

SHUFFLEMONO: RETHINKING LIGHTWEIGHT NETWORK FOR SELF-SUPERVISED MONOCULAR DEPTH ESTIMATION

Yingwei Feng, Zhiyong Hong*, Liping Xiong, Zhiqiang Zeng, Jingmin Li

*College of Electronic and Information Engineering, Wuyi University,
Jiangmen, Guangdong, China*

**E-mail: hongmr@163.com*

Submitted: 11th December 2023; Accepted: 27th February 2024

Abstract

Self-supervised monocular depth estimation has been widely applied in autonomous driving and automated guided vehicles. It offers the advantages of low cost and extended effective distance compared with alternative methods. However, like automated guided vehicles, devices with limited computing resources struggle to leverage state-of-the-art large model structures. In recent years, researchers have acknowledged this issue and endeavored to reduce model size. Model lightweight techniques aim to decrease the number of parameters while maintaining satisfactory performance. In this paper, to enhance the model's performance in lightweight scenarios, a novel approach to encompassing three key aspects is proposed: (1) utilizing LeakyReLU to involve more neurons in manifold representation; (2) employing large convolution for improved recognition of edges in lightweight models; (3) applying channel grouping and shuffling to maximize the model efficiency. Experimental results demonstrate that our proposed method achieves satisfactory outcomes on KITTI and Make3D benchmarks while having only 1.6M trainable parameters, representing a reduction of 27% compared with the previous smallest model, Lite-Mono-tiny, in monocular depth estimation.

Keywords: lightweight, self-supervised learning, depth estimation, monocular

1 Introduction

Depth estimation is a prominent problem in the field of computer vision (CV), aiming to infer the depth information of objects in a scene from images captured by cameras. This technology finds extensive applications across various domains, including 3D reconstruction [1], augmented reality [2], robotic vision [3, 4], virtual reality [5, 6], and more. Common methods for depth estimation typically rely on multiple cameras or sensors like LiDAR [7, 8, 9], which impose limitations on their practicality and cost-effectiveness. In contrast, monoc-

ular depth estimation technology enables accurate scene depth estimation using only a single camera, thereby reducing costs and enhancing application flexibility.

The traditional monocular depth estimation is to calculate the depth map from 2D images. The main methods include manual optimization [10], supervised [11] and self-supervised learning [12]. Inspired by the Structure From Motion (SFM) algorithm [13], some recent studies [14, 15] use a self-supervised learning approach to calculating depth from a single image. Despite the model's com-

mendable performance, [16, 17] have come to realize that conventional models cannot be used effectively on terminals with limited computing resources. They keep shrinking the size of the models to speed up the efficiency of the models, but at the same time, the performance decreases significantly.

On the one hand, the number of parameters in a deep learning model determines the upper limit of its nonlinear representational capacity, and generally models with fewer parameters exhibit suboptimal performance. On the other hand, activation functions introduce nonlinearity into the model by selectively activating or deactivating neurons, implying that not all neurons are activated. Consequently, the activation ratio of neurons bears a close relationship with the model's nonlinear representational capacity, which holds particular significance for lightweight models operating within constrained parameter spaces. To address these issues, we have undertaken subsequent efforts: (1) further reducing the trainable parameters to gain advantages in training, application, and storage, and enabling their utilization on lightweight devices; (2) additionally, increasing the utilization rate of neurons by employing diverse activation functions, which activate additional neurons to represent more complex nonlinear functions and subsequently enhance overall model performance.

Inspired by ShuffleNet [18], the present study introduces two novel blocks, they are the Group Convolution Block (GCB) and Multi Convolution Block (MCB), as a lightweight method for image feature extraction. These blocks are utilized to construct the model named ShuffleMono. The model demonstrates superior performance compared with previous competitive models, as evidenced by its significantly lower error percentage, and fewer parameters. The ShuffleMono exhibits three notable strengths:

Lightweight architecture. In the case of optimal model performance, the Group Convolution Block (GCB) is only about $\frac{1}{8}$ of the trainable parameters of the full connection layer. ShuffleMono's overall parameter count is only 73% of the previous smallest model, Lite-Mono-tiny, at 1.6M, which is suitable for scenarios with insufficient memory.

Effective architecture. The LeakyReLU enhances the parametric efficiency of ShuffleMono compared with previous models, achieving im-

provements from 95%, 49%, and 24.91% to 98.68%, 49.67%, and 24.96% at stages 2, 3, and 4 respectively (in Figure 1) of the structure from Lite-Mono-tiny [17] to ShuffleMono, as depicted in Table 4.

State-of-the-art accuracy. The ShuffleMono achieves state-of-the-art performance, surpassing Lite-Mono [17] with a depth accuracy of 0.872 on the KITTI Eigen split test set [19]. This represents a notable advancement over the previous relevant lightweight models.

The remainder of the paper is organized as follows. Section 2 reviews some related research work. Section 3 illustrates the proposed method in detail. Section 4 elaborates on the experimental results. Section 5 concludes the paper.

2 Related Work

Although self-supervised monocular depth estimation has been under development for numerous years, researchers have recently observed that the suitability of larger models for devices with limited resources are diminishing. In order to create a lightweight network structure, inspired by general advanced lightweight techniques, researchers consider the specific characteristics of depth estimation networks [16, 20, 17]. Therefore, this section elaborates on three aspects: self-supervised monocular depth estimation, lightweight network architecture for CV, and lightweight network architecture for self-supervised monocular depth estimation.

2.1 Self-supervised monocular depth estimation

Supervised monocular depth estimation necessitates ground-truth data, typically provided by LiDAR, to facilitate model training. However, the utilization of LiDAR incurs additional costs and poses challenges for further enhancing model performance in real-world scenarios without LiDAR's support. In contrast, Zhou et al. [14] propose a self-supervised learning approach that leverages depth values and pose transformations to reconstruct the original image. By comparing losses with the structural characteristics of the original image, these components can be jointly trained while also being used independently. To mitigate the impact of

static objects and occlusions on depth calculation, Godard et al. [15] introduce dynamic masks and continuous smoothing loss in Monodepth2, which remain widely adopted to date. Yan et al. [21] present CADepth-Net incorporating a channel attention mechanism to enhance depth accuracy at object edges further. Zhao et al.'s MonoViT [22] integrates transformer modules into both encoder and parallel convolution layers simultaneously, resulting in significant improvements in depth precision. He et al.'s resolution adaptive method [23], comprising multi-path encoders and decoders, enables accurate depth inference. Additionally, Shim et al.'s SwinDepth [24], a convolution-free Swin Transformer architecture, captures local geometric features as well as global semantic features for robust depth estimation.

