between the stereo views, computed at different disparity values. This is next refined by a cost aggregation stage, and finally used for disparity estimation. Different from these approaches, and inspired by semi-global and local matching cost aggregation methods [9, 10], we propose our semi-global guided aggregation (SGA) and local guided aggregation (LGA) layers, as outlined below.

3.1.1 Semi-Global Aggregation

Traditional SGM [9] aggregates the matching cost iteratively in different directions (Eq. (3)). There are several difficulties in using such a method in end-to-end trainable deep neural network models.

First, SGM has many user-defined parameters (P_1, P_2), which are not straightforward to tune. All of these parameters become unstable factors during neural network training. Second, the cost aggregations and penalties in SGM are fixed for all pixels, regions and images without adaptation to different conditions. Third, the hard-minimum selection leads to a lot of fronto parallel surfaces in depth estimations.

We design a new semi-global cost aggregation step which supports backpropagation. This is more effective than the traditional SGM and can be used repetitively in a deep neural network model to boost the cost aggregation effects. The proposed aggregation step is:

$$C^A_{\mathbf{r}}(\mathbf{p},d) = C(\mathbf{p},d) + \text{sum} \begin{cases} \mathbf{w}_1(\mathbf{p},\mathbf{r}) \cdot C^A_{\mathbf{r}}(\mathbf{p}-\mathbf{r},d), \\ \mathbf{w}_2(\mathbf{p},\mathbf{r}) \cdot C^A_{\mathbf{r}}(\mathbf{p}-\mathbf{r},d-1), \\ \mathbf{w}_3(\mathbf{p},\mathbf{r}) \cdot C^A_{\mathbf{r}}(\mathbf{p}-\mathbf{r},d+1), \\ \mathbf{w}_4(\mathbf{p},\mathbf{r}) \cdot \max\limits_i C^A_{\mathbf{r}}(\mathbf{p}-\mathbf{r},i). \end{cases} \quad (4)$$

This is different from the SGM in three ways. First, we make the user-defined parameters learnable and add them as penalty coefficients/weights of the matching cost terms. These weights would therefore be adaptive and more flexible at different locations for different situations. Second, we replace the first/external minimum selection in Eq. (3) with a weighted sum, without any loss in accuracy. This change was proven effective in [24], where convolutions with strides were used to replace the max-pooling layers to get an all convolutional network without loss of accuracy. Third, the internal/second minimum selection is changed to a maximum. This is because the learning target in our models is to maximize the probabilities at the ground truth depths instead of minimizing the matching costs. Since $\max\limits_i C^A_{\mathbf{r}}(\mathbf{p}-\mathbf{r},i)$ in Eq. (4) can be shared by $C^A_{\mathbf{r}}(\mathbf{p},d)$ for d different locations, here, we do not use another weighted summation to replace it in order to reduce the computational complexity.

For both Eq. (3) and Eq. (4), the values of $C^A_{\mathbf{r}}(\mathbf{p},d)$ increase along the path, which may lead to very large values. We normalize the weights of the terms to avoid such a problem. This leads to our new semi-global aggregation:

$$C^A_{\mathbf{r}}(\mathbf{p},d) = \text{sum} \begin{cases} \mathbf{w}_0(\mathbf{p},\mathbf{r}) \cdot C(\mathbf{p},d) \\ \mathbf{w}_1(\mathbf{p},\mathbf{r}) \cdot C^A_{\mathbf{r}}(\mathbf{p}-\mathbf{r},d), \\ \mathbf{w}_2(\mathbf{p},\mathbf{r}) \cdot C^A_{\mathbf{r}}(\mathbf{p}-\mathbf{r},d-1), \\ \mathbf{w}_3(\mathbf{p},\mathbf{r}) \cdot C^A_{\mathbf{r}}(\mathbf{p}-\mathbf{r},d+1). \\ \mathbf{w}_4(\mathbf{p},\mathbf{r}) \cdot \max\limits_i C^A_{\mathbf{r}}(\mathbf{p}-\mathbf{r},i). \end{cases} \quad (5)$$
$$s.t. \quad \sum_{i=0,1,2,3,4} \mathbf{w}_i(\mathbf{p},\mathbf{r}) = 1$$

$C(\mathbf{p},d)$ is known as the cost volume (with a size of $H \times W \times D_{max} \times F$). Same as the traditional SGM [9], the cost volume can be sliced into D_{max} slices at the third dimension for

