MABNet: A Lightweight Stereo Network Based on Multibranch Adjustable Bottleneck Module

Jiabin Xing, Zhi Qi, Jiying Dong, Jiaxuan Cai, and Hao Liu

National ASIC System Engineering Research Center, Southeast University, China
{xingjumping,101011256,213140498,213161230,nicky_lh}@seu.edu.cn

Abstract. Recently, end-to-end CNNs have presented remarkable performance for disparity estimation. But most of them are too heavy to resource-constrained devices, because of enormous parameters necessary for satisfactory results. To address the issue, we propose two compact stereo networks, MABNet and its light version MABNet_tiny. MABNet is based on a novel Multibranch Adjustable Bottleneck (MAB) module, which is less demanding on parameters and computation. In a MAB module, feature map is split into various parallel branches, where the depthwise separable convolutions with different dilation rates extract features with multiple receptive fields however at an affordable computational budget. Besides, the number of channels in each branch is adjustable independently to tradeoff computation and accuracy. On SceneFlow and KITTI datasets, our MABNet achieves competitive accuracy with fewer parameters of 1.65M. Especially, MABNet_tiny reduces the parameters to 47K by cutting down the channels and layers in MABNet.

Keywords: Stereo Matching, Disparity Estimation, Multibranch Adjustable Bottleneck module, Compact Networks

1 Introduction

Disparity estimation from a stereo pair of images provides depth information which is a significant cue for many computer vision applications, such as autonomous driving [19], 3D reconstruction [32] and augmented reality [1]. These applications usually run on mobile devices or embedded platforms, including drones [20], smart phones and vehicles. These resource-constrained devices prefer the stereo system with low power consumption and small memory footprint. Besides, stereo system has to be of low latency and high accuracy to ensure the safety and the comfort, especially in autonomous driving. However, in order to achieve high accuracy, we have to design complex model with a large number of parameters and floating-point-operations (FLOPs), which conflicts the energy efficiency required by resource-constrained devices. In this paper, we propose two lightweight end-to-end stereo networks to tradeoff computation and accuracy, namely MABNet and its light version MABNet_tiny. They have fewer parameters and FLOPs thus are more suitable to be deployed on embedded devices.

In general, traditional stereo matching pipeline consists of four steps: matching cost calculation, cost aggregation, disparity computation and disparity refinement [12]. It computes the matching cost within a finite window, with the limitation of dealing with the large texture-less areas, occlusions and repeating textures. The accuracy and speed of traditional stereo matching methods are still unable to meet the actual application requirements.

With the rapid development of deep convolutional neural networks (CNNs), people proposed many learning-based stereo methods to overcome the limitation of traditional methods. MC-CNN [44] first introduced CNNs in stereo to calculate the matching cost by comparing image patches, and proved the great potential of CNNs. Gradually, it replaced some of the aforementioned steps of the traditional stereo pipeline. CCNN [28] and PBCP [31] estimate confidence by CNNs, while LRCR [16] and RecResNet [3] train CNNs to refine the disparity. Learning-based stereo methods improve the accuracy but have to take more time to process.

Inspired by the successes of end-to-end neural networks in optical flow computation [8], object detection and semantic segmentation [2, 5], CNNs have replaced the total traditional stereo matching pipeline. The first end-to-end stereo network is DispNet [24] proposed in 2016. DispNet achieves competitive accuracy with MC-CNN [44] on KITTI dataset [10, 26] but runs 100× faster on GPU. It utilizes encoder-decoder architecture which extracts unary features from a stereo pair of images by 2D CNNs, correlates the features and then restores the original resolution by consecutive deconvolutions. CRL [27], iResNet [21], MADNet [34] encode similarity into feature channels by this feature correlation method. However, their results loss the real geometric context and have to improve accuracy at the expense of more parameters in filters. Instead of simply correlating features, some networks, such as GC-Net [17], PSMNet [4], GA-Net [45] and AMNet [9], correlate features at different disparity levels to build a 4D cost volume and aggregate cost by 3D CNNs. They have fewer parameters but take a longer time because of more operations in 3D CNNs.

Although end-to-end CNNs show superior performance in stereo, it is challenging to deploy end-to-end stereo networks on practical devices with limited resource due to their enormous parameters and excessive FLOPs. People pay too much attention to the high accuracy, constructing more complex networks. For example, in comparison with GC-Net [17], GA-Net-deep [45] reduced three-pixel-error (3PE) from 2.87% to 1.81% on KITTI2015 [26] but doubling the number of parameters and runtime. In contrast to previous works, we focus on the model size and feasibility of implementation on hardware and manage to build as compact as possible stereo networks on the precondition of guaranteeing precision.

We propose a lightweight bottleneck module constructed by depthwise separable convolutions [7] with fewer parameters and FLOPs than standard convolutions. In addition, in order to compensate the accuracy loss, it incorporates standard convolution and dilated convolution [13] with different dilation rates by split-transform-merge strategy [41], and uses channel shuffle operation [46] to

promote the information communication between different groups. We name the bottleneck module as Multibranch Adjustable Bottleneck (MAB) module since it has several branches with different dilation rates and adjustable scale factors. As for 3D MAB module, we factorize a standard 3D convolution into disparity-wise convolution and spatial convolution to further reduce parameters and FLOPs. Details of our 2D and 3D MAB modules are described in Section 3.1. Based on the 2D and 3D MAB modules, we design our compact stereo networks MABNet and MABNet_tiny. MABNet with 1.65M parameters achieves 2.41% three-pixel-error (3PE) on KITTI2015, while MABNet_tiny with 47K parameters achieves 3.88% 3PE.