2.2 Lightweight network architecture for CV

Due to the limited volume and power consumption, the computing power of edge computing terminals is constrained. There exist various approaches to enabling model utilization on edge terminals, with one such approach being the reduction of model parameters. An approximated convolution structure has been proposed [25], which achieves comparable performance to traditional convolution while reducing parameter count in convolutional networks. MobileNet [26, 27, 28] introduces depth-wise separable filters that divide conventional convolution layers into depth-wise and point-wise convolutions, thereby enhancing CPU computation efficiency. ShuffleNet [18, 29] presents grouping convolution and channel shuffle operations to further enhance parameter efficiency. In the era of visual transformers, hybrid models combining convolution and transformers have become prevalent in lightweight scenarios. Notable examples include MobileViT [30], ScalableViT [31], and MobileFormer [32]. To reduce computational load within transformer modules, a split self-attention module with vertical and horizontal spatial attention is designed [33]. Additionally, a novel self-attention computation method employing vector computations instead of complex matrix computations is proposed [34], which results in reduced memory usage.

2.3 Lightweight network architecture for self-supervised monocular depth estimation

With the increasing influence of self-attention mechanisms in computer vision, researchers have observed a significant growth in model size and computational requirements. In response to this, lightweight model structures have emerged to cater to devices with limited computing resources. Zhou et al. [16] propose R-MSFM, which reduces the parameters of Monodepth2 by 73%, from 14.3M to 3.8M. Similar to U-Net [35], R-MSFM incorporates a multi-scale feature modulation module and iteratively updates an inverse depth estimation. Hui et al. [20] introduce RM-Depth which utilizes recurrent modulation units for fusing encoder and decoder features while incorporating a motion network during training for improved performance, resulting in a reduced parameter count of 2.97M. Zhang et al. [17] employ dilated convolutions within their convolutional layers to enhance image feature extraction without compromising on accuracy or model efficiency despite being a hybrid model combining convolution and transformer architectures; its minimal configuration consists of only 2.2M parameters.

3 Method

In this section, the design of lightweight is first introduced, followed by the model architecture, ShuffleMono, mainly including two components, DepthNet and PoseNet. Finally, the image reprojection and loss function for self-supervised monocular depth estimation are introduced.

3.1 The Design of Lightweight

In the following, we illustrate the design of lightweight from three aspects.

Reduce the trainable parameters. A fully connected layer (FC) refers to a neural network architecture where each neuron applies a linear transformation to the input vector using a weights matrix [22, 17, 24]. It is commonly employed in self-attention modules for channel-wise projection of inputs, multi-layer perception (MLP), and output layers, among others. The number of parameters asso-

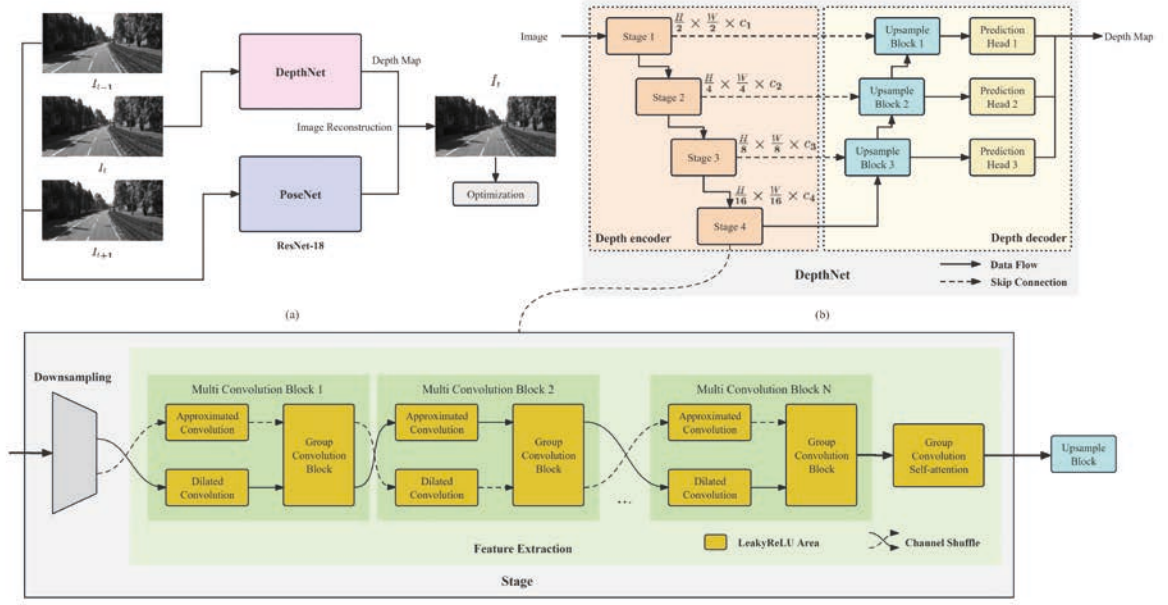


Figure 1. Structures of the proposed ShuffleMono. (a) The depth map and pose information obtained from DepthNet and PoseNet are utilized to reconstruct the target image, serving as the supervisory signal for self-supervised learning of depth. (b) The network is divided into four stages, each comprising two parts: downsampling and feature extraction. For the sake of efficiency, stage 1 employs traditional convolution layers for feature extraction. In stages 2 to 4, multiple multi-convolution blocks are utilized along with a single group convolution self-attention block.

ciated with this layer can be computed as follows:

$$Params_{FC} = N_{in\_dim} \times N_{out\_dim} + N_{out\_dim} \quad (1)$$

where $Params_{FC}$ is the number of FC parameters, N_{in_dim} , and N_{out_dim} is the number of input dimension and output dimension in MLP, respectively. The fully connected (FC) layer in the multi-layer perceptron (MLP) serves as a channel-wise projection, thereby approximating $1 \times 1 \times N_{out\_dim}$ convolution in the spatial dimension. To further reduce parameter count, group convolution can be employed in the spatial domain. The effectiveness of group convolution, initially introduced in distributing the AlexNet [36] across two GPUs, has been convincingly demonstrated in ResNeXt [37]. The number of parameters for this approach is computed as follows:

$$Params_{conv} = K \times K \times \frac{N_{in\_dim}}{G} \times \frac{N_{out\_dim}}{G} \times G \quad (2)$$

where $Params_{conv}$ is the number of convolution parameters, K is the size of the convolution kernel, and G is the number of grouping. The number of parameters in the equivalent grouped convolution

can be approximated as $\frac{1}{G}$ of the fully connected layer, based on a comparison between the two formulas.

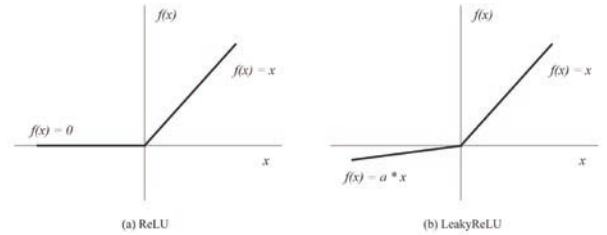


Figure 2. Various activation functions. (a) The rectified linear unit (ReLU) function is truncated to zero in the negative value region. (b) The LeakyReLU activation function exhibits a weighted negative response in the region of negative values, characterized by a small gradient.

