Adaptive Unimodal Cost Volume Filtering for Deep Stereo Matching

Youmin Zhang,[1*] **Yimin Chen,**[1*] **Xiao Bai,**[1†] **Suihanjin Yu,**[1] **Kun Yu,**[2] **Zhiwei Li,**[2] **Kuiyuan Yang**[2]
[1]State Key Laboratory of Software Development Environment, School of Computer Science and Engineering,
Beijing Advanced Innovation Center for Big Data and Brain Computing,
Jiangxi Research Institute, Beihang University, Beijing, China [2]DeepMotion
{youmi, minwellcym, baixiao, fakecoderemail}@buaa.edu.cn {kunyu, zhiweili, kuiyuanyang}@deepmotion.ai

Abstract

State-of-the-art deep learning based stereo matching approaches treat disparity estimation as a regression problem, where loss function is directly defined on true disparities and their estimated ones. However, disparity is just a byproduct of a matching process modeled by cost volume, while indirectly learning cost volume driven by disparity regression is prone to overfitting since the cost volume is under constrained. In this paper, we propose to directly add constraints to the cost volume by filtering cost volume with unimodal distribution peaked at true disparities. In addition, variances of the unimodal distributions for each pixel are estimated to explicitly model matching uncertainty under different contexts. The proposed architecture achieves state-of-the-art performance on Scene Flow and two KITTI stereo benchmarks. In particular, our method ranked the 1^{st} place of KITTI 2012 evaluation and the 4^{th} place of KITTI 2015 evaluation (recorded on 2019.8.20). The codes of AcfNet are available at: https://github.com/youmi-zym/AcfNet.

Introduction

Stereo matching is one of the core technologies in computer vision, which recovers 3D structures of real world from 2D images. It has been widely used in areas such as autonomous driving (Sivaraman and Trivedi 2013), augmented reality (Zenati and Zerhouni 2007) and robotics navigation (Schmid et al. 2013; Luo, Yu, and Ren 2017; Luo et al. 2019). Given a pair of rectified stereo images, the goal of stereo matching is to compute the disparity d for each pixel in the reference image (usually refers to the left image), where disparity is defined as the horizontal displacement between a pair of corresponding pixels in the left and right images.

According to the seminar work (Scharstein and Szeliski 2002), a stereo matching algorithm typically consists of four steps: matching cost computation, cost aggregation, disparity regression and disparity refinement. Among them, matching cost computation, i.e., obtaining cost volume, is arguably the most crucial first step. A cost volume is usually denoted by a $H \times W \times D$ tensor, where H, W, D are the height, width and maximal disparity of the reference image. In traditional methods, cost volume is computed by a predefined cost function of manually designed image features, e.g., squared or absolute difference of image patches.

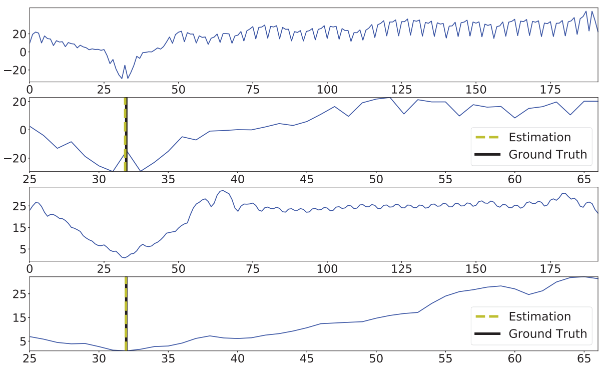

Figure 1: An example of cost distribution along the disparity dimension of cost volume. The first row is the output of PSMNet (Chang and Chen 2018) trained with *soft argmin*. The third row is output of our model. For better visualization, we also zoom into the disparity interval $[25, 66]$ in the second and fourth row, where Estimation and Ground Truth are the estimated and the groundtruth disparity respectively. Our method generates a more reasonable cost distribution peaked at the true disparity.

In the deep learning era, both image feature and cost function are modeled as network layers (Kendall et al. 2017; Chang and Chen 2018). To make all layers differentiable and achieve sub-pixel estimation of disparity, *soft argmin* is used to estimate disparity by softly weighting indices according to their costs, which is in contrast to *argmin* that takes the index with minimal cost as estimated disparity. The loss function is defined on the estimated disparity and the ground truth for end-to-end training. Benefit from large-scale training data and end-to-end training, deep learning based stereo approaches achieve state-of-the-art performance.

In the deep learning models, the cost volume is indi-

[*]Joint first authorship. {youmi,minwellcym}@buaa.edu.cn.
[†]Corresponding author, baixiao@buaa.edu.cn.

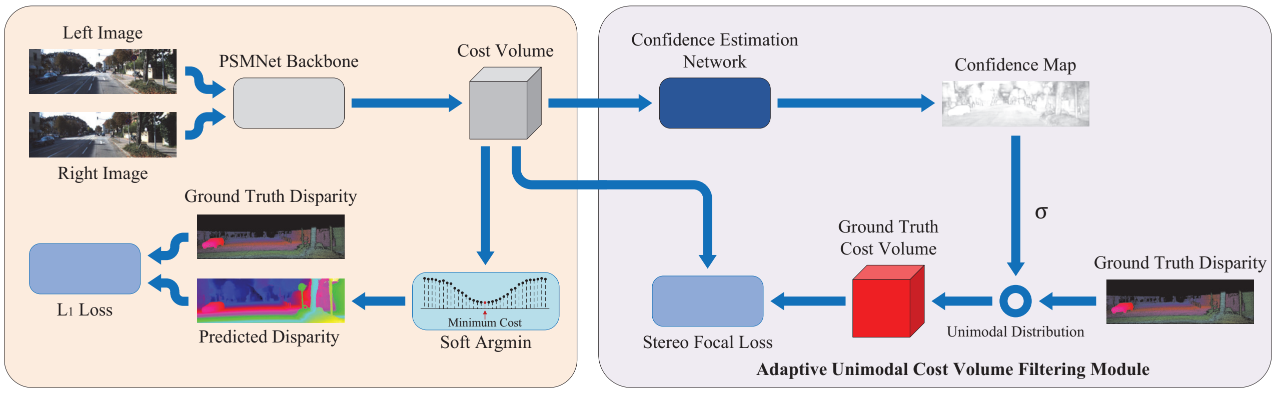

Figure 2: Architecture of the proposed end-to-end AcfNet. The input stereo images are fed to PSMNet (Chang and Chen 2018) backbone with stacked hourglass architecture to get three cost volumes after aggregation. For each cost volume, we generate the confidence map by a Confidence Estimation Network (CENet), and modulate the ground truth cost volume with confidence values to generate pixel-wise unimodal distribution as training labels. The proposed *Stereo Focal Loss* is added to the cost volume using the generated training labels. Finally, a sub-pixel disparity map is estimated by the *soft argmin* function followed by regression loss as PSMNet.