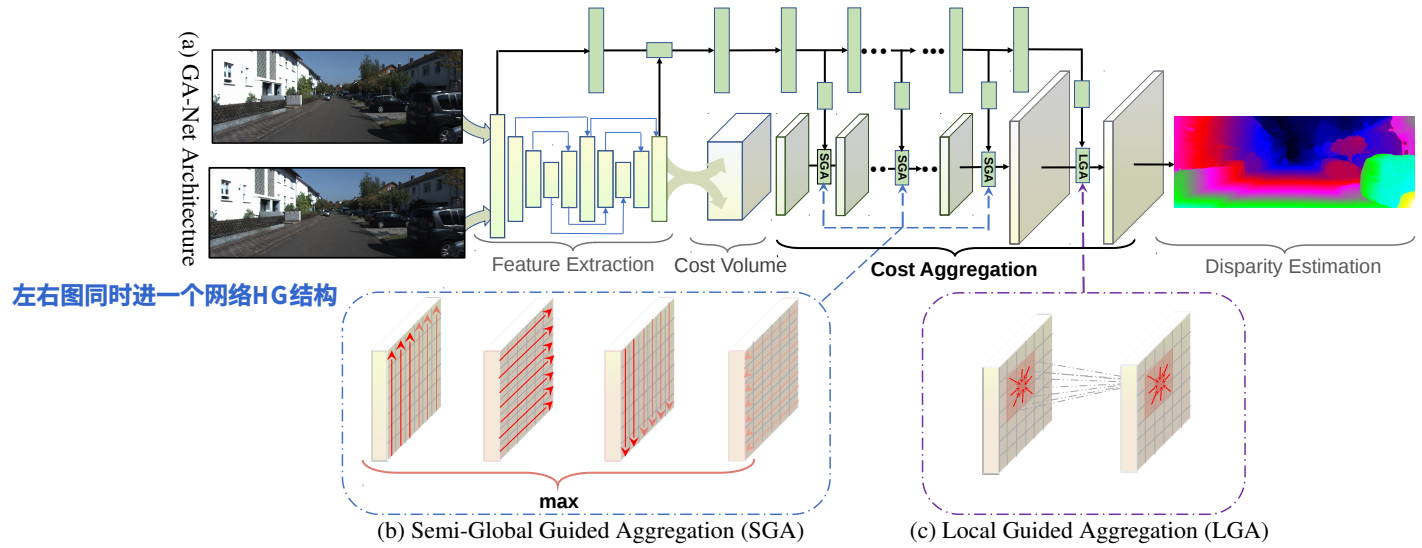


Figure 2: (a) Architecture overview. The left and right images are fed to a weight-sharing feature extraction pipeline. It consists of a stacked hourglass CNN and is connected by concatenations. The extracted left and right image features are then used to form a 4D cost volume, which is fed into a cost aggregation block for regularization, refinement and disparity regression. The guidance subnet (green) generates the weight matrices for the guided cost aggregations (SGA and LGA). (b) SGA layers semi-globally aggregate the cost volume in four directions. (c) The LGA layer is used before the disparity regression and locally refines the 4D cost volume for several times.

each candidate disparity d and each of these slices repeats the aggregation operation of Eq. (5) with the shared weight matrices ($\mathbf{w}_{0...4}$). All the weights $\mathbf{w}_{0...4}$ can be achieved by a guidance subnet (as shown in Fig. 2). Different to the original SGM which aggregates in sixteen directions, in order to improve the efficiency, the proposed aggregations are done in totally four directions (left, right, up and down) along each row or column over the whole image, namely $\mathbf{r} \in \{(0,1),(0,-1),(1,0),(-1,0)\}$.

The final aggregated output $C^A(\mathbf{p})$ is obtained by selecting the maximum between the four directions:

$$C^A(\mathbf{p},d) = \max_{\mathbf{r}} C^A_{\mathbf{r}}(\mathbf{p},d) \tag{6}$$

The last maximum selection keeps the best message from only one direction. This guarantees that the aggregation effects are not blurred by the other directions. The backpropagation for $\mathbf{w}$ and $C(\mathbf{p},d)$ in the SGA layer can be done inversely as Eq. (5) (details are available in the Appendix A.). Our SGA layer can be repeated several times in the neural network model to obtain better cost aggregation effects (as illustrated in Fig. 2).

3.1.2 Local Aggregation

We now introduce the local guided aggregation (LGA) layer which aims to refine the thin structures and object edges. Down-sampling and up-sampling are widely used in stereo matching models which blurs thin structures and object edges. The LGA layer learns several guided filters to refine the matching cost and aid in the recovery of thin structure information. The local aggregation follows the cost filter definition [10] (Eq. (1)) and can be written as:

$$C^A(\mathbf{p},d) = \text{sum}\begin{cases} \sum_{\mathbf{q}\in N_\mathbf{p}} \omega_0(\mathbf{p},\mathbf{q})\cdot C(\mathbf{q},d), \\ \sum_{\mathbf{q}\in N_\mathbf{p}} \omega_1(\mathbf{p},\mathbf{q})\cdot C(\mathbf{q},d-1), \\ \sum_{\mathbf{q}\in N_\mathbf{p}} \omega_2(\mathbf{p},\mathbf{q})\cdot C(\mathbf{q},d+1). \end{cases} \tag{7}$$
$$s.t. \sum_{\mathbf{q}\in N_\mathbf{p}} \omega_0(\mathbf{p},\mathbf{q}) + \omega_1(\mathbf{p},\mathbf{q}) + \omega_2(\mathbf{p},\mathbf{q}) = 1$$

Different slices (totally D_{max} slices) of cost volume share the same filtering/aggregation weights in LGA. This is the same as the original cost filter framework [10] and the SGA (Eq.(5)) in this paper. While, different with the traditional cost filter [10] which uses a $K\times K$ filter kernel to filter the cost volume in a $K\times K$ local/neighboor region $N_\mathbf{p}$, the proposed LGA layer has three $K\times K$ filters (ω_0, ω_1 and ω_2) at each pixel location $\mathbf{p}$ for disparities d, $d-1$ and $d+1$ respectively. Namely, it aggregates with a $K\times K\times 3$ weight matrix in a $K\times K$ local region for each pixel location $\mathbf{p}$. The setting of the weight matrix is also similar to [11], but, weights and filters are shared during the aggregation as designed in [10].

3.1.3 Efficient Implementation

We use several 2D convolutional layers to build a fast guidance subnet (as illustrated in Fig. 2). The implementation is similar to [32]. It uses the reference image as input and outputs the aggregation weights $\mathbf{w}$ (Eq. (5)). For a 4D cost volume C with size of $H\times W\times D\times F$ (H: height, W: width, D: max disparity, F: feature size), the output of the guidance subnet is split, reshaped and normalized as four $H\times W\times K\times F$ ($K=5$) weight matrices for four directions' aggregation using Eq. (5). Note that aggregations for dif-

ferent disparities corresponding to a slice d share the same aggregation weights. Similarly, the LGA layer need to learn a $H \times W \times 3K^2 \times F$ ($K = 5$) weight matrix and aggregates using Eq. (7).