Improve the efficiency of limited parameters. Since the group convolution's convolution kernel can only capture a portion of the feature maps, it is not precisely equivalent to fully connected layers (FC). Drawing inspiration from ShuffleNet [18], which enhances parameter efficiency by exchange-

ing channels between group convolution layers, this work extends the concept of the ShuffleNet Unit in the channel dimension for lightweight models. Additionally, The ReLU [38] function is observed that does not promote neuron activation. As depicted in Figure 2, unlike the ReLU function where negative values are always zero, LeakyReLU [39] assigns a weighted negative value to the negative range, resulting in a small gradient during network back-propagation. This is anticipated to stimulate more neurons and consequently enhance the performance of lightweight networks.

Enhanced edge features of objects. When analyzing the performance of the previous model [15, 16, 17], as depicted in Figure 3, it is observed that errors tend to be more pronounced in regions containing small objects or along their edges compared with other areas. This phenomenon can be attributed to the sensitivity of the model’s receptive field towards relevant features. In Lite-Mono, multiple dilated convolutions are employed to achieve diverse receptive fields without increasing parameters and computational complexity. However, dilated convolution relies on sparse sampling which leads to some pixels being unused when overlapping multiple dilated convolutions, thereby compromising information continuity and correlation. Consequently, this diminishes the significance of neighboring pixels for small objects or edge detection. To address this issue while maintaining model lightness (as large convolution kernels require additional parameters and computation), decomposing the convolution operation is proposed by employing traditional convolutions with identical receptive fields.

In all, the aforementioned three aspects facilitate the reconstruction of the depth encoder, which is included in the DepthNet (as depicted in Figure 1), enabling it to effectively represent a wider range of image features within constrained parameters. This not only reduces parameter complexity but also enhances overall performance.

3.2 Model Architecture

The model architecture of the ShuffleMono is introduced (refer to Figure 1). ShuffleMono mainly includes two parts, they are DepthNet and PoseNet. Firstly, we briefly outline the structure of PoseNet, which is proposed in [15, 16, 17]. The pose en-

coder is a pre-trained ResNet18 that takes in a pair of color images as input. A pose decoder with four convolutional layers is employed to estimate the 6-DoF relative pose between adjacent images.

Secondly, our primary focus lies on DepthNet, which is extensively elaborated upon and emphasizes some of our significant contributions. The DepthNet follows a structure similar to that of U-Net, comprising two main components: depth encoder and depth decoder, which are further divided into four stages. The Depth Encoder encompasses downsampling and feature extraction processes, with the latter incorporating multiple Multi Convolution Blocks and one Group Convolution Self-attention module.

3.2.1 Depth Encoder

In Figure 1, like Lite-Mono, the proposed ShuffleMono consists of four stages, and the feature maps of each stage are extracted for final fusion. The input image with size $H \times W \times 3$ is fed into the first stage for down-sampling, which consists of three 3×3 convolutions. The feature maps of size $\frac{H}{2} \times \frac{W}{2} \times c_1$ can be obtained after the first stage. The next second to fourth stages is composed of three parts, as illustrated in Figure 1, namely downsampling, Multi Convolution Block (MCB), and Group Convolution Self-attention (GCS). The proposed MCB and GCS can capture rich local and global feature representations, and output features of dimensions $\frac{H}{4} \times \frac{W}{4} \times c_2$, $\frac{H}{8} \times \frac{W}{8} \times c_3$, and $\frac{H}{16} \times \frac{W}{16} \times c_4$. The network configuration is presented in Table 1.

The downsampling is responsible for reducing data dimensionality and augmenting the channel capacity. It can be observed from Table 1 that the feature map decreases at intervals of $\frac{1}{4}$ resolution while the channel expands at a rate of 2x. MCB and GCS are accountable for extracting image features, with varying convolution kernel sizes employed in each MCB to refine richer features. Detailed configuration information is provided in Table 1. Different sizes of convolution kernels can yield varying receptive fields, which are commonly employed by researchers to enhance image features in parallel methods. However, this approach often leads to increased network size. To address this issue and reduce the number of parameters in the network, concatenated kernels with different sizes are adopted similar to Lite-Mono. In contrast to Lite-



Figure 3. Distribution of errors. The uppermost image depicts the error distribution map, wherein white dots are utilized to visually represent the magnitude of errors. (The opacity of each dot decreases as the error value increases.) It can be observed that errors primarily concentrate around small objects and edges. The bottom left image represents the original data, while the bottom right image displays the depth map. In the depth map, small objects are not discernible and edges lack clarity.

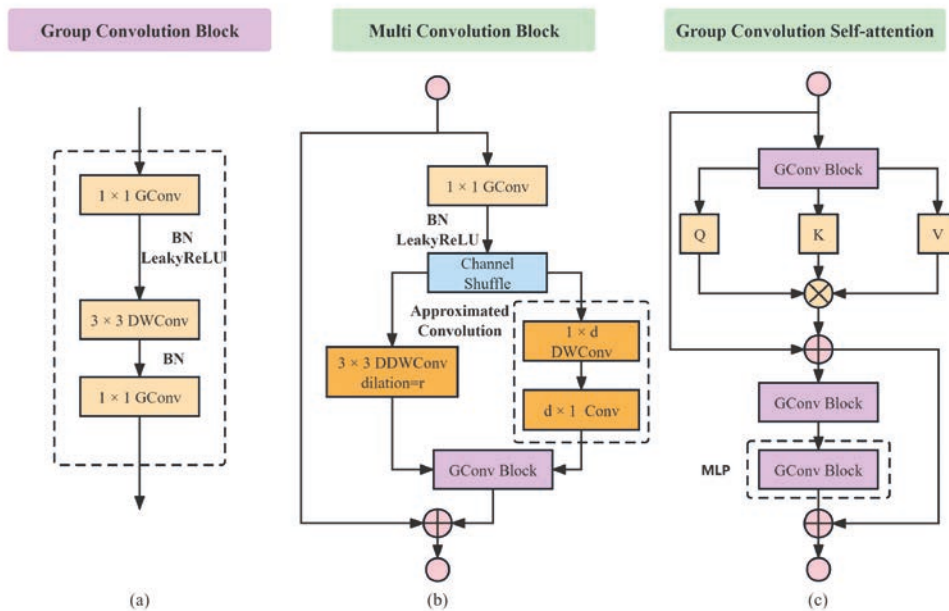


Figure 4. Structures of the proposed three different blocks. (a) The Group Convolution Block consists of two point-wise convolution layers sandwiched between 3×3 depth-wise convolutions. (b) The Multi Convolution Block comprises a series of dilated convolutions and approximated convolutions, followed by a GConv Block, ultimately yielding a residual output. (c) Both the projection layers and MLP of the Group Convolution Self-attention employ GConv Blocks instead of fully connected layers

Table 1. Structures of the proposed encoder

Output Size	Stages	Channels	Groups	Downsampling	Multi Convolution Block	Group Convolution Self-attention
					Kernel Size	
640×192	Input					
320×96	Stage 1	$c_1 = 48$	$g_1 = 1$	✓		
160×48	Stage 2	$c_2 = 48$	$g_2 = 2$	✓	[3, 5, 7]	✓
80×24	Stage 3	$c_3 = 96$	$g_3 = 4$	✓	[3, 5, 7]	✓
40×12	Stage 4	$c_4 = 192$	$g_4 = 8$	✓	[3, 5, 7, 5, 9, 13]	✓

Note: Except for stage 1 and stage 2, which maintain their original number of channels, the subsequent stages exhibit a twofold increase in both channel count and group count compared with the preceding stage. This effectively regulates the overall parameter growth.