rectly supervised as an intermediate layer, which leaves cost volume less constrained since infinitely many cost distributions can generate the same disparity, where only cost distributions peaked at the true disparity are reasonable ones. Accordingly, we propose to directly supervise cost volume with unimodal ground truth distributions. To reveal network matching uncertainties (Kendall and Gal 2017; Ilg et al. 2018) of different pixels, we design a confidence estimation network to estimate per-pixel confidence and control sharpness of the unimodal ground truth distributions accordingly. Figure 1 compares the cost distributions at the same pixel by PSMNet and our method, where our method generates the correct minimal cost around the true disparity, while PSMNet generates two local minimal costs away from the true disparity.

We evaluate the proposed **A**daptive unimodal **c**ost volume **f**iltering **Net**work (AcfNet) on three stereo benchmarks including Scene Flow, KITTI 2012 and KITTI 2015. Ablation studies and detailed analysis on Scene Flow demonstrate the effectiveness of AcfNet. We also submit our stereo matching results to KITTI 2012 and 2015 evaluation server, and ranked the 1^{st} place on KITTI 2012 evaluation and the 4^{th} place on KITTI 2015 evaluation (recorded on 2019.8.20).

Related Work

Deep learning for stereo matching starts from learning image features for classical methods (Zbontar and LeCun 2016; Luo, Schwing, and Urtasun 2016). DispNetC (Mayer et al. 2016) is the first breakthrough for stereo matching by proposing an end-to-end trainable network, where cost function is predefined as a correlation layer in the network to generate the cost volume, then a set of convolutional layers are added to the cost volume to regress disparity map. Based on DispNetC, stack refinement sub-networks are proposed to improve the performance (Pang et al. 2017; Liang et al. 2018), and the performance could be further improved by using additional information such edges (Song et al. 2018) and semantics (Yang et al. 2018). To add more capacity for network to learn the cost function, (Guo et al. 2019) propose to use group-wise correlation layer and generate multiple cost volumes for latter aggregation.

GC-Net (Kendall et al. 2017) gives more flexibility for network to learn cost function by using 3D convolutional layers on concatenated feature volume, with cost volume produced by the learned cost function, disparity is estimated by *soft argmin* according to the cost distribution. Follow-up works improve results by using better image features (Chang and Chen 2018) and cost aggregation layers inspired by classical methods (Cheng, Wang, and Yang 2018; Zhang et al. 2019). In these end-to-end stereo matching networks, cost volume is the output of an intermediate layer without direct supervision, which leaves the possibilities to learn unreasonable cost distributions as illustrated in Figure 1.

In this work, the proposed AcfNet directly adds supervision to the cost volume estimation using ground truth cost distributions peaked at true disparities. In addition, the sharpness of ground truth cost distribution is adjusted according to matching confidence. Concurrent to our work, sparse LiDAR points are used to enhance cost volume by weighting estimated cost distribution with a Gaussian distribution centered at the disparity provided by its corresponding LiDAR point (Poggi et al. 2019), which serves as a multi-sensor fusion method for disparity estimation. In contrast, our method only takes images as input during both training and testing, and unimodal supervision is added to each pixel in a dense and adaptive way.

AcfNet

Figure 2 illustrates the overall framework, where the proposed adaptive unimodal cost volume filtering module is applied to the cost volume, and an additional loss is introduced to directly supervise the learning of cost volume towards desired property. Here, we choose PSMNet (Chang and Chen 2018) as the basic network to calculate cost volume for its state-of-the-art performance on stereo matching.

Overview

Given a pair of rectified images, for each pixel $p = (x, y)$ in the left image, stereo matching aims to find its corresponding pixel in the right image, i.e., $p' = (x + d, y), d \in \mathbb{R}^+$, where disparity d is often represented by a floating-point number for sub-pixel matching. For both computation and memory tractable, disparity is discrete into a set of possible disparities, i.e., $\{0, 1, \cdots, D-1\}$ to build an $H \times W \times D$ cost volume, where H, W and D are the image height, width and maximum disparity respectively. To recover sub-pixel matching, costs over disparities are used in a weighted interpolation. The whole process is implemented through a network as illustrated in the left part of Figure 2.

Formally, the cost volume contains D costs for each pixel denoted by $\{c_0, c_1, \cdots, c_{D-1}\}$, and the sub-pixel disparity is estimated through *soft argmin* (Kendall et al. 2017)

$$\hat{d} = \sum_{d=0}^{D-1} d \times \hat{P}(d), \tag{1}$$

where $\hat{P}(d) = \frac{\exp(-c_d)}{\sum_{d'=0}^{D-1} \exp(-c_{d'})}$, and disparities with small cost contribute more during interpolation. Given the groundtruth disparity d_p for each pixel p, smooth L_1 loss is defined for training, i.e.,

$$\mathcal{L}_{regression} = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} smooth_{L_1}(d_p - \hat{d}_p), \tag{2}$$

where

$$smooth_{L_1}(x) = \begin{cases} 0.5x^2, & \text{if } |x| < 1, \\ |x| - 0.5, & \text{otherwise.} \end{cases} \tag{3}$$

The whole process is differentiable by supervising with the groundtruth disparity, while cost volume is indirectly supervised through providing weights for disparity interpolation. However, the supervision is underdetermined and there could be infinitely possible sets of weights to achieve correct interpolation results. The flexibility of cost volume is prone to overfitting since many improperly learned cost volumes could interpolate disparities close to ground truth (i.e., small training loss).

To address this problem raised from indirectly supervising cost volume with underdetermined loss function, we propose to directly supervise the cost volume according to its unimodal property.