2 Related Work

There have been several concurrent works pushing towards learning-based stereo in different directions, such as high accuracy, low latency and strong self-adaption. In this paper, we are concerned with lightweight end-to-end stereo networks, which is prone to be applied on embedded devices instead of GPUs. Some works optimize the GPU to process neural network more efficiently and faster, or improve the neural network to adapt the operation mode of GPU. They speed up the stereo networks on GPU, but have fewer substantive benefit on the high energy efficient implementation on embedded devices than reducing model size.

Different from them, StereoDRNet [33] devoted to reducing FLOPs. Based on PSMNet [4], it improved the feature extraction module by vortex pooling [40], and proposed a novel cost filtering network with fewer FLOPs to aggregate cost. PDSNet [35], an applications-friendly deep stereo, designed a novel bottleneck module, drastically reducing the memory footprint in inference. It also proposed sub-pixel cross-entropy loss combined with a MAP estimator, making the system applicable to different disparity ranges without retraining. Besides, LWSM [43] utilized group convolution and dilated convolution to build upon an enhancement block which has low computation complexity and memory consumption. Their accuracy is competitive with classical end-to-end stereo networks, but their models are smaller obviously.

Furthermore, there are more compact models, such as StereoNet [18] and AnyNet [38]. StereoNet [18] achieved real-time performance by using a very low-resolution cost volume, reducing the parameters by an order of magnitude in comparison with PDSNet [35] or LWSM [43]. AnyNet [38] is a tiny stereo network with only 40K parameters by the aid of U-Net [29] and SPNet [22]. It can process 1242×375 resolution images within a range of 10-35 FPS on an NVIDIA Jetson TX2 module. However, due to excessive compression, both of StereoNet and AnyNet's accuracy drop severely.

In comparison with prior models on the approximate order of magnitude, our MABNet and MABNet_tiny achieve a noticeable improvement in accuracy. Moreover, our models with fewer parameters and FLOPs are easier to be deployed on embedded devices. The detailed experimental data is given in the Section 4.3.

3 Proposed Network

As the foundation of our networks, Multibranch Adjustable Bottleneck (MAB) module is introduced firstly, including the structure and the advantage. Then we provide an overview. In each part of its introduction, we describe the difference between MABNet_tiny and the origin MABNet.

3.1 Multibranch Adjustable Bottleneck (MAB) module

Most prior works, such as PSMNet [4], StereoDRNet [33] and GA-Net [45], utilized ResNet block [11] (see Fig. 1) to design their feature extraction backbone, leading to oversized models. Recently, there have been more compact and more accurate networks, such as MobileNet [14, 30], SqueezeNet [15] and ShuffleNet [46, 23]. They are used or referenced in different fields, showing great performance. Besides for classification, FastDepth [39] adopted MobileNet [14, 30] to design an encoder-decoder architecture for monocular depth estimation, while ESPNet [25] used techniques in compact CNNs to build a network for semantic segmentation. Driven by the successful experience, we designed Multibranch Adjustable Bottleneck (MAB) module, as shown in Fig. 2 and Fig. 3.

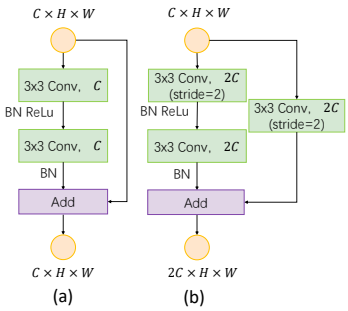

Fig. 1. ResNet block in PSMNet [4] does not apply ReLU after summation. (a)ResNet block with *stride*=1. (b)ResNet block with *stride*=2.

We adopt split-transform-merge strategy [33] to design our MAB module. Firstly, a 2D MAB module (see Fig. 2(a)) equally splits the input into two subblocks by channel split operation [23]. Then, the first subblock is fed into three parallel branches to generate features. The corresponding scale factors λ_i in Fig. 2 and Fig. 3 controls the number of channels for input feature maps in these branches separately. Generally, the first layer projects a high dimension feature map onto a low dimension space via a pointwise convolution. Then a depthwise convolution [6] with a certain dilation rate extracts features. The three branches with dilation rates of {1,2,4} grab multilevel context information. Next, we combine the outputs of three branches along channel dimension, followed by another pointwise convolution to merge them. Finally, we concatenate it with

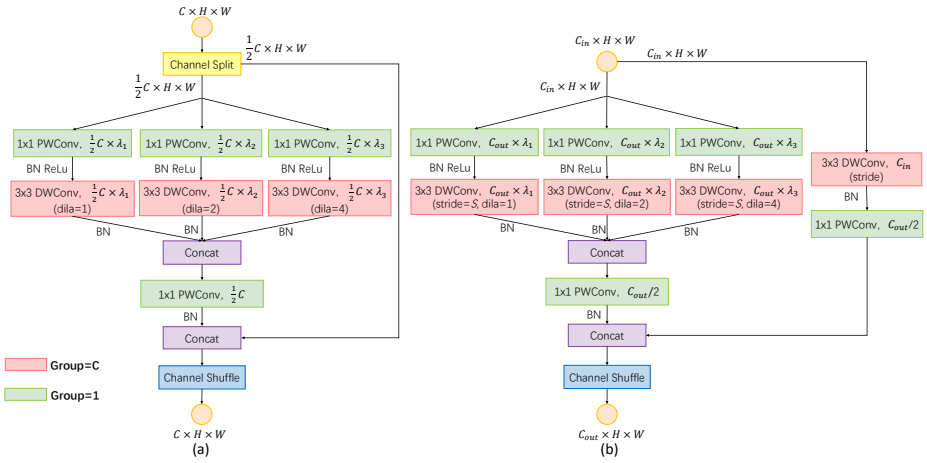

Fig. 2. (a)2D MAB module with $size_{in}$=$size_{out}$ ($stride$=1 and $C_{in} = C_{out} = C$). (b)2D MAB module with $size_{in}$≠$size_{out}$ (($stride = S$) $\neq 1$ or $C_{in} \neq C_{out}$). PWConv: pointwise convolution. DWConv: depthwise convolution. λ_i={λ_1,λ_2 λ_3}.

another subblock of input, then perform channel shuffle operation proposed in ShuffleNet [46, 23] to make the input and output channels fully related.