Even though the SGA layer involves an iterative aggregation across the width or the height, the forward and backward can be computed in parallel due to the independence between elements in different feature channels or rows/columns. For example, when aggregating in the left direction, the elements in different channels or rows are independent and can be computed simultaneously. The elements of the LGA layer can also be computed in parallel by simply decomposing it into element-wise matrix multiplications and summations. In order to increase the receptive field of the LGA layer, we repeat the computation of EQ. (7) twice with the same weight matrix, which is similar to [5].

3.2. Network Architecture

As illustrated in Fig.2, the GA-Net consists of four parts: the feature extraction block, the cost aggregation for the 4D cost volume, the guidance subnet to produce the cost aggregation weights and the disparity regression. For the feature extraction, we use a stacked hourglass network which is densely connected by concatenations between different layers. The feature extraction block is shared by both left and right views. The extracted features for left and right images are then used to form a 4D cost volume. Several SGA layers are used for the cost aggregation and LGA layers can be implemented before and after the softmax layer of the disparity regression. It refines the thin-structures and compensate for the accuracy loss caused by the down-sampling done for the cost volume. The weight matrices (in Eq.(5) and Eq.(7)) are generated by an extra guidance subnet which uses the reference view (*e.g.* the left image) as input. The guidance subnet consists of several fast 2D convolutional layers and the outputs are reshaped and normalized into required weight matrices for these GA layers.[1]

3.3. Loss Function

We adopt the smooth L_1 loss function to train our models. Smooth L_1 is robust at disparity discontinuities and has low sensitivity to outliers or noises, as compared to L_2 loss. The loss function for training our models is defined as:

$$\begin{aligned} L(\hat{d}, d) &= \tfrac{1}{N} \sum_{n=1}^{N} l(|\hat{d} - d|) \\ l(x) &= \begin{cases} x - 0.5, & x \geq 1 \\ x^2/2, & x < 1 \end{cases} \end{aligned} \tag{8}$$

where, $|\hat{d} - d|$ measures the absolute error of the disparity predictions, N is the number of valid pixels with ground truths for training.

[1]The parameter settings of "GA-Net-15" used in our experiments are detailed in the Appendix B.

For the disparity estimation, we employ the disparity regression proposed in [13]:

$$\hat{d} = \sum_{d=0}^{D_{max}} d \times \sigma(-C^A(d)) \tag{9}$$

The disparity prediction $\hat{d}$ is the sum of each disparity candidate weighted by its probability. The probability of each disparity d is calculated after cost aggregation via the softmax operation $\sigma(\cdot)$. The disparity regression is shown more robust than classification based methods and can generate sub-pixel accuracy.

4. Experiments

In this section, we evaluate our GA-Nets with different settings using Scene Flow [15] and KITTI [7, 16] datasets. We implement our architectures using pytorch or caffe [12] (only for real-time models' implementation). All models are optimized with Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$). We train with a batch size of 16 on eight GPUs using 240×576 random crops from the input images. The maximum of the disparity is set as 192. Before training, we normalize each channel of the image by subtracting their means and dividing their standard deviations. We train the model on Scene Flow dataset for 10 epochs with a constant learning rate of 0.001. For the KITTI datasets, we fine-tune the models pre-trained on Scene Flow dataset for a further 640 epochs. The learning rate for fine-tuning begins at 0.001 for the first 300 epochs and decreases to 0.0001 for the remaining epochs.

4.1. Ablation Study

We evaluate the performance of GA-Nets with different settings, including different architectures and different number (0-4) of GA layers. As listed in Table 1, The guided aggregation models significantly outperform the baseline setting which only has 3D convolutional layers for cost aggregation. The new architectures for feature extraction and cost aggregation improve the accuracy by 0.14% on KITTI dataset and 0.9% on Scene Flow dataset. Finally, the best setting of GA-Net with three SGA layers and one LGA layer gets the best 3-pixel threshold error rate of 2.71% on KITTI 2015 validation set. It also achieves the best average EPE of 0.84 pixel and the best 1-pixel threshold error rate of 9.9% on the Scene Flow test set.

4.2. Effects of Guided Aggregations

In this section, we compare the guided aggregation strategies with other matching cost aggregation methods. We also analyze the effects of the GA layers by observing the post-softmax probabilities output by different models.

Firstly, our proposed GA-Nets are compared with the cost aggregation architectures in GC-Net (with nineteen 3D convolutions) and PSMNet (with twenty-five 3D convolutions). We fixed the feature extraction architecture as pro-

Table 1: Evaluations of GA-Nets with different settings. Average end point error (EPE) and threshold error rate are used for evaluations.

Feature Extraction	Cost Aggregation			Scene Flow		KITTI 2015
Stacked Block	Densely Concatenate	SGA Layer	LGA Layer	EPE Error	Error Rates (%)	Error Rates (%)
				1.26	13.4	3.39
√				1.19	13.0	3.31
√	√			1.14	12.5	3.25
√	√	+1		1.05	11.7	3.09
√	√	+2		0.97	11.0	2.96
√	√	+3		0.90	10.5	2.85
√	√	+4		0.89	10.4	2.83
√	√	+3	√	**0.84**	**9.9**	**2.71**

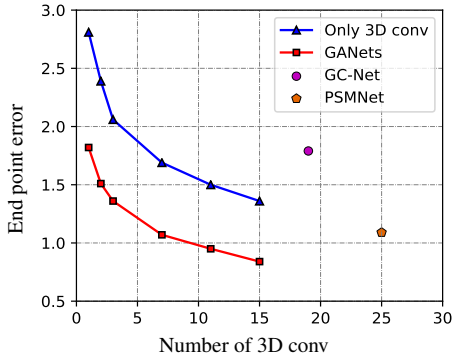


Figure 3: Illustration of the effects of guided aggregations. GA-Nets are compared with the same architectures without GA Layers. Evaluations are on Scene Flow dataset using average EPE.