Unimodal distribution

Cost volume is defined to reflect the similarities between candidate matching pixel pairs, where the true matched pair should have the lowest cost (i.e., the highest similarity), and the costs should increase with the distance to the truly matched pixel. This property requires unimodal distribution be peaked at the true disparity at each position in the cost volume. Given the ground truth disparity d^{gt}, the unimodal distribution is defined as

$$\begin{aligned} P(d) &= \text{softmax}\left(-\frac{|d - d^{gt}|}{\sigma}\right) \\ &= \frac{\exp(-c_d^{gt})}{\sum_{d'=0}^{D-1} \exp(-c_{d'}^{gt})}, \end{aligned} \tag{4}$$

where $c_d^{gt} = \frac{|d - d^{gt}|}{\sigma}$, $\sigma > 0$ is the variance (a.k.a temperature in literature) that controls the sharpness of the peak around the true disparity.

The ground truth cost volume constructed from $P(d)$ has the same sharpness of peaks across different pixels, which cannot reflect similarity distribution differences across different pixels. For example, a pixel on the table corner should have a very sharp peak while pixels in uniform regions should have relative flat peaks. To build such more reasonable labels for cost volume, we add a confidence estimation network to adaptively predict σ_p for each pixel.

Confidence estimation network

Considering matching properties are embedded in the estimated cost volume (Fu and Fard 2018; Park and Yoon 2018; Kim et al. 2018), then the confidence estimation network takes the estimated cost volume as input, and uses a few layers to determine the matching confidence of each pixel by checking the matching states in a small neighborhood around each pixel. Specifically, the network employs a 3×3 convolutional layer followed by batch normalization and ReLU activation, and another 1×1 convolutional layer followed by sigmoid activation to produce a confidence map $f \in [0, 1]^{H \times W}$, where a pixel p with large confidence f_p means a unique matching can be confidently found for this pixel, while small confidence values denote there are matching ambiguities. Then, σ_p for generating ground truth cost distribution is scaled from the estimated confidence,

$$\sigma_p = s(1 - f_p) + \epsilon, \tag{5}$$

where $s \geq 0$ is a scale factor that reflects the sensitivity of σ to the change of confidence f_p, $\epsilon > 0$ defines the lower bound for σ and avoids numerical issue of dividing 0. Accordingly, $\sigma_p \in [\epsilon, s + \epsilon]$. Our experiments show that two kinds of pixels are likely to have large σ, i.e., texture-less pixels and occluded pixels, where the texture-less pixels tend to have multiple matches, while occluded pixels have no correct matches. With the per-pixel adpatively estimated σ_p, the ground truth cost volume defined in Eq. (4) is modified accordingly.

Stereo focal loss

At pixel position p, we now have both estimated cost distribution $\hat{P}_p(d)$ and the ground truth $P_p(d)$. It is straightforward to define a distribution loss via cross entropy. However, there is a severe sample imbalance problem since each pixel has only one true disparity (positive) comparing with hundreds of negative ones (Zbontar and LeCun 2016). Similar to focal loss designed to solve the sample imbalance problem in one-stage object detection (Lin et al. 2017), we design a stereo focal loss to focus on positive disparities to avoid the total loss dominated by negative disparities,

$$\mathcal{L}_{SF} = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \left(\sum_{d=0}^{D-1} (1 - P_p(d))^{-\alpha} \cdot \left(-P_p(d) \cdot \log \hat{P}_p(d) \right) \right), \tag{6}$$

where $\alpha \geq 0$ is a *focusing* parameter, and the loss is reduced to cross entropy loss when $\alpha = 0$, while $\alpha > 0$ gives more weights to positive disparities in proportion to their $P_p(d)$. Thus easy negative disparities are further suppressed explicitly with quite small weights and let the positive disparity only compete with a few hard ones.

Total loss function

In sum, our final loss function contains three parts defined as

$$\begin{aligned} \mathcal{L} = \mathcal{L}_{SF} &+ \lambda_{regression} \mathcal{L}_{regression} \\ &+ \lambda_{confidence} \mathcal{L}_{confidence}, \end{aligned} \tag{7}$$

where $\lambda_{regression}, \lambda_{confidence}$ are two trade-off hyper-parameters. $\mathcal{L}_{SF}$ supervises the cost volume while $\mathcal{L}_{regression}$ supervises the disparity. $\mathcal{L}_{confidence}$ is added as a regularizer to encourage more pixels to have high confidence values,

$$\mathcal{L}_{confidence} = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} -\log f_p. \tag{8}$$

Experiments and Analysis

Implementation details

Our network is implemented using PyTorch (Paszke et al. 2017) framework, and all models are end-to-end trained using RMSprop with standard settings. Our data processing is the same as PSMNet (Chang and Chen 2018). We train our models from scratch using the Scene Flow dataset with a constant learning rate of 0.001 for 10 epochs. For Scene Flow, the trained model is directly used for testing. For KITTI, we use the model trained with Scene Flow data after fine-tuning on the KITTI training set for 600 epochs. The learning rate of this fine-tuning begins at 0.001 and is decayed by $\frac{1}{3}$ at 100 and 300 epochs. For submission to the KITTI test benchmark, we prolong the training process on Scene Flow with a constant learning rate of 0.001 for 20 epochs to obtain a better pre-training model. The batch size is set to 3 for training on 3 NVIDIA GTX 1080Ti GPUs. All ground truth disparities out of range of $[0, D-1]$ are excluded in our experiments, where $D = 192$.

Datasets

We evaluate AcfNet qualitatively and quantitatively on three challenging stereo benchmarks, i.e., Scene Flow (Mayer et al. 2016), KITTI 2012 (Geiger, Lenz, and Urtasun 2012) and KITTI 2015 (Menze and Geiger 2015).

Scene Flow: Scene Flow is a large synthetic dataset containing 35,454 training image pairs and 4,370 testing image pairs, where the ground truth disparity maps are densely provided, which is large enough for directly training deep learning models. Following the setting as GC-Net (Kendall et al. 2017), we mainly use this dataset for ablation study.

KITTI: KITTI 2015 and KITTI 2012 are two real-world datasets with street views captured from a driving car. KITTI 2015 contains 200 training stereo image pairs with sparse groundtruth disparities obtained using LiDAR and 200 testing image pairs with ground truth disparities held by evaluation server for submission evaluation only. KITTI 2012 contains 194 training image pairs with sparse ground truth disparities and 195 testing image pairs with ground truth disparities held by evaluation server for submission evaluation only. These two datasets are challenging due their small size.

Metrics: The performance is measured using two standard metrics: (1) 3-Pixel-Error (3PE), i.e., the percentage of pixels for which the predicted disparity is off the true one by more than 3 pixels, and (2) End-Point-Error (EPE), i.e., the average difference of the predicted disparities and their true ones. 3PE is robust to outliers with large disparity errors, while EPE measures errors to sub-pixel level.