Mono’s utilization of dilated convolution, our proposed method focuses on capturing object edge features through full receptive field convolution. To enhance the network’s ability to perceive finer details, we vary the convolution from 3×3 to 13×13 .

Group Convolution Block (GCB). Since MLP operates in the channel dimension, it is conceptually equivalent to a 1×1 point-wise convolution in the spatial dimension. Consequently, employing a 1×1 group point-wise convolution can effectively reduce the number of trainable parameters. To compensate for any potential performance degradation, we incorporate both 3×3 depth-wise separable convolutions and 1×1 group point-wise convolutions in the spatial dimension to extract features, as shown in Figure 4 (a). Taking stage 4 as an example, with a channel count of 192, eight groups, and an expansion coefficient of six, this approach results in a total parameter count that is only 12.93% compared with when MLP is utilized.

Multi Convolution Block (MCB). The convolution has the advantage of lower parameter requirement and computational complexity compared with transformer modules, which typically serve as the primary function for capturing image features in a lightweight CV network. To compress features, we propose a 1×1 layer with point-wise group convolution as an input. In addition to utilizing dilated convolution modules, approximated convolution modules [25] of equivalent size are incorporated for integrating features from different receptive fields, as shown in Figure 4 (b). Convolution by one of the original filters $W_n * z = \sum_{c=1}^C W_n^c * z^c$ is approximated by

$$W_n * z \cong \sum_{c=1}^C \left[\sum_{k=1}^K h_n^k * v_k^c \right] * z^c \quad (3)$$

where z^c is the input channel. The first convo-

lutional layer has K filters of spatial size $d \times 1$, which results in a filter bank $\{v_k \in \mathbb{R}^{d \times 1 \times C} : k \in [1 \dots K]\}$ and produces output feature maps V such that $V(u, v) \in \mathbb{R}^K$. The second convolutional layer has N filters of spatial size $1 \times d$, which results in a filter bank $\{h_n \in \mathbb{R}^{1 \times d \times K} : n \in [1 \dots N]\}$.

Group Convolution Self-attention (GCS).

The structure presented in Figure 4 (c) demonstrates that Group Convolution Self-attention uses GCB in place of full connection projection layers and MLP. Unlike full connection projection on channel-wise, GCB is projected on spatial-wise. For example, the input feature map X with $C \times H \times W$ is projected to the same channel queries $Q = XW_q$, keys $K = XW_k$, and values $V = XW_v$, where W_q , W_k , and W_v are weight matrices. The number of parameters for using a full connection projection is:

$$W = H \times W \times C_{in} \times 3 \times C_{out} \quad (4)$$

The number of parameters for GCB is:

$$W = (K_{dw}^2 \times 3 \times C_{out}) + 2 \times (K_{pw}^2 \times \frac{C_{in}}{G} \times 3 \times C_{out}) \quad (5)$$

where W is the sum of W_q , W_k , and W_v , K is the kernel size of depth-wise and point-wise filters respectively, and G is the number of groups for grouped filters. Since $K_{pw} < K_{dw} \ll H$ and W , the overall parameter number is greatly reduced.

3.2.2 Depth Decoder

By the Lite-Mono [17] design, DepthNet’s decoder incorporates two 3×3 convolutional layers sandwiched between a bilinear upsampling structure, as shown in Figure 5. Diverging from Lite-Mono’s approach of combining feature maps from the last three stages, the ShuffleMono additionally integrates the feature combination of stage 1. Each

upsampling module is linked to a prediction head for generating depth maps at full, $\frac{1}{2}$, and $\frac{1}{4}$ resolutions respectively.

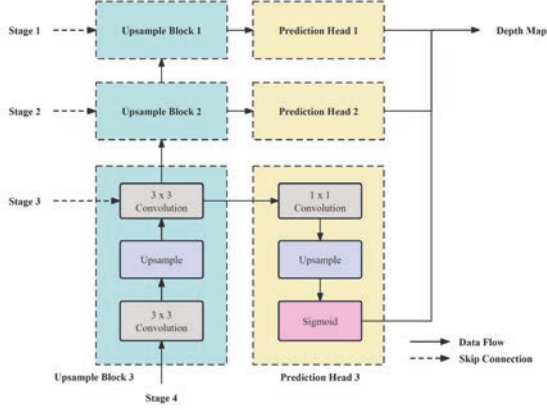


Figure 5. Structures of the proposed three different components. The Upsample Block is composed of two layers of 3×3 convolutions and a bilinear interpolation method for upsampling.

The Prediction Head comprises a 1×1 convolution, upsample, and sigmoid output.

3.3 Self-supervised monocular depth learning

Self-supervised monocular depth estimation [14, 15] uses depth map and pose data to reconstruct the original image I_t . By comparing with the original I_t and reconstructed image $\hat{I}_t$, training the DepthNet and PoseNet simultaneously eliminates the dependence on the ground-truth of depth.

Image reprojection. The image reprojection is calculated as:

$$\hat{I}_t = K \hat{P}_{t-1 \rightarrow t} \hat{D}_t(I_t) K^{-1} I_{t-1} \quad (6)$$

where K is the camera's intrinsics, and $\hat{P}_{t-1 \rightarrow t}$ is the pose estimation matrix from $t-1$ to t , and $\hat{D}_t$ is the predicted depth. Following the image reconstruction of Monodepth2 [15], $\mathcal{L}_p$ consists of a sum of the pixel-wise similarity SSIM [40] and a SmoothL1 [41] between $\hat{I}_t$ and I_t :

$$\mathcal{L}_p(\hat{I}_t, I_t) = \frac{a}{2}(1 - \text{SSIM}(\hat{I}_t, I_t)) + (1 - a)\|\hat{I}_t - I_t\| \quad (7)$$

where a is set to 0.85 empirically [15].

Following an edge-aware of Monodepth2 [15] and DIFFNet [42], smoothness loss is used to

smooth the generated depth maps:

$$L_{\text{smooth}} = |\partial_x d_t^*| e^{-|\partial_x I_t|} + |\partial_y d_t^*| e^{-|\partial_y I_t|} \quad (8)$$

where $d_t^* = \frac{d_t}{d_t}$ denotes the mean-normalized depth.