Table 2: Comparisons of different cost aggregation methods. Average end point error (EPE) and 1-pixel threshold error rate are used for evaluations on Scene Flow dataset.

Models	3D Conv Number	Param	Time(s)	EPE Error	Error Rates
GC-Net	19	2.9M	4.4	1.80	15.6
PSMNet	25	3.5M	2.1	1.09	12.1
GA-Net-1	1	0.5M	0.17	1.82	16.5
GA-Net-2	2	0.7M	0.35	1.51	15.0
GA-Net-3	3	0.8M	0.42	1.36	13.9
GA-Net-7	7	1.3M	0.62	1.07	11.9
GA-Net-11	11	1.8M	0.95	0.95	10.8
GA-Net-15	15	2.3M	1.5	**0.84**	**9.9**

posed above. As shown in Table 2, GA-Nets have fewer parameters, run at a faster speed and achieve better accuracy. *E.g.*, with only two GA layers and two 3D convolutions, our GA-Net-2 outperforms the GC-Net by 0.29 pixel in average EPE. Also, the GA-Net-7 with three GA layers and seven 3D convolutions outperforms the current best PSMNet [3] which has twenty-five 3D convolutional layers.

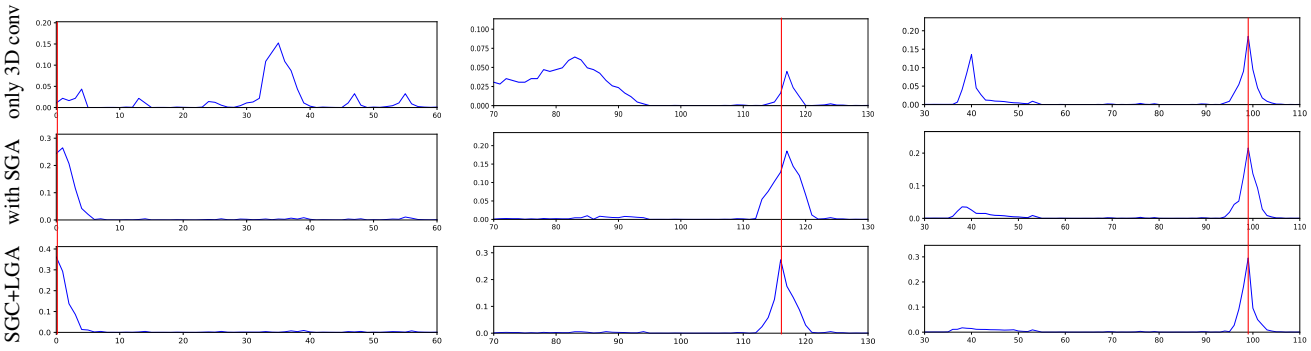
We also study the effects of the GA layers by comparing with the same architectures without GA steps. These baseline models "GA-Nets*" have the same network architectures and all other settings except that there is no GA layer implemented. As shown in Fig. 3, for all these models, GA layers have significantly improved the models' accuracy (by 0.5-1.0 pixels in average EPE). For example, the GA-Net-2 with two 3D convolutions and two GA layers produces lower EPE (1.51) compared with GA-Net*-11 (1.54) which utilizes eleven 3D convolutions. This implies that two GA layers are more effective than nine 3D convolutional layers.

Finally, in order to observe and analyze the effects of GA layers, in Fig. 4, we plot the post-softmax probabilities with respect to a range of candidate disparities. These probabilities are directly used for disparity estimation using Eq. (9) and can reflect the effectiveness of the cost aggregation strategies. The data samples are all selected from some challenging regions, such as a large textureless region (sky), the reflective region (window of a car) and pixels around the object edges. Three different models are compared. The first model (first row of Fig. 4) only has 3D convolutions (without any GA layers), the second model (second row of Fig. 4) has SGA layers and the last model (last row of Fig. 4) has both SGA layers and LGA layer.

As illustrated in Fig. 4(a), for large textureless regions, there would be a lot of noise since there is no any distinctive features in these regions for correct matching. The SGA layers successfully suppress these noise in the probabilities by aggregating surrounding matching information. The LGA layer further concentrates the probability peak on the ground truth value. It could refine the matching results. Similarly, in the sample of reflective region (Fig. 4(b)), the SGA and LGA layers correct the wrong matches and concentrate the peak on the correct disparity value. For the samples around the objects edges (Fig. 4(c)), there are usually two peaks in the probability distribution which are influenced by the background and the foreground respectively. The SGA and LGA use spatial aggregation along with appropriate maximum selection to cut down the aggregation of the wrong matching information from the background and therefore suppress the false probability peak appeared at the background's disparity value.

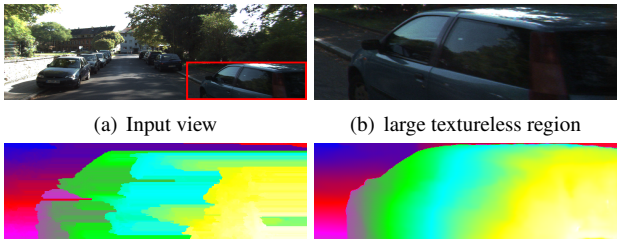
4.3. Comparisons with SGMs and 3D Convolutions

The SGA layer is a differentiable approximation of the SGM [9]. But, it produces far better results compared with both the original SGM with handcrafted features and the MC-CNN [30] with CNN based features (as shown in Table 5). This is because 1) SGA does not have any user-defined parameters that are all learned in an end-to-end fashion. 2) The aggregation of SGA is fully guided and controlled by



(a) large textureless region (sky) (b) reflective region (car window) (c) object edges

Figure 4: Post-softmax probability distributions with respect to disparity values. Red lines illustrate the ground truth disparities. Samples are selected from three challenging regions: (a) the large smooth region (sky), (b) the reflective region from one car window and (c) one region around the object edges. The *first row* shows the probability distributions without GA layers. The *second row* shows the effects of semi-global aggregation (SGA) layers and the *last row* is the refined probabilities with one extra local guided aggregation (LGA) layer.