To further evaluate the ability on handling occluded regions, we divide the testing images of Scene Flow into occluded region (OCC) and non-occluded regions (NOC) through left-right consistency check. In total, there are 16% occluded pixels in all pixels. The performance is measured on all pixels if no prefix such as OCC, NOC and ALL are added before 3PE or EPE.

Ablation studies

We conduct ablation studies on Scene Flow (Mayer et al. 2016) considering it has large enough training data for end-to-end training from scratch. In all experiments, α is set to 5.0 in stereo focal loss to balance positive and negative samples. Considering disparities of most pixels are with sub-pixel errors (i.e., error smaller than one pixel) while 3PE cannot reveal errors within 3 pixels, we use EPE to study the performance variance for different hyper-parameter settings.

Table 1: Results of comparison between stereo focal loss and cross entropy loss in our model AcfNet.

AcfNet	+ Cross Entropy Loss	+ Stereo Focal Loss
EPE [px]	0.965	**0.920**

The variance σ of unimodal distribution

The variance σ adjusts the shape of unimodal distribution, which plays an important role in AcfNet. In our method, $\sigma \in [\epsilon, s + \epsilon]$ is bounded by s and ϵ.

Firstly, we study the case when the variance σ is fixed for all pixels, i.e. $s = 0, \sigma = \epsilon$. By grid search, we find that $\sigma = 1.2$ achieves the best result, which indicates most pixels favor $\sigma = 1.2$ for building unimodal distributions. Thus, we set the lower bound ϵ of σ to 1.0 for adaptive variance study. Furthermore, we compare the stereo focal loss with cross entropy loss under this condition, i.e. $\sigma = 1.2$. As shown in Table 1, equipping AcfNet with stereo focal loss get a significantly better result than cross entropy loss, which demonstrates the effectiveness of stereo focal loss in balancing losses from positive and negative disparities.

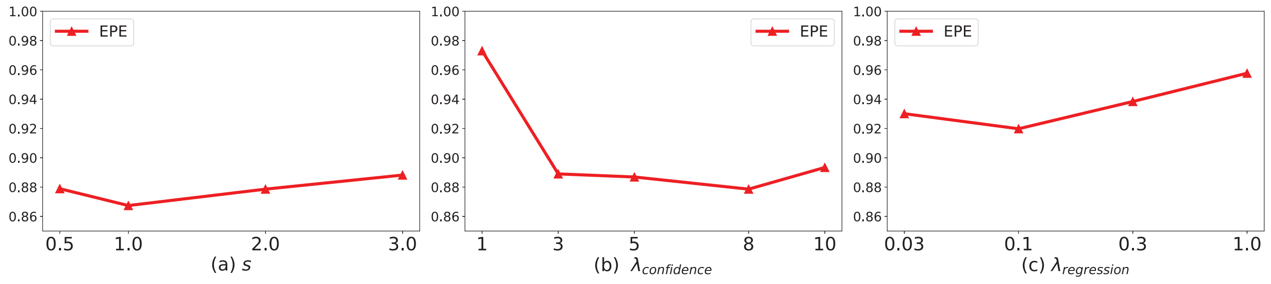

Figure 3: Ablation study results for different hyper-parameters in our method, where s controls the upper bound of variance σ. $\lambda_{confidence}$ and $\lambda_{regression}$ are balance weights for confidence loss and disparity regression loss respectively.

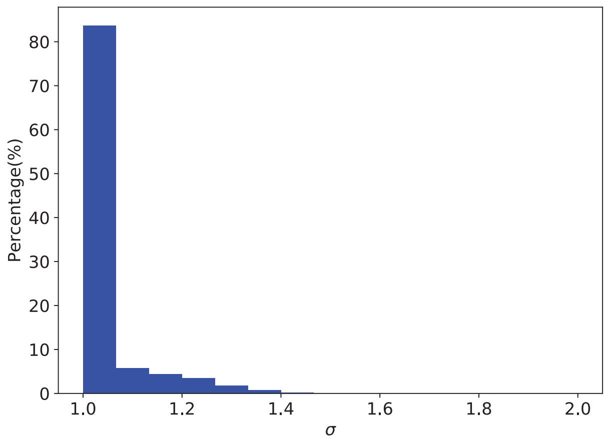

Figure 4: Histogram distribution of variance σ on the whole test dataset of Scene Flow after AcfNet has been converged.

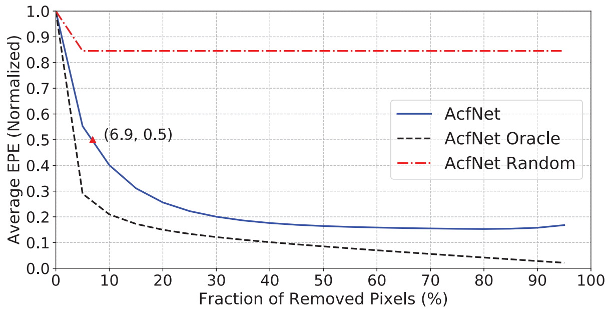

Figure 5: Sparsification plot of our AcfNet on the Scene Flow test dataset. The plot shows the normalized average end-point-error (EPE) for each fraction of pixels with highest variances has been removed. The curve 'AcfNet Oracle' shows the ideal case by removing each fraction of pixels ranked by the ground truth EPE. The curve 'AcfNet Random' shows the worst case by removing each fraction of pixels randomly. Removing only 6.9% of the pixels by AcfNet results in halving the average EPE.

Secondly, we study the sensitivity s which controls the upper bound of σ. Figure 3(a) shows the performance by varying s, where $s = 1$ performs best and the performance is rather stable by varying s from 0.5 to 3.0. Figure 4 shows the histogram of σ when $s = 1$ (i.e., $\sigma \in [1.0, 2.0]$), where most pixels favor small variances, i.e., sharp distributions, and a long tail of pixels require larger variances for flatten distributions.

Loss balance weights

Hyperparameter $\lambda_{confidence}$ balances the total variance and other losses. Figure. 3(b) shows the performance curve by varying $\lambda_{confidence}$, where both overconfident learning with large $\lambda_{confidence}$ and underconfident learning with small $\lambda_{confidence}$ lead to inferior performance while $\lambda_{confidence} = 8.0$ performs the best.