Although dilated convolution used in our MAB enlarges the effective receptive field without increasing the number of parameters, it may cause gridding artifact [37] sacrificing the accuracy. Fortunately, our multibranch structure not only contains context information of multiple receptive fields but also diminishes gridding artifact through fusing output features of the three branches. Furthermore, considering that larger dilation rate leads to more paddings (filling in zeros) which fades the effective information in feature maps and increases computational cost, we finally choose the dilation rate as {1,2,4} in 2D MAB module. In Section 4.2, we prove the choice by ablation studies.

In addition, features with different receptive fields benefit the depth estimation of different scenes. Usually, large dilation rate learns coarse-grained relationship, such as houses, cars and roads, helpful for disparity estimation in background. On the contrary, small dilation rate needs more convolutions to get the same receptive field, and learns more fine-grained information, like windows, wheels and traffic lights. In Section 4.2, we research the best proportional relationship between the three scale factors and a reasonable receptive field in outputs.

Unlike the Fig. 2(a), the subfigure of Fig. 2(b) exhibits another 2D MAB module when $stride \neq 1$ or $C_{in} \neq C_{out}$. In the (b) module used to downsample, we skip the channel split operation, but concatenate the input as a residual to the result through a shortcut connection.

Besides the 2D MAB, we also implement two kinds of computational efficient 3D MAB modules, as shown in Fig. 3. Inspired by the spatial and temporal convolutions in 3D CNNs [42, 36] for video recognition, we factorize a standard 3D

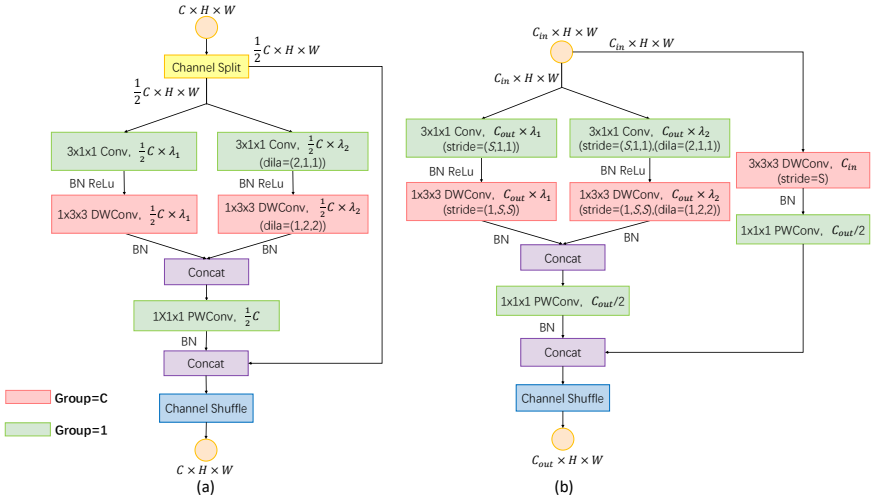

Fig. 3. (a)3D MAB module with $size_{in}$=$size_{out}$ ($stride$=1 and $C_{in} = C_{out} = C$). (b)3D MAB module with $size_{in} \neq size_{out}$ (($stride = S$) $\neq 1$ or $C_{in} \neq C_{out}$). PWConv: pointwise convolution. DWConv: depthwise convolution. λ_i=$\{\lambda_1,\lambda_2\}$.

convolution into two stages, namely disparity-wise convolution and spatial convolution. The disparity-wise convolution plays the same role as the first pointwise convolution in 2D MAB module. Meanwhile, the resolution of 3D MAB in cost aggregation is smaller than that of 2D MAB in feature extraction, asking 3D MAB for fewer paddings. Therefore, 3D MAB deletes the branches with dilation rate=4 and remains only two branches, which also reduce computation.

3.2 MABNet Overview

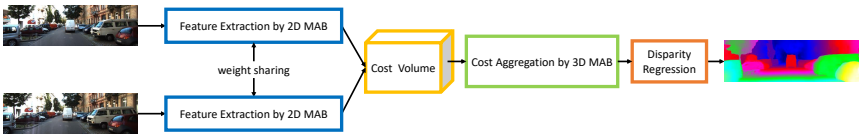

Fig. 4. Architecture overview of MABNet and MABNet_tiny.

As shown in Fig. 4, our stereo matching pipeline consists of four classical steps. Firstly, the stereo pair of images each with the size of $3 \times H \times W$ are fed to two weight-sharing feature extractors respectively. The resolution of each output feature map is reduced to a quarter of the original input image as $C \times \frac{1}{4}H \times \frac{1}{4}W$. Then we correlate the two feature maps at different disparity levels to build a 4D cost volume of $2C \times \frac{1}{4}D \times \frac{1}{4}H \times \frac{1}{4}W$. Next, the cost volume is aggregated by 3D

MAB modules followed by a bilinear interpolation to upsample the cost volume back to the resolution of $1\times D\times H\times W$. Finally, we apply a regression procedure in D dimension to obtain the disparity map of the same resolution as the input images. Note that C, D, H and W denote the number of channels, the maximum disparity, the height and the weight of the input image respectively in the paper.