(a) Input view (b) large textureless region

(c) result of traditional SGM (d) result of our GA-Net-15

Figure 5: Comparisons with traditional SGM. More results and comparisons are avaiable at GA-Net-15 and SGM.

the weight matrices. The guidance subnet learns effective geometrical and contextual knowledge to control the directions, scopes and strengths of the cost aggregations.

Moreover, compared with original SGM, most of the fronto-parallel approximations in large textureless regions have been avoided. (Example is in Fig. 5.) This might be benefited from: 1) the use of the soft weighted sum in Eq. (5) (instead of the hard min/max selection in Eq. (3)); and 2) the regression loss of Eq. (9) which helps achieve the subpixel accuracy.

Our SGA layer is also more efficient and effective than the 3D convolutional layer. This is because the 3D convolutional layer could only aggregate in a local region restricted by the kernel size. As a result, a series of 3D convolutions along with encoder and decoder architectures are indispensable in order to achieve good results. As a comparison, our SGA layer aggregates semi-globally in a single layer which is more efficient. Another advantage of the SGA is that the aggregation's direction, scope and strength are fully guided by variable weights according to different geometrical and contextual information in different locations. *E.g.*, the SGA behaves totally different in the occlusions and the large smoothness regions. But, the 3D convolutional layer has fixed weights and always perform the same for all locations in the whole image.

Table 3: Comparisons with existing real-time algorithms

Methods	End point error	Error rates	Speed (fps)
Our GA-Net	**0.7 px**	**3.21 %**	15 (GPU)
DispNet [15]	1.0 px	4.65 %	22 (GPU)
Toast [20]	1.4 px	7.42 %	25 (CPU)

4.4. Complexity and Real-time Models

The computational complexity of one 3D convolutional layer is $O(K^3CN)$, where N is the elements number of the output blob. K is the size of the convolutional kernel and C is the channel number of the input blob. As a comparison, the complexity of SGA is $O(4KN)$ or $O(8KN)$ for four or eight-direction aggregations. In GC-Net [13] and PSMNet [3], $K = 3$, $C = 32, 64$ or 128 and in our GA-Nets, K is used as 5 (for SGA layer). Therefore, the computational complexity in terms of floating-point operations (FLOPs) of the proposed SGA step is less than 1/100 of one 3D convolutional layer.

The SGA layer are much faster and more effective than 3D convolutions. This allows us to build an accurate real-time model. We implement one caffe [12] version of the GA-Net-1 (with only one 3D convolutional layer and without LGA layers). The model is further simplified by using $4\times$ down-sampling and up-sampling for cost volume. The real-time model could run at a speed of 15$\sim$20 fps for 300×1000 images on a TESLA P40 GPU. We also compare the accuracy of the results with the state-of-the-art real-time models. As shown in Table 3, the real-time GA-Net far outperforms other existing real-time stereo matching models.

4.5. Evaluations on Benchmarks

For the benchmark evaluations, we use the GA-Net-15 with full settings for evaluations. We compare our GA-Net with the state-of-the-art deep neural network models on the Scene Flow dataset and the KITTI benchmarks.

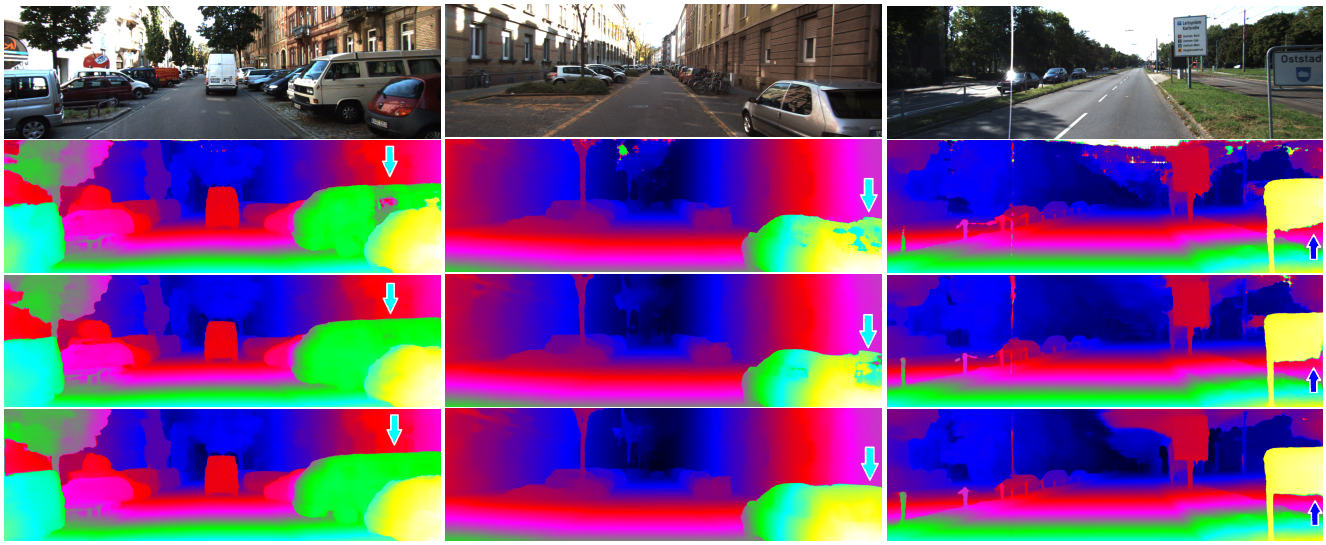


Figure 6: Results visualization and comparisons. *First row:* input image. *Second row:* Results of GC-Net [13]. *Third row:* Results of PSMNet [3]. *Last row:* Results of our GA-Net. Significant improvements are pointed out by blue arrows. The guided aggregations can effectively aggregate the disparity information to the large textureless regions (*e.g.* the cars and the windows) and give precise estimations. It can also aggregate the object knowledge and preserve the depth structure very well (last column).