Hyperparameter $\lambda_{regression}$ balances the regression loss that is widely used in recent state-of-the-art models, and large value for $\lambda_{regression}$ will eliminate effects of the other two losses proposed in this paper. Figure 3(c) shows the performance curve, it could be observed that regression loss can be improved through proper tradeoff with the proposed two losses.

Variance analysis

Variance estimation is an important component of our cost filtering scheme, which automatically adjusts the flatness of the unimodal distribution according to the matching uncertainty. To assess the quality of the estimated variances, sparsification plot (Ilg et al. 2018) is adopted to reveal the relevance of the estimated variances with the true errors through plotting evaluation results by gradually removing pixels according their variances. For comparison, we also plot the curves of randomly assigned variances (AcfNet Random) and variances assigned by EPE errors (AcfNet Oracle) in Figure 5, where the estimated variances are highly relevant to EPE errors and demonstrates the ability of AcfNet in explaining outlier pixels with estimated variances.

Figure 6 shows several per-pixel results from Scene Flow, where hard regions mainly appear at occlusions (1a, 1c and 2a), repeated patterns (1b, 3a) and thin structures (3a). In these hard regions, AcfNet provides high variances to flatten the corresponding cost distributions. AcfNet can balance the learning for different pixels, and pushes informative pixels towards high confidences (i.e, low variances), while allows hard uninformative pixels with high variances to avoid

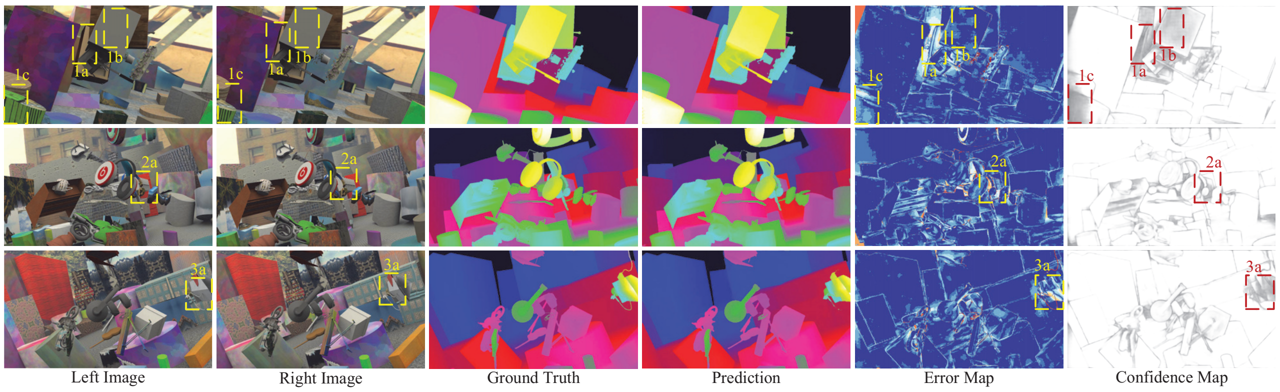

Figure 6: Qualitative results on three samples from Scene Flow test set. Columns from left to right are: left stereo input image, right stereo input image, disparity ground truth, disparity prediction, error map and confidence map. Cold colors in the error map denote small prediction errors while warm colors denote large prediction errors. In confidence map, bright colors mean small variances while dark colors denote high variances.

overfitting.

Table 2: Evaluation of adaptive unimodal cost volume filtering results, where PSMNet is re-implemented. AcfNet(uniform) denotes setting a uniform unimodal distribution for all pixels, and AcfNet(adaptive) denotes adaptively adjust the per-pixel variances.

Method	Scene Flow					
	EPE [px]			3PE [%]		
	ALL	OCC	NOC	ALL	OCC	NOC
PSMNet	1.101	3.507	0.637	4.56	17.64	2.12
AcfNet (uniform)	0.920	2.996	0.504	4.39	16.47	**2.10**
AcfNet (adaptive)	**0.867**	**2.736**	**0.495**	**4.31**	**15.77**	2.13

Adaptive unimodal cost volume filtering

AcfNet adds direct cost volume supervision to PSMNet. Table 2 compares two versions of AcfNet with PSMNet, where uniform version of AcfNet is significantly better than PSMNet and adaptive version of AcfNet further improves the performance significantly. The results demonstrate the effectiveness of unimodal supervision and adaptive per-pixel variance estimation. Comparing with AcfNet(uniform), AcfNet(adaptive) improves more on OCC (i.e., occluded regions), which is consistent with conclusion in variance analysis.

Table 3: Results of cost volume filtering comparison, where all methods are trained on Scene Flow from scratch using the same base model PSMNet, and directly test on KITTI 2012, 2015 training datasets. * denotes disparities of sparse LiDAR points are also used as model input when testing.

Method	EPE[px]	3PE[%]	
	Scene Flow	KITTI 2012	KITTI 2015
PSMNet	1.101	29.18	30.19
(Poggi et al. 2019)	0.991*	-	23.13*
AcfNet	**0.867**	**17.54**	**19.45**

Cost volume filtering comparisons

To further validate the superiority of the proposed cost volume filtering, experiments are designed to compare with the concurrent work (Poggi et al. 2019). In contrast to our work, (Poggi et al. 2019) uses disparities by sparse LiDAR points to filter cost volume during both training and testing. Both AcfNet and the method of (Poggi et al. 2019) are trained on Scene Flow from scratch, and directly evaluated on training sets of KITTI 2012 and 2015 since (Poggi et al. 2019) requires sparse LiDAR points as inputs. Table 3 reports the comparison results, where AcfNet outperforms (Poggi et al. 2019) on all performance metrics by large margins even without using LiDAR points as inputs. In addition, comparing with PSMNet, AcfNet shows much better generalization performance from Scene Flow to KITTI, which further proves the ability of AcfNet in preventing overfitting.

Comparisons with the state-of-the-art methods

To further validate the proposed AcfNet, Table 4 compares AcfNet with state-of-the-art methods on both KITTI 2012 and 2015, where AcfNet outperforms others by notable margins on all evaluation metrics. To be noted, Scene Flow is used for pretraining in all methods considering the small size of KITTI training data. Figure 7 and 8 show several exemplar results from KITTI 2015 and 2012 by comparing AcfNet with PSMNet (Chang and Chen 2018) and PDS (Tulyakov, Ivanov, and Fleuret 2018), where significantly improved regions are marked out with dash boxes. As expected, most improvements of AcfNet come from challenging areas such as thin structures, sky boundaries and image borders.