3.3 Feature Extraction by 2D MAB

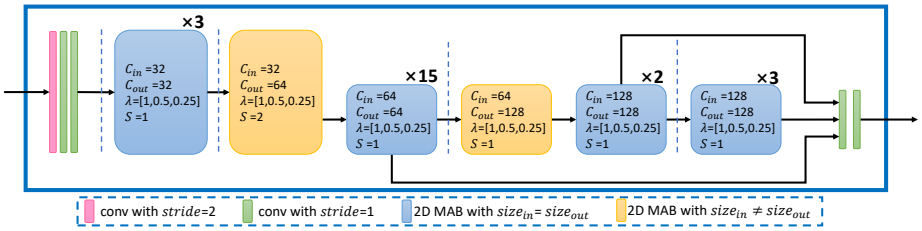

Fig. 5. Feature extraction in MABNet. ×2, ×3 and ×15 represent the number of repetitions. λ=[λ_1,λ_2,λ_3]. The height of the rectangle is proportional to the resolution of the output feature maps.

The schematic diagram of the feature extraction in MABNet and the parameters for each 2D MAB are presented in Fig. 5. We first use three cascaded 3×3 convolution filters, where the first filter has a stride of 2, downsampling the input image. Next, four groups of 2D MAB modules extract further features. The number of 2D MAB modules in the four groups are {3,16,3,3} individually, generating the output feature maps of {32,64,128,128} channels respectively. The outputs of the last three groups are concatenated to form a unary feature map of 320 channels. Finally, through two convolution layers, we fuse the 320-channel feature map to a cost volume of 32 channels then output it. Instead of using four groups of 2D MAB modules, MABNet_tiny uses only three groups of {4,8,4} modules corresponding {8,16,32} channels, and gives a cost volume of 8 channels by fusing the outputs of the last two groups.

3.4 Cost Volume

After the feature extraction, we get the left and right feature maps, both with the size of $C\times\frac{1}{4}H\times\frac{1}{4}W$. As illustrated in Fig. 6, following GC-Net [17], we form a 4D cost volume of $2C\times\frac{1}{4}D\times\frac{1}{4}H\times\frac{1}{4}W$ by concatenating left feature map with their corresponding right feature map at different disparity levels, rather than concatenating a bulk of right feature maps at the end of the entire group of left feature maps. Specifically, when a $\frac{1}{4}H\times\frac{1}{4}W$ feature map of left feature builds a $\frac{1}{4}D\times\frac{1}{4}H\times\frac{1}{4}W$ cost, the data stays at the original position. But the corresponding right feature map shifts to the right sequentially with the necessary trimming and padding.

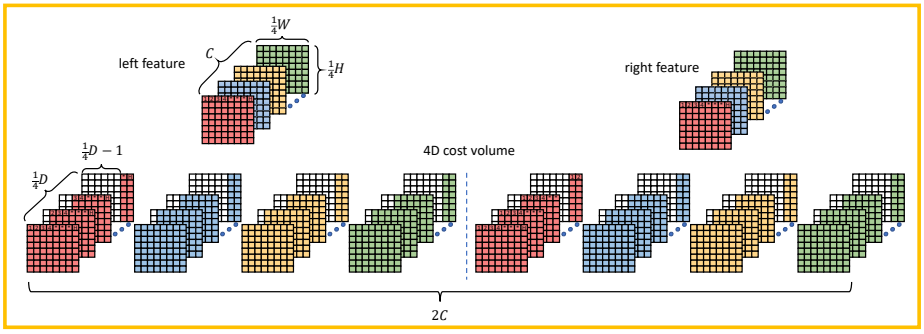

Fig. 6. Illustration of cost volume building. The colored parts represent data in feature maps, while the white parts in the illustration of 4D cost volume represent data filled with 0.

3.5 Cost Aggregation by 3D MAB

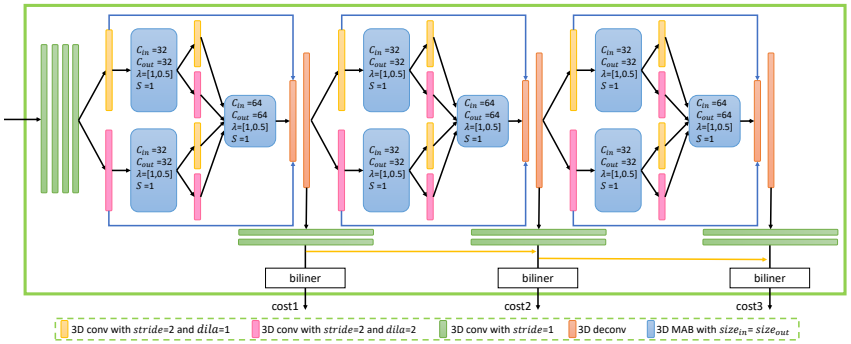

Fig. 7. Cost aggregation in MABNet.

At cost aggregation stage, instead of stacking standard hourglass (encoder-decoder) architecture proposed in PSMNet [4], we design a novel multibranch hourglass, a bit similar to split-transform-merge structure. MABNet takes the advantage of the three successive hourglass networks, as shown in Fig. 7. In each hourglass network, a combination of four parallel branches and a 3D MAB module encodes information with different receptive fields. The encoded information is upsampled back to the same size as the input image by two 3D deconvolutions and one bilinear interpolation. The two outputs in the first layer of each hourglass network are concatenated and added to the output of first 3D deconvolution through a short path to compensate the loss of features during the encoding procedure, as linked by the blue arrows in Fig. 7. In MABNet, the three successive hourglass networks generate three aggregated costs and three

training losses ($Loss_1$, $Loss_2$, and $Loss_3$) correspondingly. The loss function is described in Section 3.7. As for the simplification of MABNet_tiny, we keep only one hourglass network in cost aggregation.

