Complex & Intelligent Systems
https://doi.org/10.1007/s40747-024-01575-0

ORIGINAL ARTICLE

Repmono: a lightweight self-supervised monocular depth estimation architecture for high-speed inference

Guowei Zhang[1] · Xincheng Tang[1] · Li Wang[2] · Huankang Cui[1] · Teng Fei[1] · Hulin Tang[1] · Shangfeng Jiang[1]

Received: 30 March 2024 / Accepted: 21 July 2024

Abstract

Self-supervised monocular depth estimation has always attracted attention because it does not require ground truth data. Designing a lightweight architecture capable of fast inference is crucial for deployment on mobile devices. The current network effectively integrates Convolutional Neural Networks (CNN) with Transformers, achieving significant improvements in accuracy. However, this advantage comes at the cost of an increase in model size and a significant reduction in inference speed. In this study, we propose a network named Repmono, which includes LCKT module with a large convolutional kernel and RepTM module based on the structural reparameterisation technique. With the combination of these two modules, our network achieves both local and global feature extraction with a smaller number of parameters and significantly enhances inference speed. Our network, with 2.31MB parameters, shows significant accuracy improvements over Monodepth2 in experiments on the KITTI dataset. With uniform input dimensions, our network's inference speed is 53.7% faster than R-MSFM6, 60.1% faster than Monodepth2, and 81.1% faster than MonoVIT-small. Our code is available at https://github.com/txc320382/Repmono.

Keywords Depth estimation · Large convolutional · Structural reparameterisation · Inference speed

Introduction

Depth estimation, as a fundamental task in the field of computer vision, has been widely used in areas such as autonomous driving [1], robot navigation [2, 3] and virtual reality [4]. Depth estimation methods can primarily be divided into two types: active ranging and passive ranging. Active ranging relies on distance-measuring sensors to acquire depth information. These sensors mainly include costly LiDAR (Light Detection and Ranging) and Time-of-Flight (ToF) cameras. Passive ranging techniques estimate distances by calculating disparity. Against this backdrop, the robust capability of Convolutional Neural Networks (CNNs) for image feature extraction has greatly facilitated the advancement of monocular depth estimation technologies based on deep learning [5]. In supervised monocular depth estimation, the training process requires the use of accurate ground truth depth data. However, in complex environments, the unpredictability of these data makes their collection particularly challenging. Self-supervised monocular depth estimation methods employ synchronized stereo image pairs or monocular videos for training. Although training with monocular videos necessitates an additional pose estimation network to calculate the camera's motion, it requires only a single camera for data collection. Therefore, the use of monocular videos in self-supervised monocular depth estimation remains widely adopted.

Visual Transformer, with its ability to model the global sensory field, continues to make breakthroughs in the visual domain [6], and the use of Transformer in self-supervised depth estimation is also being attempted. For example, MonoVIT [7] utilizes advanced Transformer block in its encoder to achieve accurate prediction of fine-grained depth features, overcoming the limited receptive field of CNNs. However, multi-head self-attention modules inside the Trans-

Guowei Zhang, Xincheng Tang and Li Wang contributed equally to this work.

✉ Shangfeng Jiang
zwcv1005@126.com

[1] School of Mechanical and Automotive Engineering, Xiamen University of Technology, 600 Ligong Road, Xiamen 361024, Fujian, China

[2] Research and Development Department, Shunfeng Technology Co., Ltd., Xuefu Road, Shenzhen 518000, Guangdong, China

Published online: 10 August 2024

former make it difficult to achieve fast inference due to its complex parallel operations. Comparing to using CNN in the architecture, MT-SFMLearner [8] demonstrates that using Transformer in the architecture has higher robustness, but also brings higher parameters and lower inference speed. Lite-Mono [9] introduces Continuous Dilation Convolution (CDC) module and Local–Global Feature Interaction (LGFI) module to lighten the hybrid architecture of CNN and Transformer, however it still retains the core module of multi-head self-attention to capture global features. While several studies have existed designing Transformer with CNN in a self-supervised monocular depth estimation architecture, the fast inference of the model is neglected in order to extract rich details. We believe that using Transformer in the architecture of self-supervised monocular depth estimation will affect the computational cost and inference speed. Pursuing higher performance while bringing more computational complexity and slower inference speed is an outcome we do not wish to see, as it would limit the network's capability in practical applications [10, 11]. Exploring how to simultaneously achieve excellent performance and fast reasoning of the network becomes the main focus of our research (Fig. 1).