4 Experiments

After introducing two datasets KITTI and Make3D, the implementation details are given, including hyperparameters, data argumentation approaches, and evaluation metrics. Then two experimental results about KITTI and Make3D are presented respectively. Finally, we perform three ablation experiments on numbers of groups, activation functions, and approximated convolution.

4.1 Datasets

KITTI. The KITTI [43] dataset, co-sponsored by Karlsruhe and the Toyota Institute for research in autonomous driving, comprises up to six hours of real-world traffic data. This comprehensive dataset includes synchronized and corrected images, LiDAR scans, high-precision GPS information, and IMU acceleration data. It consists of 61 stereo road scenes commonly used for research in the fields of autonomous driving and robotics. This dataset is employed to train and evaluate the proposed method.

Specifically, the Eigen split [19] is utilized, which provides a total of 39,180 monocular triplets for training purposes, along with 4,424 triplets for evaluation and 697 triplets for testing. To ensure consistency during training, the same intrinsics are adopted by averaging the focal lengths across all images in the KITTI dataset [15]. During evaluation, we constrain predicted depth values within a range of [0m - 80m], following common practice.

Make3D. The Make3D dataset [44], introduced by Ashutosh Saxena et al. for 3D structure reconstruction from a single image, comprises 400 single training RGB and depth map pairs, and 134 test samples depicting outdoor scenes, which is used to be a test set for the model trained on the KITTI dataset. This set can serve as a valuable resource for assessing the proposed model's generalization capabilities.

4.2 Implementation details

Hyperparameters. The proposed method is implemented using PyTorch and trained with a batch size of 12. When training models from scratch, an initial learning rate of $5e^{-4}$ is selected, and the training epoch is set to 35.

Data augmentation. To enhance the robustness of the training, data augmentation is utilized in a preprocessing step. Specifically, horizontal flips and color augmentation are each applied with a 50% probability. Color augmentation includes brightness, saturation, and contrast adjustments within the range of [80% - 120%], as well as hue jitter within the range of [-10% - +10%]. These augmentations are randomly applied in a specific order, and the same method is also adopted by [15, 16, 17].

Evaluation metrics. The accuracy is assessed based on seven commonly employed metrics proposed in [45], encompassing four depth errors, they are *AbsRel*, *SqRel*, *RMSE*, *RMSElog*, and three kinds of depth accuracy, it is *Accuracies*. They are formulated as follows:

$$AbsRel = \frac{1}{|N|} \sum_{i \in N} \frac{|d_i - d_i^*|}{d_i^*} \quad (9)$$

$$SqRel = \frac{1}{|N|} \sum_{i \in N} \frac{\|d_i - d_i^*\|_2}{d_i^*} \quad (10)$$

$$RMSE = \sqrt{\frac{1}{|N|} \sum_{i \in N} \|d_i - d_i^*\|_2} \quad (11)$$

$$RMSElog = \sqrt{\frac{1}{|N|} \sum_{i \in N} \|\log(d_i) - \log(d_i^*)\|_2} \quad (12)$$

$$Accuracies = \max\left(\frac{d_i}{k_i^*}, \frac{d_i^*}{d_i}\right) = \delta < threshold \quad (13)$$

where N is the total number of pixels for depth ground truth, d_i denotes the predicted depth value at pixel i , and d_i^* stands for the ground truth at pixel i . In addition, the *threshold* controls the percentage of correct pixels in the estimated depth, which can be taken as $\delta < 1.25$, $\delta < 1.25^2$, or $\delta < 1.25^3$.

4.3 KITTI results

The results of Table 2 present a comparison with 7 competitive models, they are Monodepth2-Res18 [15], Monodepth2-Res50 [15], R-MSFM6

[16], R-MSFM6 [16], Lite-Mono-tiny [17], Lite-Mono-small [17], and Lite-Mono [17]. ShuffleMono stands out with the smallest number of parameters (1.6M), which is 0.6M less than the previous smallest Lite-Mono-tiny (2.2M), and only 1/20 of the largest Monodepth2-Res50 (32.5M). In terms of performance, it surpasses the best Lite-Mono (3.1M) while having nearly half the parameter count due to a higher proportion of valid parameters that retain more effective features. Take the comparison with Lite-Mono as an example, in ShuffleMono, *AbsRel* decreases from 0.121 to 0.114, and *Accuracy* increases from 0.859 to 0.872, demonstrating greater improvement compared with previous model enhancements.

When analyzing five distinct images featuring small objects, as depicted in the first column of Figure 6, ShuffleMono consistently exhibits satisfactory performance in recognizing small objects and edges. Remarkably, it precisely discerns the narrow gap between the two objects positioned in the upper right corner of the first image. In comparison to Lite-Mono-tiny, ShuffleMono demonstrates enhanced capability in detecting rapid variations in scene depth. Furthermore, ShuffleMono demonstrates enhanced depth estimation capabilities for small objects, such as the circular traffic sign in the top right corner of the second image and the pole located to the right of the third image. Lastly, through examination of the fourth and fifth images, it becomes apparent that ShuffleMono excelled at precisely delineating object boundaries.

Additionally, the inference speed of ShuffleMono is compared with that of Lite-Mono-tiny. Two different experimental settings are established: (Setting 1) on the NVIDIA A100 GPU platform, both models are tested eight times using 697 KITTI test set images; (Setting 2) on the AMD EPYC 7702 CPU platform, eight tests are conducted using 16 KITTI test set images. Table 3 presents the maximum, minimum, and average time taken for each of these tests. The ShuffleMono exhibits a processing time of 6ms per image on the A100, slightly longer than lite-mono-tiny's 5ms. However, it achieves a significantly improved inference time of only 235ms per image on the EPYC 7702, representing a remarkable enhancement of 34.17% compared with lite-mono-tiny.

Table 2. Experiment results on the KITTI

Method	Depth Error(↓)				Depth Accuracy(↑)			Model Size(↓)
	AbsRel	SqRel	RMSE	RMSElog	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	Params.
Monodepth2-Res18 [15]	0.132	1.044	5.142	0.210	0.845	0.948	0.977	14.3M
Monodepth2-Res50 [15]	0.131	1.023	5.064	0.206	0.849	0.951	0.979	32.5M
R-MSFM3 [16]	0.128	0.965	5.019	0.207	0.853	0.951	0.977	3.5M
R-MSFM6 [16]	0.126	0.944	4.981	0.204	0.857	0.952	0.978	3.8M
Lite-Mono-tiny [17]	0.125	0.935	4.986	0.204	0.835	0.950	0.978	<u>2.2M</u>
Lite-Mono-small [17]	0.123	0.919	4.926	0.202	0.859	0.951	0.977	2.5M
Lite-Mono [17]	<u>0.121</u>	<u>0.876</u>	4.918	0.199	<u>0.859</u>	<u>0.953</u>	0.980	3.1M
ShuffleMono(Ours)	0.114	0.850	4.821	0.193	0.872	0.957	0.980	1.6M

Note: Comparison of ShuffleMono with some recent representative methods on the KITTI [43] benchmark using the Eigen split. The size of all input images is 640×192 , and all models are trained from scratch. The best results are highlighted in **bold**, and the second best is underlined.