3.6 Disparity Regression

We employ the disparity regression proposed in GC-Net [17] to estimate the continuous disparity map:

$$\hat{d} = \sum_{d=0}^{D_{max}} d \times \sigma\left(-c_d\right) \tag{1}$$

where the estimation disparity $\hat{d}$ denotes the sum of each disparity d weighted by its probability. And the probability is calculated from the cost volume $-c_d$ via the softmax operation $\sigma\left(\cdot\right)$.

3.7 Training Loss

During training, we adopt smooth L1 loss to measure the difference between the output of MABNet and the ground truth. L1 loss is robust but nondifferentiable at disparity discontinuities, while L2 loss is differentiable everywhere but too sensitivity to outliers. Hence, we formulate smooth L1 loss, defined as

$$L\left(d, \hat{d}\right) = \frac{1}{N} \sum_{i=1}^{N} smooth_{L_1}\left(d_i - \hat{d}_i\right) \tag{2}$$

in which

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

where N is the total number of labeled pixels, $\hat{d}_i$ is the predicted disparity and d_i is the ground-truth disparity. Similar to PSMNet [4], the total training loss is calculated as the weighted summation of the three losses. The total training loss L_{total} in MABNet is defined as

$$L_{total} = 0.5 \times Loss_1 + 0.7 \times Loss_2 + Loss_3. \tag{4}$$

Because MABNet_tiny uses just one hourglass network, its L_{total} equals to $Loss_1$.

4 Experiments

In this section, we evaluate our MABNet on SceneFlow, KITTI2012 and KITTI2015 stereo datasets. We first introduce the datasets and the experiment settings. Then we present ablation studies to compare different models with different parameter configurations. Finally, we compare the proposed stereo networks with other state-of-the-art published methods.

4.1 Implementation Details

Datasets SceneFlow [24] is a large-scale synthetic dataset with 35454 stereo pairs for training and 4370 stereo pairs for testing, all being of 540×960 resolution. It provides dense disparity maps as ground truth. We use the end-point error (EPE) as the evaluation metric, which means the average absolute disparity error in pixels.

Unlike SceneFlow, KITTI is a real-world dataset with street views from a driving car, consisting of KITTI2012 [10] and KITTI2015 [26]. KITTI2012 contains 194 training stereo pairs with semi-dense ground truth disparities acquired using a LIDAR sensor and 195 testing stereo pairs without ground truth disparities, both of which are of 376×1240 resolution. KITTI2015 contains 200 training stereo pairs and 200 testing stereo pairs. Since there is no ground truth disparity in testing set, we divide the whole training data into a training set (80%) and a validation set (20%) in our ablation studies. Finally, we submit the results to the KITTI online benchmark to evaluate our models. Note that KITTI consider a pixel to be correctly estimated if the disparity end-point error is <3 pixels or <5%. The percentage of erroneous pixels (3PE) is the evaluation metric for KITTI dataset.

Experiment settings We implement our stereo networks in PyTorch and train them by Adam optimizer with β_1=0.9 and β_2=0.999 on four Nvidia RTX 2080Ti. During training, we crop the input image to size 256×512 randomly and perform color normalization to process all of them. Besides, we fix the batch size to 8 and set the maximum disparity D_{max} to 192.

4.2 Ablation Studies

In the proposed MAB module, the number of branches, scale factors are adjustable. Through the following ablation studies, we want to figure out the impact of the parameter configuration to the accuracy of MABNet. Basically, a larger number of branches and higher scale factors lead to more expensive computational demand while not necessarily a better accuracy of the stereo network. We performed two sets of experiments in MABNet to decide the proper parameter configuration in 2D and 3D MAB modules. Each model was evaluated on SceneFlow and KITTI2015. For SceneFlow dataset, we trained the model from scratch for 10 epochs with learning rate=0.001. For KITTI2015, there are only 160 training stereo pairs and 40 validation stereo pairs, making models susceptible to over-fitting. Thus, we used the pretrained SceneFlow model and finetuned it for 300 epochs. The learning rate of this fine-tuning began at 0.001 for first 200 epochs, then drops to 0.0001 for remaining 100 epochs. Finally, we computed EPE on the SceneFlow test set and the percentage of 3PE on the KITTI 2015 validation set.

Different numbers of branches in 2D MAB module To figure out the best choice for the number of branches in 2D MAB module, we did four experiments in

MABNet with the fixed scale factors as 0.5 both in 2D and 3D MAB modules. The reason we only did experiments with 2D MAB is because the number of branches in 3D MAB can be inferred from the results about 2D MAB. And adding a branch in 3D MAB will greatly increase FLOPs, which is time and resources consuming. In the experiments, λ_1, λ_2 and λ_3 are the scale factor in the branch with $dila$=1, $dila$=2 and $dila$=4 respectively. Besides, we added the $4th$ branch with $dila$=8 and λ_4.

Table 1. Evaluation of MABNet with different numbers of branches in 2D MAB module. λ_1, λ_2, λ_3 and λ_4 are scale factors in 2D MAB. FLOPs represent floating-point-operations in processing a stereo pair of 256×512, including convolution, activation function and batch normalization.