In order to solve the above problems, this study designs a lightweight and efficient self-supervised monocular depth estimation using a pure CNN architecture. Drawing on the idea of Mateformer [12], we design Large Convolution Kernel Transformer (LCKT) module for multi-scale feature extraction, starting with token mixing and channel mixing, and enhance local information extraction through the use of the simple yet efficient Senet [13] attention mechanism. RepTM module proposed in this study reparameterizes deep convolutions and achieves feature extraction of global information and local details when used in conjunction with the LCKT module, which effectively improves the inference speed without sacrificing model performance. Our contributions in this paper can be summarized in the following three aspects:

(1) We propose a self-supervised monocular deep estimation network called Repmono, which is capable of high speed inference while maintaining high performance.
(2) We carefully design LCKT module with a large convolutional kernel and RepTM module based on the structural reparameterization technique. Experiments show that the LCKT module is able to achieve effective local and global

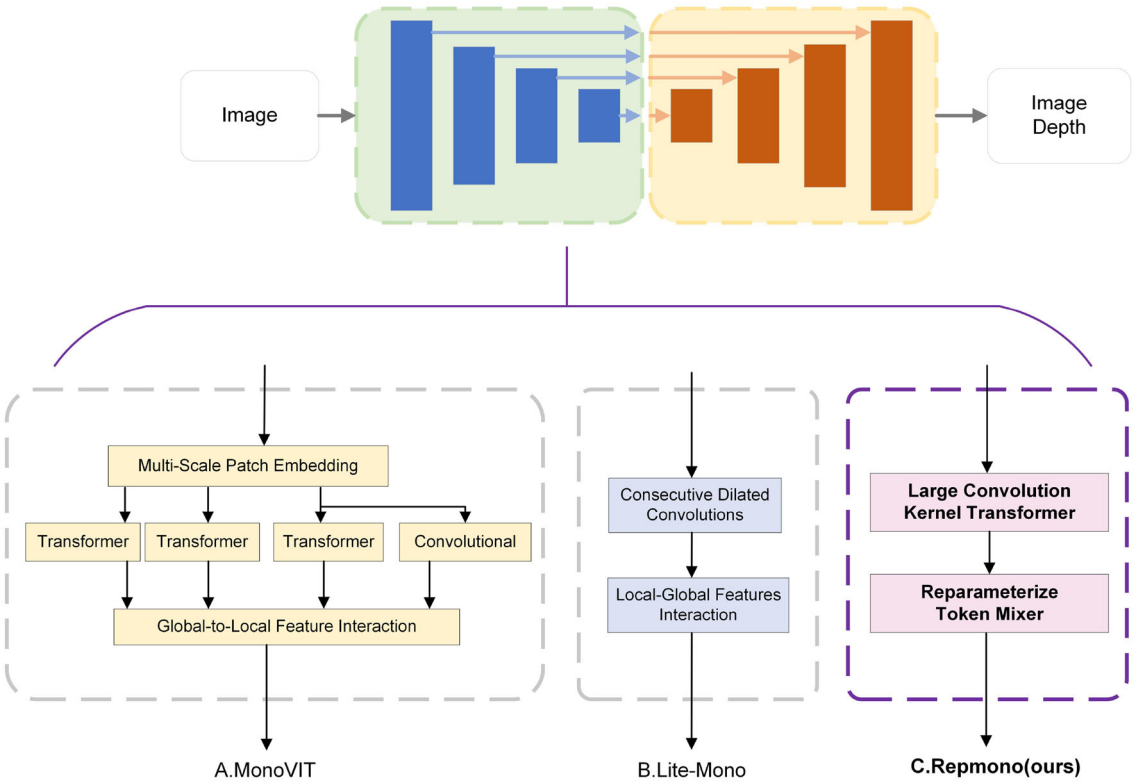

Fig. 1 Comparison of Monovit [7], Lite-Mono [9] and our proposed Repmono. **A** MonoVIT [7] combines the VIT model with self-supervised depth estimation. **B** Lite-Mono [9] introduces continuous dilated convolution modules and local–global feature interaction modules. **C** Repmono proposes a large convolutional kernel feature extraction module and a reparameterized token mixer module

feature extraction, and RepTM module is able to further optimize the efficiency of the network while extracting detailed features.

(3) Evaluations on the publicly available KITTI dataset [14] show that our lightweight model has fewer parameters and higher accuracy. We also explore the generalization capability of our network architecture on the Make3D dataset [15] and the DrivingStereo dataset [16], and our model also exhibits better generalization performance compared to other lightweight models.