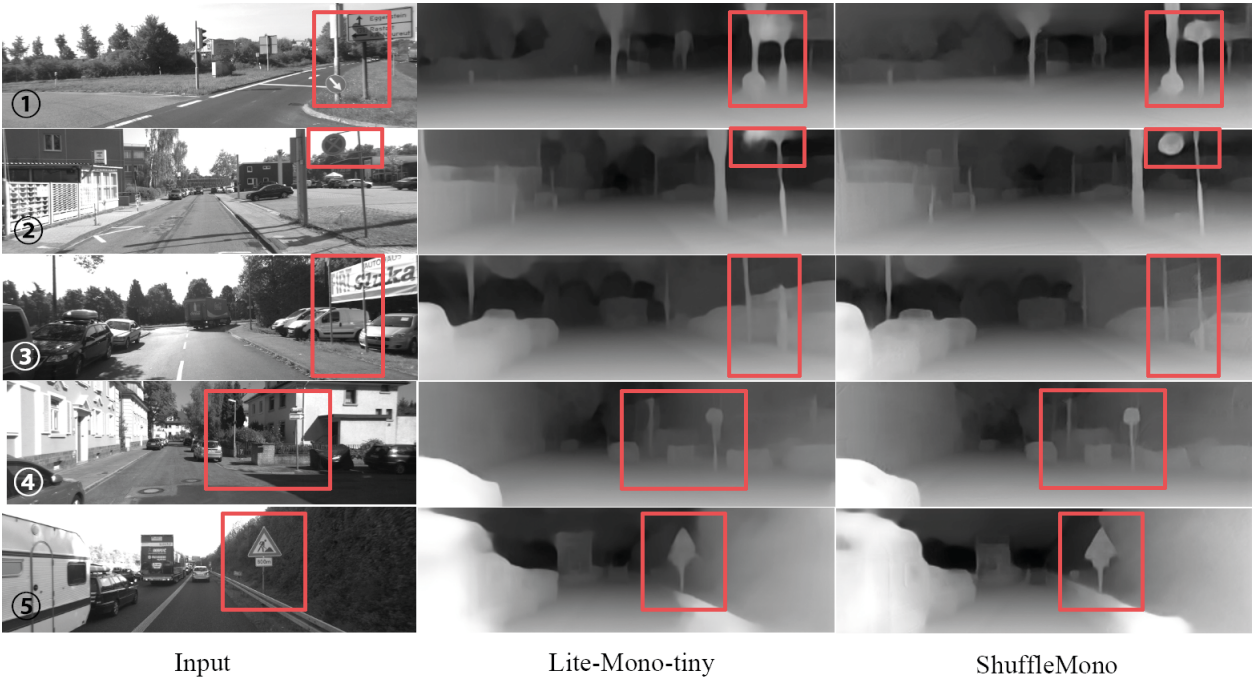


Figure 6. Qualitative results on KITTI. Here are some depth maps generated by Lite-Mono-tiny [17] and ShuffleMono (our model). Despite its smaller size, ShuffleMono exhibits superior edge recognition of objects compared with previous models. Specifically, it demonstrates enhanced depth estimation capabilities for small objects or edges with significant depth variations.

Table 3. Experiment results of inference speed

Setting	Lite-Mono-tiny			ShuffleMono		
	Min(s)	Max(s)	Average(s)	Min(s)	Max(s)	Average(s)
1	3.097	3.947	3.533	3.605	4.522	4.223
	Per Image: 5ms			Per Image: 6ms		
2	5.528	5.900	5.711	3.340	4.142	3.756
	Per Image: 357ms			Per Image: 235ms		

Note: The best results are highlighted in **bold**.

Finally, the effective features of ShuffleMono are compared with that of Lite-Mono-tiny. The output feature maps of each stage in the ShuffleMono are extracted, and features with an absolute value greater than 0.01 are considered effective for identification. In the experiments, the feature maps are counted from stages 2 to 4 for both Lite-Mono-tiny and ShuffleMono models separately. The two results of the experiments are counted: (Result 1) the rate of effective features; (Result 2) the number of effective features. Table 4 presents the experimental data for both models. Based on the experimental results, ShuffleMono demonstrates higher feature efficiency compared with Lite-Mono-tiny, particularly at stage 2.

Table 4. Experiment results of activated features

Result	Lite-Mono-tiny			ShuffleMono		
	Stage2	Stage3	Stage4	Stage2	Stage3	Stage4
1	94.99%	48.97%	24.91%	98.68%	49.67%	24.96%
2	233,460	60,182	15,309	363,790	91,551	23,009

4.4 Make3D results

The model’s generalization ability is validated by using diverse outdoor images from the Make3D [44] dataset. The models are trained on the KITTI [43] dataset and directly evaluated using test images. Among them, ShuffleMono exhibits satisfactory performance in terms of AbsRel, SqRel, and RMSE metrics, as illustrated in Table 5. Notably, the SqRel metric demonstrates a 33% improvement compared with the four models, they are DDVO [46], Monodepth2 [15], R-MSFM6 [16], and Lite-Mono [17], indicating a reduced error range for this model. Furthermore, despite having fewer parameters, ShuffleMono showcases an enhanced capability to generalize across unknown scenarios when compared with previous models.

Table 5. Experiment results on the Make3D

Method	AbsRel	SqRel	RMSE	RMSElog
DDVO	0.387	4.720	8.090	0.204
Monodepth2	0.322	3.589	7.417	<u>0.163</u>
R-MSFM6	0.334	3.285	7.212	0.169
Lite-Mono	<u>0.305</u>	<u>3.060</u>	<u>6.981</u>	0.158
ShuffleMono	0.281	2.049	6.519	0.167

Note: The size of all input images is 640×192 , and all models are trained from scratch. The best results are highlighted in **bold**, and the second best is underlined.

4.5 Ablation experiments

Ablation study on various number of groups.

When the number of feature maps is fixed, the number of parameters varies inversely with the number of groups. As the number of groups decreases, the number of parameters increases, which increases the computational complexity and improves the performance, and vice versa. Three different settings are employed for experiment: (Setting 1) set the number of groups in stages 2, 3, and 4 as 2, 4, and 8 respectively; (Setting 2) set the number of groups in stages 2, 3, and 4 as 4, 8, and 16 respectively; (Setting 3) set the number of groups in stages 2, 3, and 4 as 8, 16, and 32 respectively. Table 6 illustrates variations in evaluation metrics for different group settings. From the results, it can be observed that increasing the number of groups leads to a further decline in model performance, this is attributed to that each group of convolution has fewer observed feature maps, resulting in additional challenges for image feature extraction.