Branches	λ_1	λ_2	λ_3	λ_4	Parameters	FLOPs	SceneFlow(EPE)	KITTI2015(3PE)
1	0.5	-	-	-	$1.525M$	$188.26G$	1.072	3.236%
2	0.5	0.5	-	-	$1.573M$	$189.16G$	1.111	3.349%
3	0.5	0.5	0.5	-	$1.621M$	$190.07G$	**1.056**	**3.174%**
4	0.5	0.5	0.5	0.5	$1.669M$	$190.97G$	1.588	3.225%

The results listed in Table 1 show that setting three branches in 2D MAB is the most reasonable case. As for experiments with one and two branches, they do not extract enough multilevel features, causing lower accuracy than that with three branches. As for the experiment with four branches, although it has features with more level at the cost of more parameters and FLOPs, its result is comparatively unacceptable especially when tested on Sceneflow [24] which contains lots of pictures of monkey and flying objects. As mentioned in Section 3.1, larger dilation rate in the additional branch leads to more paddings, fading the effective information in feature maps. On the other hand, larger dilation rate good for the estimation of background disparity contains scanty foreground information. Therefore, the experiment with four branches yields a worse depth estimation.

Different scale factors in MAB module After determining the number of branches, we further explored the proportional relationship between the three scale factors. Similar to 2D MAB, λ_1, λ_2 in 3D MAB are the scale factors in the branches with $dila$=1 and $dila$=2 respectively. To narrow the search space, we constrained the λ_i to be $\frac{1}{2^n}$ (n=0,1,2), which ensured that the number of output channels was an integer and greater than 1. Besides, we only performed experiments on SceneFlow since KITTI validation dataset has too few samples to get regular results.

We first carry out the experiments with fixed λ in 3D MAB module as 0.5 and 0.25. The results about 2D MAB modules in Table 2 proves that λ_1 had positive

impact on the accuracy but λ_3 had opposite effect. Therefore, we empirically fixed the scale factors of 2D MAB in the next three experiments in Table 3, trying to explore the importance of λ_i in 3D MAB. We found that the configuration of {1,0.5,0.25,0.5,0.25} gives the best result.

Table 2. Evaluation of MABNet with different scale factors in 2D MAB module.

2D MAB			3D MAB		Parameters	SceneFlow (EPE)
λ_1	λ_2	λ_3	λ_1	λ_2		
0.25	0.5	0.5	0.5	0.25	$1.591M$	1.107
0.5	0.5	0.5	0.5	0.25	$1.615M$	1.072
1	0.5	0.5	0.5	0.25	$1.663M$	1.041
0.5	**0.25**	0.5	0.5	0.25	$1.591M$	1.075
0.5	**0.5**	0.5	0.5	0.25	$1.615M$	1.072
0.5	**1**	0.5	0.5	0.25	$1.663M$	1.073
0.5	0.5	**0.25**	0.5	0.25	$1.591M$	1.091
0.5	0.5	**0.5**	0.5	0.25	$1.615M$	1.072
0.5	0.5	**1**	0.5	0.25	$1.663M$	1.161

Table 3. Evaluation of MABNet with different scale factors in 3D MAB modules.

2D MAB			3D MAB		Parameters	SceneFlow (EPE)
λ_1	λ_2	λ_3	λ_1	λ_2		
1	0.5	0.25	0.25	0.5	$1.639M$	1.127
1	0.5	0.25	0.5	0.25	$1.639M$	0.987
1	0.5	0.25	0.5	0.5	$1.645M$	1.006
1	0.5	0.25	1	0.5	$1.655M$	0.994

4.3 Evaluations on Benchmarks

We evaluated our MABNet and MABNet_tiny on SceneFlow and KITTI to prove the effectiveness of our MAB module. Unlike ablation studies, we increased the number of trainings to 20 epochs for SceneFlow and 600 epochs for KITTI. For SceneFlow, the learning rate was 0.001 initially, 0.0005 for $16th$-$18th$ epoch and 0.0001 for $19th$-$20th$. For KITTI, we used the pretrained SceneFlow model in $15th$ epoch. The learning rate was 0.001 initially, 0.0005 for $300th$-$399th$ epoch, 0.0002 for $400th$-$499th$ epoch and 0.0001 for $500th$-$600th$ epoch.

According to the online KITTI2015 leaderboard, as shown in Table 4, in comparison with other models on the approximate order of magnitude, our compact models with fewer parameters achieves competitive accuracy. Especially,

MABNet_tiny improves the accuracy significantly over StereoNet and AnyNet. We obtain the same observation through the experimental results on KITTI2012 and SceneFlow, illustrated in Table 5.

Table 4. Evaluation results on KITTI2015 benchmark. The percentages of erroneous pixels for non-occluded (Noc) and all (All) pixels in background (D1-bg), foreground (D1-fg) and all areas (D1-all) are reported. Note that AnyNet [38] is tested on the Nvidia Jetson TX2 GPU computing module.

Method	All(%)			Noc(%)			Parameters	Runtime
	D1-bg	D1-fg	D1-all	D1-bg	D1-fg	D1-all		
GC-Net [17]	2.21	6.16	2.87	2.02	5.58	2.61	$3.5M$	0.9s
PSMNet [4]	1.86	4.62	2.32	1.71	4.31	2.14	$5.2M$	0.41s
PDSNet [35]	2.29	4.05	2.58	2.09	3.68	2.36	$2.2M$	0.5s
LWSM [43]	1.86	5.35	2.44	1.69	4.68	2.18	$1.8M$	0.24s
MABNet	1.89	5.02	2.41	1.74	4.59	2.21	$1.65M$	0.38s
StereoNet [18]	4.30	7.45	4.83	-	-	-	$360K$	0.015s
AnyNet [38]	-	-	6.20	-	-	-	$40K$	0.0973s
MABNet_tiny	3.04	8.07	3.88	2.80	7.28	3.54	$47K$	0.11s

Table 5. Evaluation results on KITTI2012 and SceneFlow benchmark. The percentages of erroneous pixels (Out) and the average end-point errors (Avg) for both non-occluded (Noc) and all (All) pixels are reported on KITTI2012. The error threshold is set to 2.