Table 4: Evaluation Results on KITTI 2012 Benchmark

Models	error rates (2 pixels)	error rates (3 pixels)	Reflective regions	Avg-All (end point)
Our GA-Net	**2.18 %**	**1.36 %**	**7.87%**	**0.5 px**
PSMNet [3]	2.44 %	1.49 %	8.36%	0.6 px
GC-Net [13]	2.71 %	1.77 %	10.80%	0.7 px
MC-CNN [30]	3.90 %	2.43 %	17.09%	0.9 px

Table 5: Evaluation Results on KITTI 2015 Benchmark

Models	Non Occlusion		All Areas	
	Foreground	Avg All	Foreground	Avg All
Our GA-Net-15	**3.39%**	**1.84%**	**3.91%**	**2.03%**
PSMNet [3]	4.31%	2.14 %	4.62%	2.32%
GC-Net [13]	5.58%	2.61%	6.16%	2.87%
SGM-Nets [23]	7.43%	3.09%	8.64%	3.66%
MC-CNN [30]	7.64%	3.33%	8.88%	3.89%
SGM [9]	11.68%	5.62%	13.00%	6.38%

4.5.1 Scene Flow Dataset

The Scene Flow synthetic dataset [15] contains 35,454 training and 4,370 testing images. We use the "final" set for training and testing. GA-Nets are compared with other state-of-the-art DNN models by evaluating with the average end point errors (EPE) and 1-pixel threshold error rates on the test set. The results are presented in Table 2. We find that our GA-Net outperforms the state-of-the-arts on both of the two evaluation metrics by a noteworthy margin (2.2% improvement in error rate and 0.25 pixel improvement in EPE compared with the current best PSMNet [3].).

4.5.2 KITTI 2012 and 2015 Datasets

After training on Scene Flow dataset, we use the GA-Net-15 to fine-tune on the KITTI 2015 and KITTI 2012 data sets respectively. The models are then evaluated on the test sets. According to the online leader board, as shown in Table 4 and Table 5, our GA-Net has fewer low-efficient 3D convolutions but achieves better accuracy. It surpasses current best PSMNet in all the evaluation metrics. Examples are shown in Fig. 6. The GA-Nets can effectively aggregate the correct matching information into the challenging large textureless or reflective regions to get precise estimations. It also keeps the object structures very well.

5. Conclusion

In this paper, we developed much more efficient and effective guided matching cost aggregation (GA) strategies, including the semi-global aggregation (SGA) and the local guided aggregation (LGA) layers for end-to-end stereo matching. The GA layers significantly improve the accuracy of the disparity estimation in challenging regions, such as occlusions, large textureless/reflective regions and thin structures. The GA layers can be used to replace computationally costly 3D convolutions and get better accuracy.

Appendix

A. Backpropagation of SGA

The backpropagation for $\mathbf{w}$ and $C(\mathbf{p},d)$ in SGA (Eq.(5)) can be computed inversely. Assume the gradient from next layer (max-selection) of Eq. (6) is $\frac{\partial E}{\partial C_r^A}$. The backpropagation of SGA can be implemented as:

$$\frac{\partial E}{\partial C(\mathbf{p},d)} = \sum_{\mathbf{r}} \frac{\partial E}{\partial C_r^b(\mathbf{p},d)} \cdot \mathbf{w}_0(\mathbf{p},\mathbf{r}). \tag{10}$$

$$
\begin{aligned}
\frac{\partial E}{\partial \mathbf{w}_0(\mathbf{p},\mathbf{r})} &= \sum_d \frac{\partial E}{\partial C_r^b(\mathbf{p},d)} \cdot C(\mathbf{p},d), \\
\frac{\partial E}{\partial \mathbf{w}_1(\mathbf{p},\mathbf{r})} &= \sum_d \frac{\partial E}{\partial C_r^b(\mathbf{p},d)} \cdot C_{\mathbf{r}}^A(\mathbf{p}-\mathbf{r},d), \\
\frac{\partial E}{\partial \mathbf{w}_2(\mathbf{p},\mathbf{r})} &= \sum_d \frac{\partial E}{\partial C_r^b(\mathbf{p},d)} \cdot C_{\mathbf{r}}^A(\mathbf{p}-\mathbf{r},d-1), \\
\frac{\partial E}{\partial \mathbf{w}_3(\mathbf{p},\mathbf{r})} &= \sum_d \frac{\partial E}{\partial C_r^b(\mathbf{p},d)} \cdot C_{\mathbf{r}}^A(\mathbf{p}-\mathbf{r},d+1), \\
\frac{\partial E}{\partial \mathbf{w}_4(\mathbf{p},\mathbf{r})} &= \sum_d \frac{\partial E}{\partial C_r^b(\mathbf{p},d)} \cdot \max_i C_{\mathbf{r}}^A(\mathbf{p}-\mathbf{r},i).
\end{aligned} \tag{11}
$$

where, $\frac{\partial E}{\partial C_{\mathbf{r}}^b}$ is a temporary gradient variable which can be calculated iteratively by (if $d \neq i_{max}$):