Conclusions

In this paper, we solve the under-constrain problem of cost volume in existing deep learning based stereo matching approaches. The proposed AcfNet supervises the cost volume with ground truth unimodal distributions peaked at true disparities, and variances for per-pixel distributions are adap-

Table 4: Results on Scene Flow and KITTI Benchmarks. Following standard setting, on KITTI 2012, percentages of erroneous pixels for both Non-occluded (Out-Noc) and all (Out-All) pixels are reported, on KITTI 2015, percentages of disparity outliers D_1 averaged over all ground truth pixels (D1-all) for both Non-occluded and All pixels are reported. The outliers are defined as those pixels whose disparity errors are larger than $\max(3\text{px}, 0.05d^{gt})$, where d^{gt} is the ground-truth disparity.

Method	Scene Flow	KITTI 2012								KITTI 2015	
	EPE	2px		3px		4px		5px		ALL	NOC
		Out-Noc	Out-All	Out-Noc	Out-All	Out-Noc	Out-All	Out-Noc	Out-All	D1-all	D1-all
MC-CNN (Zbontar and LeCun 2016)	3.79	3.90	5.45	2.43	3.63	1.90	2.85	1.64	2.39	3.88	3.33
GC-Net (Kendall et al. 2017)	2.51	2.71	3.46	1.77	2.30	1.36	1.77	1.12	1.46	2.67	2.45
iResNet-i2 (Liang et al. 2018)	1.40	2.69	3.34	1.71	2.16	1.30	1.63	1.06	1.32	2.44	2.19
PSMNet (Chang and Chen 2018)	1.09	2.44	3.01	1.49	1.89	1.12	1.42	0.90	1.15	2.32	2.14
SegStereo (Yang et al. 2018)	1.45	2.66	3.19	1.68	2.03	1.25	1.52	1.00	1.21	2.25	2.08
PDS (Tulyakov, Ivanov, and Fleuret 2018)	1.12	3.82	4.65	1.92	2.53	1.38	1.85	1.12	1.51	2.58	2.36
GwcNet-gc (Guo et al. 2019)	**0.77**	2.16	2.71	1.32	1.70	0.99	1.27	0.80	1.03	2.21	1.92
HD^3-Stereo (Yin, Darrell, and Yu 2019)	1.08	2.00	2.56	1.40	1.80	1.12	1.43	0.94	1.19	2.02	1.87
GA-Net (Zhang et al. 2019)	0.84	2.18	2.79	1.36	1.80	1.03	1.37	0.83	1.10	1.93	1.73
AcfNet	0.87	**1.83**	**2.35**	**1.17**	**1.54**	**0.92**	**1.21**	**0.77**	**1.01**	**1.89**	**1.72**

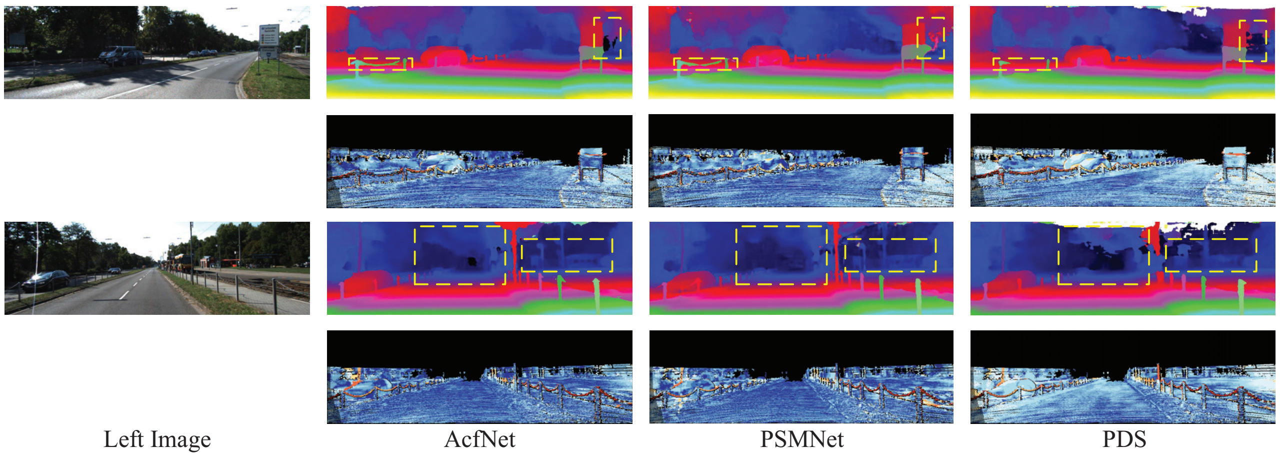

Figure 7: Visualization results on the KITTI 2015 dataset. Significantly improved regions are highlighted with dash boxes. For each example, the first row shows the disparity map, and the second row shows the error map. Warmer color indicate larger prediction errors.

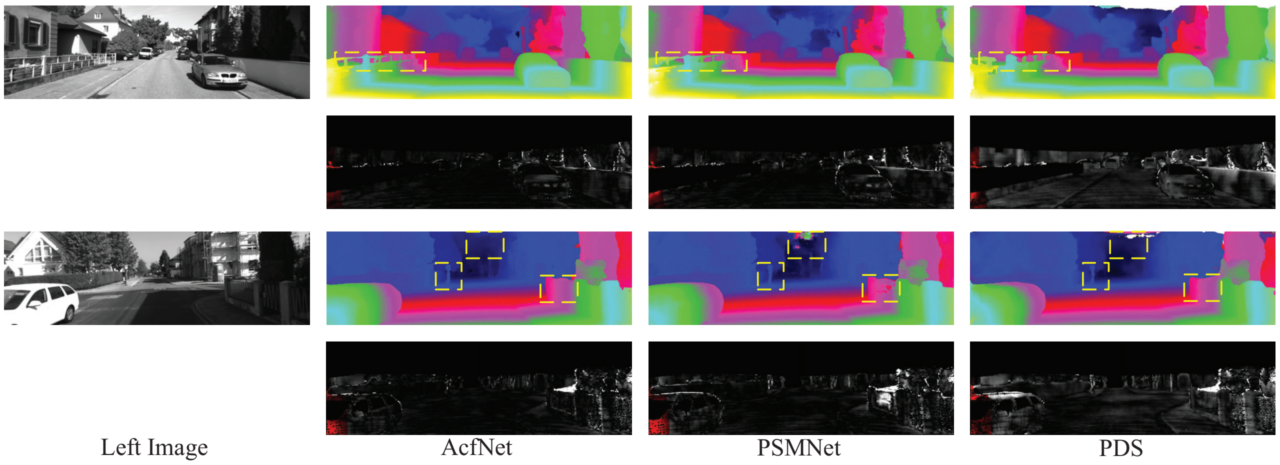

Figure 8: Visualization results on the KITTI 2012 dataset. Significantly improved regions are highlighted with dash boxes. For each example, the first row shows the disparity map, and the second row shows the error map, bright colors indicate inaccurate predictions.

tively estimated to modulate the learning according the informativeness of each pixel. AcfNet shows better testing performance on the same dataset and even superior performance on cross-dataset evaluation.