Method	KITTI2012				SceneFlow
	Out-Noc(%)	Out-All(%)	Avg-Noc(px)	Avg-all(px)	(EPE)
GC-Net [17]	2.71	3.46	0.6	0.7	2.51
PSMNet [4]	2.44	3.01	0.5	0.6	1.09
PDSNet [35]	3.82	4.65	0.9	1.0	1.12
LWSM [43]	2.48	3.17	0.5	0.6	0.8
MABNet	2.43	3.02	0.5	0.5	0.797
StereoNet [18]	4.91	6.02	0.8	0.9	1.101
MABNet_tiny	4.45	5.27	0.7	0.8	1.663

As shown in Table 6, compared with PSMNet [4] and StereoNet [18], the proposed MABNet and MABNet_tiny that have much less network parameters do not exhibit the advantage of running time on the platform of GPU. We discover that the depthwise convolutions involved in the 2D and 3D MAB modules cannot perform efficiently on GPUs. Because the procedure of a convolution is similar to a general matrix multiplication (GEMM) operation, before which GPU must perform an im2col operation, transforming the input 3D data into

a 2D matrix. Therefore, a depthwise convolution needs to repeat the im2col and GEMM operation C times because it has C groups of feature map, while a standard convolution needs only once in2col operation. Since the number of parameters influences the memory access cost [46], as well as the number of FLOPs determines the number of multiply-and-accumulate (MAC) operations, we believe MABNet and its light version should achieve superior efficiency on the other hardware flatforms, like embedded DNN accelerators for edge devices.

Table 6. Evaluation results of FLOPs for processing a stereo pair of 256×512, including convolution, activate function and batch normalization.

Method	PSMNet [4]	MABNet	StereoNet [18]	MABNet_tiny
FLOPs	$257.01G$	$190.75G$	$14.08G$	$6.60G$
Runtime	$0.41s$	$0.38s$	$0.015s$	$0.11s$

5 Conclusions

We propose the MAB module which can extract features with multiple receptive fields and is adjustable in the number of channels in each branch. Moreover, we are looking forward to applying the MAB module to more fields, such as object detection, semantic segmentation and classification. Based on the MAB module, we propose two lightweight stereo network MABNet and its light version MABNet_tiny. Experimental results on SceneFlow and KITTI demonstrate the effectiveness of the MAB module, MABNet and MABNet_tiny. More importantly, our models with few parameters and low computational complexity are easy to be deployed on resource-constrained devices.

Acknowledgement

This research was supported by the Key Science and Technology Projects in Jiangsu Province (Grant No. BE2018002-2) and the National Nature Foundation of China (Grant No.61974024).

References

1. Alhaija, H.A., Mustikovela, S.K., Mescheder, L., Geiger, A., Rother, C.: Augmented reality meets computer vision: Efficient data generation for urban driving scenes. International Journal of Computer Vision **126**(9), 961–972 (2018)
2. Badrinarayanan, V., Kendall, A., Cipolla, R.: Segnet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE transactions on pattern analysis and machine intelligence **39**(12), 2481–2495 (2017)
3. Batsos, K., Mordohai, P.: Recresnet: A recurrent residual cnn architecture for disparity map enhancement. In: 2018 International Conference on 3D Vision (3DV). pp. 238–247. IEEE (2018)
4. Chang, J.R., Chen, Y.S.: Pyramid stereo matching network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5410–5418 (2018)
5. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Semantic image segmentation with deep convolutional nets and fully connected crfs. arXiv preprint arXiv:1412.7062 (2014)
6. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1251–1258 (2017)
7. Denton, E.L., Zaremba, W., Bruna, J., LeCun, Y., Fergus, R.: Exploiting linear structure within convolutional networks for efficient evaluation. In: Advances in neural information processing systems. pp. 1269–1277 (2014)
8. Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., Brox, T.: Flownet: Learning optical flow with convolutional networks. In: Proceedings of the IEEE international conference on computer vision. pp. 2758–2766 (2015)
9. Du, X., El-Khamy, M., Lee, J.: Amnet: Deep atrous multiscale stereo disparity estimation networks. arXiv preprint arXiv:1904.09099 (2019)
10. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. pp. 3354–3361. IEEE (2012)
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
12. Hirschmuller, H.: Stereo processing by semiglobal matching and mutual information. IEEE Transactions on pattern analysis and machine intelligence **30**(2), 328–341 (2007)
13. Holschneider, M., Kronland-Martinet, R., Morlet, J., Tchamitchian, P.: A real-time algorithm for signal analysis with the help of the wavelet transform. In: Wavelets, pp. 286–297. Springer (1990)
14. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Mobilenets, H.A.: Efficient convolutional neural networks for mobile vision applications. arXiv preprint ArXiv:1704.0486 (2017)
15. Iandola, F.N., Han, S., Moskewicz, M.W., Ashraf, K., Dally, W.J., Keutzer, K.: Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. arXiv preprint arXiv:1602.07360 (2016)
16. Jie, Z., Wang, P., Ling, Y., Zhao, B., Wei, Y., Feng, J., Liu, W.: Left-right comparative recurrent model for stereo matching. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3838–3846 (2018)