$$
\frac{\partial E}{\partial C_{\mathbf{r}}^b(\mathbf{p},d)} = \frac{\partial E}{\partial C_{\mathbf{r}}^A(\mathbf{p},d)} + \text{sum} \begin{cases} \frac{\partial E}{C_{\mathbf{r}}^b(\mathbf{p}+\mathbf{r},d)} \cdot \mathbf{w}_1(\mathbf{p}+\mathbf{r},\mathbf{r}), \\ \frac{\partial E}{C_{\mathbf{r}}^b(\mathbf{p}+\mathbf{r},d+1)} \cdot \mathbf{w}_2(\mathbf{p}+\mathbf{r},\mathbf{r}), \\ \frac{\partial E}{C_{\mathbf{r}}^b(\mathbf{p}+\mathbf{r},d-1)} \cdot \mathbf{w}_3(\mathbf{p}+\mathbf{r},\mathbf{r}). \end{cases} \tag{12}
$$

or (if $d = i_{max}$):

$$
\frac{\partial E}{\partial C_{\mathbf{r}}^b(\mathbf{p},d)} = \frac{\partial E}{\partial C_{\mathbf{r}}^A(\mathbf{p},d)} + \text{sum} \begin{cases} \frac{\partial E}{C_{\mathbf{r}}^b(\mathbf{p}+\mathbf{r},d)} \cdot \mathbf{w}_1(\mathbf{p}+\mathbf{r},\mathbf{r}), \\ \frac{\partial E}{C_{\mathbf{r}}^b(\mathbf{p}+\mathbf{r},d+1)} \cdot \mathbf{w}_2(\mathbf{p}+\mathbf{r},\mathbf{r}), \\ \frac{\partial E}{C_{\mathbf{r}}^b(\mathbf{p}+\mathbf{r},d-1)} \cdot \mathbf{w}_3(\mathbf{p}+\mathbf{r},\mathbf{r}), \\ \sum_i \frac{\partial E}{C_{\mathbf{r}}^b(\mathbf{p}+\mathbf{r},i)} \cdot \mathbf{w}_4(\mathbf{p}+\mathbf{r},\mathbf{r}). \end{cases} \tag{13}
$$

where i_{max} is the index of $\max_i C_{\mathbf{r}}^A(\mathbf{p},i)$ during the forward propagation in Eq. (5).

B. Details of the Architecture

Table 6 presents the details of the GA-Net-15 which is used in experiments to produce state-of-the-art accuracy on Scene Flow dataset [15] and KITTI benchmarks [7, 16]. It has three SGA layers, two LGA layers and fifteen 3D convolutional layers for cost aggregation.

References

[1] F. Besse, C. Rother, A. Fitzgibbon, and J. Kautz. Pmbp: Patchmatch belief propagation for correspondence field estimation. *International Journal of Computer Vision*, 110(1):2–13, Oct 2014. 2, 3

[2] M. Bleyer, C. Rhemann, and C. Rother. Patchmatch stereo-stereo matching with slanted support windows. In *British Machine Vision Conference (BMVC)*, pages 1–11, 2011. 3

[3] J.-R. Chang and Y.-S. 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

[4] Z. Chen, X. Sun, L. Wang, Y. Yu, and C. Huang. A deep visual correspondence embedding model for stereo matching costs. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 972–980, 2015. 2

[5] X. Cheng, P. Wang, and R. Yang. Depth estimation via affinity learned with convolutional spatial propagation network. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 103–119, 2018. 5

[6] J. Flynn, I. Neulander, J. Philbin, and N. Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5515–5524, 2016. 2

[7] A. Geiger, P. Lenz, and R. 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. 2, 5, 9

[8] K. He, J. Sun, and X. Tang. Guided image filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (6):1397–1409, 2013. 2

[9] H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):328–341, 2008. 1, 2, 3, 6, 8

[10] A. Hosni, C. Rhemann, M. Bleyer, C. Rother, and M. Gelautz. Fast cost-volume filtering for visual correspondence and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(2):504–511, 2013. 1, 2, 3, 4

[11] X. Jia, B. De Brabandere, T. Tuytelaars, and L. V. Gool. Dynamic filter networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 667–675, 2016. 4

[12] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the ACM International Conference on Multimedia*, pages 675–678. ACM, 2014. 5, 7

[13] 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. *CoRR, vol. abs/1703.04309*, 2017. 1, 2, 3, 5, 7, 8

[14] S. Liu, S. De Mello, J. Gu, G. Zhong, M.-H. Yang, and J. Kautz. Learning affinity via spatial propagation networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1520–1530, 2017. 3

[15] 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. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4040–4048, 2016. 1, 2, 5, 7, 8, 9

[16] M. Menze and A. 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, 5, 9

[17] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *Proceedings of the European Conference on Computer Vision*, pages 483–499. Springer, 2016. 1

[18] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 483–499. Springer, 2016. 2

[19] J. Pang, W. Sun, J. S. Ren, C. Yang, and Q. Yan. Cascade residual learning: A two-stage convolutional neural net-

Table 6: Parameters of the network architecture of "GA-Net-15"