Acknowledgements

This work was supported by the National Natural Science Foundation of China project no. 61772057, project no. 6141B06220202 from CAST and the support funding from State Key Lab. of Software Development Environment and Jiangxi and Qingdao Research Institute of Beihang University.

Appendices

A. Effectiveness on different backbones

We evaluate the effectiveness of our adaptive unimodal cost volume filtering scheme among different backbones, namely, the stack-hourglass version of PSMNet (Chang and Chen 2018) and GC-Net (Kendall et al. 2017). We re-implement all methods with the training protocol detailed in **Implementation details**. Specifically, the batch size of GC-Net is set to 24 for training on 8 Tesla V100. Table 5 reports the results, our method delivers better performance across different backbones.

Table 5: Evaluation our method among different stereo matching models, where * denotes equipping the model with our adaptive unimodal cost volume filtering scheme.

Method	Scene Flow					
	EPE [px]			3PE [%]		
	ALL	OCC	NOC	ALL	OCC	NOC
GC-Net	0.871	2.916	0.452	**3.89**	**15.63**	**1.65**
GC-Net*	**0.822**	**2.777**	**0.436**	4.33	16.46	2.02
PSMNet	1.101	3.507	0.637	4.56	17.64	**2.12**
PSMNet*	**0.867**	**2.736**	**0.495**	**4.31**	**15.77**	2.13

B. Architecture details

Table 6 presents the details of the AcfNet which is used in experiments to produce state-of-the-art accuracy on Scene Flow dataset (Mayer et al. 2016) and KITTI benchmarks (Geiger, Lenz, and Urtasun 2012; Menze and Geiger 2015). It is based on PSMNet with stacked hourglass architecture, which produces three cost volumes, and Confidence Estimation network(CENet) is added to each of the cost volume.

References

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

Cheng, X.; Wang, P.; and Yang, R. 2018. Learning depth with convolutional spatial propagation network. *arXiv preprint arXiv:1810.02695*.

Fu, Z., and Fard, M. A. 2018. Learning confidence measures by multi-modal convolutional neural networks. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 1321–1330. IEEE.

Table 6: Parameters of the network architecture of AcfNet.

Name	Layer setting	Output dimension
Feature Extraction		
input		$H \times W \times 3$
conv0_x	$[3\times3, 32] \times 3$	$\frac{1}{2}H \times \frac{1}{2}W \times 32$
conv1_x	$\begin{bmatrix}3\times3,32\\3\times3,32\end{bmatrix}\times 3$	$\frac{1}{2}H \times \frac{1}{2}W \times 32$
conv2_x	$\begin{bmatrix}3\times3,64\\3\times3,64\end{bmatrix}\times 16$	$\frac{1}{4}H \times \frac{1}{4}W \times 64$
conv3_x	$\begin{bmatrix}3\times3,128\\3\times3,128\end{bmatrix}\times 3, dila=2$	$\frac{1}{4}H \times \frac{1}{4}W \times 128$
conv4_x	$\begin{bmatrix}3\times3,128\\3\times3,128\end{bmatrix}\times 3, dila=4$	$\frac{1}{4}H \times \frac{1}{4}W \times 128$
branch_1	64×64, avg.pool $3\times3, 32$ bilinear interpolation	$\frac{1}{4}H \times \frac{1}{4}W \times 32$
branch_2	32×32, avg.pool $3\times3, 32$ bilinear interpolation	$\frac{1}{4}H \times \frac{1}{4}W \times 32$
branch_3	16×16, avg.pool $3\times3, 32$ bilinear interpolation	$\frac{1}{4}H \times \frac{1}{4}W \times 32$
branch_4	8×8, avg.pool $3\times3, 32$ bilinear interpolation	$\frac{1}{4}H \times \frac{1}{4}W \times 32$
concat	[conv2_16, conv4_3, branch_1, branch_2, branch_3, branch_4]	$\frac{1}{4}H \times \frac{1}{4}W \times 320$
fusion	$3\times3, 128$ $1\times1, 32$	$\frac{1}{4}H \times \frac{1}{4}W \times 32$
Cost Volume		
Concat left and shifted right		$\frac{1}{4}D \times \frac{1}{4}H \times \frac{1}{4}W \times 64$
Cost Aggregation		
3Dconv0_x	$\begin{bmatrix}3\times3\times3,32\\3\times3\times3,32\end{bmatrix}\times 2$	$\frac{1}{4}D \times \frac{1}{4}H \times \frac{1}{4}W \times 32$
3Dstack1_1	$\begin{bmatrix}3\times3\times3,64\\3\times3\times3,64\end{bmatrix}$	$\frac{1}{8}D \times \frac{1}{8}H \times \frac{1}{8}W \times 64$
3Dstack1_2	$\begin{bmatrix}3\times3\times3,64\\3\times3\times3,64\end{bmatrix}$	$\frac{1}{16}D \times \frac{1}{16}H \times \frac{1}{16}W \times 64$
3Dstack1_3	deconv $3\times3\times3, 64$ add **3Dstack1_1**	$\frac{1}{8}D \times \frac{1}{8}H \times \frac{1}{8}W \times 64$
3Dstack1_4	deconv $3\times3\times3, 32$ add **3Dconv0_2**	$\frac{1}{4}D \times \frac{1}{4}H \times \frac{1}{4}W \times 32$
3Dstack2_1	[$3\times3\times3, 64$ $3\times3\times3, 64$ add **3Dstack1_3**]	$\frac{1}{8}D \times \frac{1}{8}H \times \frac{1}{8}W \times 64$
3Dstack2_2	$\begin{bmatrix}3\times3\times3,64\\3\times3\times3,64\end{bmatrix}$	$\frac{1}{16}D \times \frac{1}{16}H \times \frac{1}{16}W \times 64$
3Dstack2_3	deconv $3\times3\times3, 64$ add **3Dstack1_1**	$\frac{1}{8}D \times \frac{1}{8}H \times \frac{1}{8}W \times 64$
3Dstack2_4	deconv $3\times3\times3, 32$ add **3Dconv0_2**	$\frac{1}{4}D \times \frac{1}{4}H \times \frac{1}{4}W \times 32$
3Dstack3_1	[$3\times3\times3, 64$ $3\times3\times3, 64$ add **3Dstack2_3**]	$\frac{1}{8}D \times \frac{1}{8}H \times \frac{1}{8}W \times 64$
3Dstack3_2	$\begin{bmatrix}3\times3\times3,64\\3\times3\times3,64\end{bmatrix}$	$\frac{1}{16}D \times \frac{1}{16}H \times \frac{1}{16}W \times 64$
3Dstack3_3	deconv $3\times3\times3, 64$ add **3Dstack1_1**	$\frac{1}{8}D \times \frac{1}{8}H \times \frac{1}{8}W \times 64$
3Dstack3_4	deconv $3\times3\times3, 32$ add **3Dconv0_2**	$\frac{1}{4}D \times \frac{1}{4}H \times \frac{1}{4}W \times 32$
output_1	$3\times3\times3, 32$ $3\times3\times3, 1$	$\frac{1}{4}D \times \frac{1}{4}H \times \frac{1}{4}W \times 1$
output_2	$3\times3\times3, 32$ $3\times3\times3, 1$ add **output_1**	$\frac{1}{4}D \times \frac{1}{4}H \times \frac{1}{4}W \times 1$
output_3	$3\times3\times3, 32$ $3\times3\times3, 1$ add **output_2**	$\frac{1}{4}D \times \frac{1}{4}H \times \frac{1}{4}W \times 1$
For each output in [output_1, output_2, output_3]		
upsampling	deconv $8\times8\times8$, stride=4	$D \times H \times W$
CENet	[$3\times3, 48$ $1\times1, 1$ sigmoid]	$H \times W$
disparity regression		$H \times W$