Table 6. Ablation study on various number of groups

Setting	AbsRel	SqRel	RMSE	RMSElog	δ_1	δ_2	δ_3
1	0.114	0.850	4.821	0.193	0.872	0.957	0.980
2	0.120	0.956	4.916	0.197	0.865	0.956	0.980
3	0.122	0.934	4.911	0.200	0.862	0.953	0.980

Ablation study on various activation functions. Although ReLU is widely used as an activation function, alternative activation functions have also demonstrated promising performance in specific domains. The experiments are conducted under two different settings: (Setting 1) selecting ReLU; (Setting 2) selecting LeakyReLU. Table 7 presents the variations in evaluation metrics for two different activation functions. The experi-

mental results indicate that employing LeakyReLU as the activation function leads to improve model performance. However, it should be noted that when ReLU is utilized as the activation function, the model exhibits smoother error patterns, particularly evident from SqRel indicators.

Table 7. Ablation study on various activation functions

Setting	AbsRel	SqRel	RMSE	RMSElog	δ_1	δ_2	δ_3
1	0.120	0.789	4.914	0.196	0.854	0.952	0.981
2	0.114	0.850	4.821	0.193	0.872	0.957	0.980

Ablation study on approximated convolution. Although dilated convolution expands the receptive field of traditional 3×3 convolution, the expanded receptive field differs from that of corresponding dimensional convolution. From the perspective of feature diversity, two different settings are employed in the experiments: (Setting 1) selecting dilated convolution as the image feature capture layer; (Setting 2) dividing the convolution kernel into halves, and using dilated convolution for one half and approximated convolution for the other half. Table 8 presents variations in evaluation parameters for different convolution kernels. The utilization of approximated convolution allows for the extraction of a greater number of features compared with dilated convolution, as evidenced by the significant enhancement observed across six main evaluation metrics. Specifically, AbsRel and SqRel exhibit a decrease of 0.006 and 0.066, respectively, while δ_1 demonstrates an increase of 0.005, thereby indicating an overall enhancement in error suppression and accuracy. As depicted in Figure 6, approximated convolution effectively captures the edge features of objects.

Table 8. Ablation study on approximated convolution

Setting	AbsRel	SqRel	RMSE	RMSElog	δ_1	δ_2	δ_3
1	0.120	0.916	4.861	0.197	0.867	0.955	0.980
2	0.114	0.850	4.821	0.193	0.872	0.957	0.980

5 Conclusions

This paper introduces a novel network architecture for lightweight self-supervised monocular depth estimation, which combines convolution and transformer modules in a hybrid structure to optimize the efficiency of each parameter while re-

ducing the overall number of network parameters. Inspired by the shuffle idea, both the convolution and transformer modules have been redesigned. This approach demonstrates significant advantages through experimental evaluations on the KITTI and Make3D datasets. Notably, adjusting the kernel size of convolution and the number of groupings can effectively impact both model performance and parameter count, enabling the model to achieve optimal results.

References

- [1] Tomasz Szmuc, Rafał Mrówka, Marek Brańka, Jakub Ficoń, Piotr Pieta, A Novel Method for Fast Generation of 3D Objects from Multiple Depth Sensors., *Journal of Artificial Intelligence and Soft Computing Research*, 2023, 13(2): 95-105.
- [2] Martin-Gomez, A., Li, H., Song, T., Yang, S., Wang, G., Ding, H., Navab, N., Zhao, Z., Armand, M., Sttar: surgical tool tracking using off-the-shelf augmented reality head-mounted displays., *IEEE Transactions on Visualization and Computer Graphics*, 2023, 1-16.
- [3] Rodrigues, R.T., Miraldo, P., Dimarogonas, D.V., Aguiar, A.P., A framework for depth estimation and relative localization of ground robots using computer vision., *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, 3719-3724.
- [4] Silva, R., Cielniak, G., Gao, J., Leaving the Lines Behind: Vision-Based Crop Row Exit for Agricultural Robot Navigation., Preprint at <https://arxiv.org/abs/2306.05869>, 2023.
- [5] Sharma, A., Nett, R., Ventura, J., Unsupervised learning of depth and ego-motion from cylindrical panoramic video with applications for virtual reality., *International Journal of Semantic Computing*, 2020, 14(03): 333-356.
- [6] Rasla, A., Beyeler, M., The relative importance of depth cues and semantic edges for indoor mobility using simulated prosthetic vision in immersive virtual reality., *Proceedings of the 28th ACM Symposium on Virtual Reality Software and Technology*, 2022, 1-11.
- [7] Patakin, N., Vorontsova, A., Artemyev, M., Konushin, A., Single-stage 3d geometry-preserving depth estimation model training on dataset mixtures with uncalibrated stereo data., *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, 1705-1714.