No.	Layer Description	Output Tensor
	Feature Extraction	
input	normalized image pair as input	H×W×3
1	3×3 conv	H×W×32
2	3×3 conv, stride 3	$1/3$H×$1/3$W×32
3	3×3 conv	$1/3$H×$1/3$W×32
4	3×3 conv, stride 2	$1/6$H×$1/6$W×48
5	3×3 conv	$1/6$H×$1/6$W×48
6-7	repeat 4-5	$1/12$H×$1/12$W×64
8-9	repeat 6-7	$1/24$H×$1/24$W×96
10-11	repeat 8-9	$1/48$H×$1/48$W×128
12	3×3 deconv, stride 2	$1/24$H×$1/24$W×96
13	3×3 conv	$1/24$H×$1/24$W×96
14-15	repeat 12-13	$1/12$H×$1/12$W×64
16-17	repeat 12-13	$1/6$H×$1/6$W×48
18-19	repeat 12-13	$1/3$H×$1/3$W×32
20-35	repeat 4-19	$1/3$H×$1/3$W×32
concat connection	(9,12), (7,14), (5,16), (3,18), (17,20), (15,22), (13,24), (11,26), (18,25) (25,28), (23,30) (21,32), (19,34)	
cost volume	by feature concatenation	$1/3$H×$1/3$W×64×32
	Guidance Branch	
input	concat 1 and up-sampled 35 as input	H×W×64
(1)	3×3 conv	H×W×16
(2)	3×3 conv, stride 3	$1/3$H×$1/3$W×32
(3)	3×3 conv	$1/3$H×$1/3$W×32
(4)	3×3 conv (no bn & relu)	$1/3$H×$1/3$W×640
(5)	split, reshape, normalize	4× $1/3$H×$1/3$W×5×32
(6)	from (3), 3×3 conv	$1/3$H×$1/3$W×32
(7)	3×3 conv (no bn & relu)	$1/3$H×$1/3$W×640
(8)	split, reshape, normalize	4× $1/3$H×$1/3$W×5×32
(9)-(11)	from (6), repeat (6)-(8)	4× $1/3$H×$1/3$W×5×32
(12)	from (1), 3×3 conv	H×W×16
(13)	3×3 conv (no bn & relu)	H×W×75
(14)	split, reshape, normalize	H×W×75
(15)-(17)	from (12), repeat (12)-(14)	H×W×75
	Cost Aggregation	
input	4D cost volume	$1/3$H×$1/3$W×48×64
[1]	3×3×3, 3D conv	$1/3$H×$1/3$W×48×32
[2]	**SGA** layer: weight matrices from (5)	$1/3$H×$1/3$W×48×32
[3]	3×3×3, 3D conv	$1/3$H×$1/3$W×48×32
output	3×3×3, 3D to 2D conv, upsamping	H×W×193
	softmax, regression, loss weight: 0.2	H×W×1
[4]	3×3×3, 3D conv, stride 2	$1/6$H×$1/6$W×48×48
[5]	3×3×3, 3D conv, stride 2	$1/12$H×$1/12$W×48×64
[6]	3×3×3, 3D deconv, stride 2	$1/6$H×$1/6$W×48×48
[7]	3×3×3, 3D conv	$1/6$H×$1/6$W×48×48
[8]	3×3×3, 3D deconv, stride 2	$1/3$H×$1/3$W×48×32
[9]	3×3×3, 3D conv	$1/3$H×$1/3$W×48×32
[10]	**SGA** layer: weight matrices from (8)	$1/3$H×$1/3$W×48×32
output	3×3×3, 3D to 2D conv, upsamping	H×W×193
	softmax, regression, loss weight: 0.6	H×W×1
[11]	3×3×3, 3D conv, stride 2	$1/6$H×$1/6$W×48×48
[12]	3×3×3, 3D conv	$1/6$H×$1/6$W×48×48
[13]	3×3×3, 3D conv, stride 2	$1/12$H×$1/12$W×48×64
[14]	3×3×3, 3D deconv, stride 2	$1/6$H×$1/6$W×48×48
[15]	3×3×3, 3D conv	$1/6$H×$1/6$W×48×48
[16]	3×3×3, 3D deconv, stride 2	$1/3$H×$1/3$W×48×32
[17]	3×3×3, 3D conv	$1/3$H×$1/3$W×48×32
[18]	**SGA** layer: weight matrixes from (11)	$1/3$H×$1/3$W×48×32
	3×3×3, 3D to 2D conv, upsamping	H×W×193
final output	**LGA**, softmax, **LGA**: weights from (14), (17) regression, loss weight: 1.0	H×W×193 H×W×1
connection	concat: (4,6), (3,8), (7,11), (5,13), (12,14), (10,16); add: (1,3)	

work for stereo matching. *IEEE International Conference on Computer Vision Workshops (ICCVW)*, 2017. 2

[20] B. Ranft and T. Strauß. Modeling arbitrarily oriented slanted planes for efficient stereo vision based on block matching. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, pages 1941–1947. IEEE, 2014. 7

[21] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1-3):7–42, 2002. 1

[22] J. L. Schönberger, S. N. Sinha, and M. Pollefeys. Learning to fuse proposals from multiple scanline optimizations in semi-global matching. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 739–755, 2018. 2

[23] A. Seki and M. Pollefeys. Sgm-nets: Semi-global matching with neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6640–6649, 2017. 2, 3, 8

[24] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014. 3

[25] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 839–846. IEEE, 1998. 2

[26] S. Tulyakov, A. Ivanov, and F. Fleuret. Practical deep stereo (pds): Toward applications-friendly deep stereo matching. *arXiv preprint arXiv:1806.01677*, 2018. 2

[27] S. Xu, F. Zhang, X. He, X. Shen, and X. Zhang. Pm-pm: Patchmatch with potts model for object segmentation and stereo matching. *IEEE Transactions on Image Processing*, 24(7):2182–2196, July 2015. 2

[28] Q. 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. 2

[29] S. Zagoruyko and N. Komodakis. Learning to compare image patches via convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4353–4361, 2015. 2

[30] J. Zbontar and Y. 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, 3, 7, 8

[31] F. Zhang, L. Dai, S. Xiang, and X. 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. 2

[32] F. Zhang and B. W. Wah. Supplementary meta-learning: Towards a dynamic model for deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 4344–4353, 2017. 4

[33] F. Zhang and B. W. Wah. Fundamental principles on learning new features for effective dense matching. *IEEE Transactions on Image Processing*, 27(2):822–836, 2018. 1, 2