Geiger, A.; Lenz, P.; and Urtasun, R. 2012. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 3354–3361. IEEE.

Guo, X.; Yang, K.; Yang, W.; Wang, X.; and Li, H. 2019. Groupwise correlation stereo network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3273–3282.

Ilg, E.; Cicek, O.; Galesso, S.; Klein, A.; Makansi, O.; Hutter, F.; and Brox, T. 2018. Uncertainty estimates and multi-hypotheses networks for optical flow. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 652–667.

Kendall, A., and Gal, Y. 2017. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems*, 5574–5584.

Kendall, A.; Martirosyan, H.; Dasgupta, S.; Henry, P.; Kennedy, R.; Bachrach, A.; and Bry, A. 2017. End-to-end learning of geometry and context for deep stereo regression. In *Proceedings of the IEEE International Conference on Computer Vision*, 66–75.

Kim, S.; Min, D.; Kim, S.; and Sohn, K. 2018. Unified confidence estimation networks for robust stereo matching. *IEEE Transactions on Image Processing* 28(3):1299–1313.

Liang, Z.; Feng, Y.; Guo, Y.; Liu, H.; Chen, W.; Qiao, L.; Zhou, L.; and Zhang, J. 2018. Learning for disparity estimation through feature constancy. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2811–2820.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, 2980–2988.

Luo, C.; Yu, L.; Yang, E.; Zhou, H.; and Ren, P. 2019. A benchmark image dataset for industrial tools. *Pattern Recognition Letters* 125:341–348.

Luo, W.; Schwing, A. G.; and Urtasun, R. 2016. Efficient deep learning for stereo matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5695–5703.

Luo, C.; Yu, L.; and Ren, P. 2017. A vision-aided approach to perching a bioinspired unmanned aerial vehicle. *IEEE Transactions on Industrial Electronics* 65(5):3976–3984.

Mayer, N.; Ilg, E.; Hausser, P.; Fischer, P.; Cremers, D.; Dosovitskiy, A.; and Brox, T. 2016. 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*, 4040–4048.

Menze, M., and Geiger, A. 2015. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3061–3070.

Pang, J.; Sun, W.; Ren, J. S.; Yang, C.; and Yan, Q. 2017. Cascade residual learning: A two-stage convolutional neural network for stereo matching. In *Proceedings of the IEEE International Conference on Computer Vision*, 887–895.

Park, M.-G., and Yoon, K.-J. 2018. Learning and selecting confidence measures for robust stereo matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41(6):1397–1411.

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic differentiation in pytorch.

Poggi, M.; Pallotti, D.; Tosi, F.; and Mattoccia, S. 2019. Guided stereo matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 979–988.

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

Schmid, K.; Tomic, T.; Ruess, F.; Hirschmüller, H.; and Suppa, M. 2013. Stereo vision based indoor/outdoor navigation for flying robots. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 3955–3962. IEEE.

Sivaraman, S., and Trivedi, M. M. 2013. A review of recent developments in vision-based vehicle detection. In *2013 IEEE Intelligent Vehicles Symposium (IV)*, 310–315. IEEE.

Song, X.; Zhao, X.; Hu, H.; and Fang, L. 2018. Edgestereo: A context integrated residual pyramid network for stereo matching. In *Asian Conference on Computer Vision*, 20–35. Springer.

Tulyakov, S.; Ivanov, A.; and Fleuret, F. 2018. Practical deep stereo (pds): Toward applications-friendly deep stereo matching. In *Advances in Neural Information Processing Systems*, 5871–5881.

Yang, G.; Zhao, H.; Shi, J.; Deng, Z.; and Jia, J. 2018. Segstereo: Exploiting semantic information for disparity estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 636–651.

Yin, Z.; Darrell, T.; and Yu, F. 2019. Hierarchical discrete distribution decomposition for match density estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 6044–6053.

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

Zenati, N., and Zerhouni, N. 2007. Dense stereo matching with application to augmented reality. In *2007 IEEE International Conference on Signal Processing and Communications*, 1503–1506. IEEE.

Zhang, F.; Prisacariu, V.; Yang, R.; and Torr, P. H. 2019. Ga-net: Guided aggregation net for end-to-end stereo matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 185–194.