- [8] Peng, R., Wang, R., Wang, Z., Lai, Y., Wang, R., Rethinking depth estimation for multi-view stereo: A unified representation., *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, 8645-8654.
- [9] Choe, J., Joo, K., Imtiaz, T., Kweon, I.S., Volumetric propagation network: Stereo-lidar fusion for long-range depth estimation., *IEEE Robotics and Automation Letters*, 2021, 6(3): 4672-4679.
- [10] Hirschmuller, H., Accurate and efficient stereo processing by semi-global matching and mutual information., *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005, 2: 807-814.
- [11] Chang, J.-R., Chen, Y.-S., Pyramid stereo matching network., *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, 5410-5418.
- [12] Liu, P., King, I., Lyu, M.R., Xu, J., Flow2stereo: Effective self-supervised learning of optical flow and stereo matching., *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, 6648-6657.
- [13] Ullman, S., The interpretation of structure from motion., *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 1979, 203(1153): 405-426.
- [14] Zhou, T., Brown, M., Snavely, N., Lowe, D.G., Unsupervised learning of depth and ego-motion from video., *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017, 1851-1858.
- [15] Godard, C., Mac Aodha, O., Firman, M., Brostow, G.J., Digging into self-supervised monocular depth estimation., *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, 3828-3838.
- [16] Zhou, Z., Fan, X., Shi, P., Xin, Y., R-msfm: Recurrent multi-scale feature modulation for monocular depth estimating., *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, 12777-12786.
- [17] Zhang, N., Nex, F., Vosselman, G., Kerle, N., Lite-mono: A lightweight cnn and transformer architecture for self-supervised monocular depth estimation., *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, 18537-18546.
- [18] Zhang, X., Zhou, X., Lin, M., Sun, J., Shufflenet: An extremely efficient convolutional neural network for mobile devices., *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, 6848-6856.
- [19] Eigen, D., Fergus, R., Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture., *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2015, 2650-2658.
- [20] Hui, T.-W., Rm-depth: Unsupervised learning of recurrent monocular depth in dynamic scenes., *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, 1675-1684.
- [21] Yan, J., Zhao, H., Bu, P., Jin, Y., Channel-wise attention-based network for self-supervised monocular depth estimation., *2021 International Conference on 3D Vision (3DV)*, 2021, 464-473.
- [22] Zhao, C., Zhang, Y., Poggi, M., Tosi, F., Guo, X., Zhu, Z., Huang, G., Tang, Y., Mattoccia, S.: Monovit: Self-supervised monocular depth estimation with a vision transformer., *2022 International Conference on 3D Vision (3DV)*, 2022, 668-678.
- [23] He, M., Hui, L., Bian, Y., Ren, J., Xie, J., Yang, J., Ra-depth: Resolution adaptive self-supervised monocular depth estimation., *European Conference on Computer Vision*, 2022, 565-581.
- [24] Shim, D., Kim, H.J., Swindepth: Unsupervised depth estimation using monocular sequences via swin transformer and densely cascaded network., *arXiv preprint arXiv:2301.06715*, 2023.
- [25] Jaderberg, M., Vedaldi, A., Zisserman, A., Speeding up convolutional neural networks with low rank expansions., *Proceeding of the British Machine Vision Conference 2014. British Machine Vision Association*, 2014.
- [26] Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H., Mobilenets: Efficient convolutional neural networks for mobile vision applications., *arXiv preprint arXiv:1704.04861*, 2017.
- [27] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.-C., Mobilenetv2: Inverted residuals and linear bottlenecks., *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, 4510-4520.
- [28] Howard, A., Sandler, M., Chu, G., Chen, L.-C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., et al., Searching for mobilenetv3., *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, 1314-1324.
- [29] Ma, N., Zhang, X., Zheng, H.-T., Sun, J., Shufflenet v2: Practical guidelines for efficient cnn architecture design., *Proceedings of the European Conference on Computer Vision*, 2018, 116-131.

- [30] Mehta, S., Rastegari, M., Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer., International Conference on Learning Representations., 2021.
- [31] Yang, R., Ma, H., Wu, J., Tang, Y., Xiao, X., Zheng, M., Li, X., Scalablevit: Rethinking the context-oriented generalization of vision transformer., Proceedings of the European Conference on Computer Vision, 2022, 480–496.
- [32] Chen, Y., Dai, X., Chen, D., Liu, M., Dong, X., Yuan, L., Liu, Z.: Mobile-former: Bridging mobilenet and transformer. , Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, 5270–5279.
- [33] Ho, J., Kalchbrenner, N., Weissenborn, D., Salimans, T., Axial attention in multidimensional transformers., arXiv preprint arXiv:1912.12180, 2019.
- [34] Mehta, S., Rastegari, M., Separable self-attention for mobile vision transformers., Transactions on Machine Learning Research, 2022.
- [35] Ronneberger, O., Fischer, P., Brox, T., U-net: Convolutional networks for biomedical image segmentation., Medical Image Computing and Computer-Assisted Intervention–MICCAI, 2015, 234–241.
- [36] Krizhevsky, A., Sutskever, I., Hinton, G.E., Imagenet classification with deep convolutional neural networks., Advances in neural information processing systems, 2012, 5.
- [37] Xie, S., Girshick, R., Dollar, P., Tu, Z., He, K., Aggregated residual transformations for deep neural networks., Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2017, 1492–1500.
- [38] Glorot, X., Bordes, A., Bengio, Y., Deep sparse rectifier neural networks., Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, 2011, 315–323.
- [39] Maas, A.L., Hannun, A.Y., Ng, A.Y., et al., Rectifier nonlinearities improve neural network acoustic models., Proc. Icml, 2013, 30(1): 3.
- [40] Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P., Image quality assessment: from error visibility to structural similarity., IEEE transactions on image processing, 2004, 600–612.
- [41] Girshick, R., Fast r-cnn., Proceedings of the IEEE/CVF International Conference on Computer Vision, 2015, 1440–1448.
- [42] Zhou, H., Greenwood, D., Taylor, S., Self-supervised monocular depth estimation with internal feature fusion, arXiv preprint arXiv:2110.09482, 2021.
- [43] Geiger, A., Lenz, P., Stiller, C., Urtasun, R., Vision meets robotics: The kitti dataset., The International Journal of Robotics Research., 2013, 32(11): 1231–1237.
- [44] Saxena, A., Sun, M., Ng, A.Y., Make3d: Learning 3d scene structure from a single still image., IEEE Transactions on Pattern Analysis and Machine Intelligence, 2008, 31(5): 824–840.
- [45] Eigen, D., Puhrsch, C., Fergus, R., Depth map prediction from a single image using a multi-scale deep network., Advances in neural information processing systems, 2014, 27.
- [46] Wang, C., Buenaposada, J.M., Zhu, R., Lucey, S., Learning depth from monocular videos using direct methods., Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, 2022–2030.



Yingwei Feng was born in Jiangmen, Guangdong, China, in August, 1980. He received the bachelor's degree in computer science and technology from Wuyi University, in 2013, where is currently pursuing the postgraduate degree. His research interests include robot navigation and positioning. <https://orcid.org/0009-0006-9649-7024>



Zhiyong Hong is a professor in College of Electronic and Information Engineering, Wuyi University. He received his Ph.D. degree in computer science and technology from Southwest Jiaotong University, Chengdu, China, in 2014, and received the B.S. and M.S. degrees in computer science and technology from the Shenyang Institute of Technology, Shenyang, China, in 2001 and 2004, respectively. His research interests include intelligent information processing, AI, block chain, and rough set theory. <https://orcid.org/0000-0002-5409-2962>



Liping Xiong was born in Yichun City, Jiangxi Province, China, in 1985. He received the Ph.D. degree from Sun Yat-sen University, Guangzhou, China, in 2016. Now, he has been a lecturer in the College of Electronic and Information Engineering, Wuyi University, China. His research interests

include knowledge representation and reasoning, formal methods, and machine learning.
<https://orcid.org/0000-0001-7924-8655>



Zhiqiang Zeng was born in Shao-guan, Guangdong, China, in 1989. He received the Ph.D. degree from the South China University of Technology, Guangzhou, China, in 2018. Since then, he has been a professor with the College of Electronic and Information Engineering, Wuyi University, China.

He is the author of a number of high-level articles. His research interests include machine learning, data mining, industrial big data, and production scheduling.
<https://orcid.org/0000-0002-9544-5605>



Jingmin Li was born in Jiangmen, Guangdong, China, in 1978. He received his master's degree from Huazhong University of Science and Technology, China, in 2004. Since 2007, he has been a lecturer at the School of Computer Science and the College of Electronic and Information Engineering at Wuyi University,

China. His research interests include big data analysis, cloud computing, data mining, and deep learning.
<https://orcid.org/0009-0008-7839-4456>