17. Kendall, A., Martirosyan, H., Dasgupta, S., Henry, P., Kennedy, R., Bachrach, A., Bry, A.: End-to-end learning of geometry and context for deep stereo regression. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 66–75 (2017)
18. Khamis, S., Fanello, S., Rhemann, C., Kowdle, A., Valentin, J., Izadi, S.: Stereonet: Guided hierarchical refinement for real-time edge-aware depth prediction. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 573–590 (2018)
19. Lee, K.J., Bong, K., Kim, C., Jang, J., Lee, K.R., Lee, J., Kim, G., Yoo, H.J.: A 502-gops and 0.984-mw dual-mode intelligent adas soc with real-time semiglobal matching and intention prediction for smart automotive black box system. IEEE Journal of Solid-State Circuits **52**(1), 139–150 (2016)
20. Li, Z., Dong, Q., Saligane, M., Kempke, B., Yang, S., Zhang, Z., Dreslinski, R., Sylvester, D., Blaauw, D., Kim, H.S.: 3.7 a 1920×1080 30fps 2.3 tops/w stereo-depth processor for robust autonomous navigation. In: 2017 IEEE International Solid-State Circuits Conference (ISSCC). pp. 62–63. IEEE (2017)
21. Liang, Z., Feng, Y., Guo, Y., Liu, H., Chen, W., Qiao, L., Zhou, L., Zhang, J.: Learning for disparity estimation through feature constancy. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2811–2820 (2018)
22. Liu, S., De Mello, S., Gu, J., Zhong, G., Yang, M.H., Kautz, J.: Learning affinity via spatial propagation networks. In: Advances in Neural Information Processing Systems. pp. 1520–1530 (2017)
23. Ma, N., Zhang, X., Zheng, H.T., Sun, J.: Shufflenet v2: Practical guidelines for efficient cnn architecture design. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 116–131 (2018)
24. Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T.: 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. pp. 4040–4048 (2016)
25. Mehta, S., Rastegari, M., Caspi, A., Shapiro, L., Hajishirzi, H.: Espnet: Efficient spatial pyramid of dilated convolutions for semantic segmentation. In: Proceedings of the european conference on computer vision (ECCV). pp. 552–568 (2018)
26. Menze, M., Heipke, C., Geiger, A.: Joint 3d estimation of vehicles and scene flow. ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences **2** (2015)
27. Pang, J., Sun, W., Ren, J.S., Yang, C., Yan, Q.: Cascade residual learning: A two-stage convolutional neural network for stereo matching. In: Proceedings of the IEEE International Conference on Computer Vision Workshops. pp. 887–895 (2017)
28. Poggi, M., Mattoccia, S.: Learning from scratch a confidence measure. In: BMVC (2016)
29. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015)
30. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4510–4520 (2018)
31. Seki, A., Pollefeys, M.: Patch based confidence prediction for dense disparity map. In: BMVC. vol. 2, p. 4 (2016)

32. Shen, S.: Accurate multiple view 3d reconstruction using patch-based stereo for large-scale scenes. IEEE transactions on image processing **22**(5), 1901–1914 (2013)
33. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1–9 (2015)
34. Tonioni, A., Tosi, F., Poggi, M., Mattoccia, S., Stefano, L.D.: Real-time self-adaptive deep stereo. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 195–204 (2019)
35. Tulyakov, S., Ivanov, A., Fleuret, F.: Practical deep stereo (pds): Toward applications-friendly deep stereo matching. In: Advances in Neural Information Processing Systems. pp. 5871–5881 (2018)
36. Wang, H., Lin, J., Wang, Z.: Design light-weight 3d convolutional networks for video recognition temporal residual, fully separable block, and fast algorithm. arXiv preprint arXiv:1905.13388 (2019)
37. Wang, P., Chen, P., Yuan, Y., Liu, D., Huang, Z., Hou, X., Cottrell, G.: Understanding convolution for semantic segmentation. In: 2018 IEEE winter conference on applications of computer vision (WACV). pp. 1451–1460. IEEE (2018)
38. Wang, Y., Lai, Z., Huang, G., Wang, B.H., Van Der Maaten, L., Campbell, M., Weinberger, K.Q.: Anytime stereo image depth estimation on mobile devices. In: 2019 International Conference on Robotics and Automation (ICRA). pp. 5893–5900. IEEE (2019)
39. Wofk, D., Ma, F., Yang, T.J., Karaman, S., Sze, V.: Fastdepth: Fast monocular depth estimation on embedded systems. In: 2019 International Conference on Robotics and Automation (ICRA). pp. 6101–6108. IEEE (2019)
40. Xie, C.W., Zhou, H.Y., Wu, J.: Vortex pooling: Improving context representation in semantic segmentation. arXiv preprint arXiv:1804.06242 (2018)
41. Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1492–1500 (2017)
42. Xie, S., Sun, C., Huang, J., Tu, Z., Murphy, K.: Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 305–321 (2018)
43. Xu, X., Hou, Y., Wang, P., Jiang, Z., Li, W.: Light weight stereo matching via deep extraction and integration of low and high level information. In: 2019 IEEE International Conference on Multimedia and Expo (ICME). pp. 320–325. IEEE (2019)
44. Žbontar, J., LeCun, Y.: Stereo matching by training a convolutional neural network to compare image patches. The journal of machine learning research **17**(1), 2287–2318 (2016)
45. Zhang, F., Prisacariu, V., Yang, R., Torr, P.H.: Ga-net: Guided aggregation net for end-to-end stereo matching. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 185–194 (2019)
46. Zhang, X., Zhou, X., Lin, M., Sun, J.: Shufflenet: An extremely efficient convolutional neural network for mobile devices. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 6848–6856 (2018)