Related work

Monocular depth estimation

Because different scales of 3D scenes can be inferred from 2D images, depth estimation from a single image has always been a challenging issue. Supervised depth estimation utilizes real depth maps as supervision signals, enabling accurate estimation of depth from a single RGB image. Eigen et al. [17] applied deep networks to depth estimation for the first time, using a multi-scale network structure to extract global and local depth features of the input image. Subsequent studies continuously improves deep networks [18–20] to achieve better performance, but these studies require ground truth data in the real world, which has been a challenge. Garg et al. [21] implicitly learn depth by using a reprojection of a stereo image, where the loss function is a pixel based reconstruction loss, representing a novel approach to view synthesis. Godard et al. [22] propose monodepth based on this, incorporating left-right consistency loss to ensure consistency in depth prediction between left and right images. However the stereo images required for training of such studies are also difficult to obtain. The above studies try to get rid of ground truth data using new image reconstruction loss techniques, but inevitably use stereo images for self-supervised training, and thus self-supervised depth estimation using exclusively unlabeled ground truth depths during training is gradually becoming a viable alternative. SFMLearner proposed by Zhou et al. [23], can learn both depth and self-motion from consecutive frames of monocular video, but it cannot remove the loss introduced to dynamic objects in consecutive frames. For the treatment of dynamic objects, Vijayanarasimhan et al. [24] learn multiple object masks in a deep network. Guizilini et al. [25] combine semantic information with depth estimation information to reduce the luminosity loss due to dynamic objects, and the GeoNet model proposed by Zhichao Yin et al. [26] introduce optical flow estimation to predict the sequence of images dynamic objects in an image sequence. The above study chose to include additional tasks in the deep network to minimize the loss caused by dynamic objects, but this also increases the parameters in the network model. Monodepth2 proposed by Godard et al. [27] use an automatic masking loss to remove dynamic objects that are at the same speed as the camera, and a minimum reprojection loss is designed to deal with occlusion that occurs in the front and back frames without the need to use additional learning tasks. Therefore, the model proposed in this study follows the self-supervised training strategy based on Monodepth2 [27].

Network architecture for depth estimation

The network architecture in monocular depth estimation has a significant impact on the final depth prediction results. Prior to the application of Transformer to vision, most deep learning efforts on monocular depth estimation focused on the design of CNN architectures such as Resnet [28], VGGnet [23], HRnet [29] and Packnet [30]. These classical networks have achieved remarkable results in the application of self-supervised monocular depth estimation tasks. However, CNN models are limited by their finite receptive fields during convolution operations. By introducing attention modules, networks can more effectively fuse features and extract depth features. For example, the literature [31] improved feature fusion capability by incorporating an attention module, while the literature [32] used self-attention to enhance semantic features in a VGG [33] encoder. R-MSFM [34], a small architecture that employs a feature modulation module to learn multi-scale features, uses only the first three stages of ResNet18 [28] as a backbone network in order to reduce the number of model parameters. PydNet [35] designed an unsupervised network capable of performing depth estimation on the CPU, although with lower computational parameters, it cannot achieve more accurate depth feature extraction. FastDepth [36] model achieves fast inference through pruning and optimization, but has limitations in dealing with the details of depth estimation. Exploring how to achieve both excellent performance and fast inference in the network has become the main focus of our research.

The network architecture in monocular depth estimation has a significant impact on the final depth prediction results. Prior to the application of Transformer to vision, most deep learning efforts on monocular depth estimation focused on the design of CNN architectures such as Resnet [28], VGGnet [23], HRnet [29] and Packnet [30]. But these models ignore parameters and inference speed. R-MSFM [34] achieves multi-scale feature learning through feature modulation modules, which uses only the first three stages of ResNet18 [28] as its backbone, offering high efficiency while lacking in deep feature extraction. PydNet [35] designs an unsupervised network capable of performing depth estimation on the CPU, which cannot extract rich hierarchical features despite the low number of parameters. FastDepth [36] model achieves fast inference through pruning and optimization, but has

limitations in dealing with the details of depth estimation. The above network are continuously improving the accuracy of depth estimation models, however, CNN models are unable to have a global receptive field when performing convolutional operations, thus making it difficult to retain to specific detailed features. When Transformer is applied in the visual domain, some studies put Transformer into the network architecture to enhance the model performance. For example, [31] uses self-attention to enhance semantic features in a VGG encoder, and MT-SFMLearner [8] points out that while Transformer-based deep estimation architectures have better robustness compared to other CNNs, they hamper operational efficiency. MonoVIT [7] combines convolution with the Transformer module to retain more detailed features, but with a higher parameter, The use of the core module of MSHA in Lite-Mono [9] hinders the fast inference of the model.

Structural reparameterisation techniques

The structure reparameterisation technique was initially proposed and applied to VGG [33] architectures by Ding et al. [29], and its core advantage lies in fully leveraging the network performance while accelerating the inference speed of the network architecture. To enhance network performance, the network employs a multi-branch structure during training, which includes a 3×3 convolution layer branch, a 1×1 convolution layer branch, and an identity mapping branch. During inference, the network transforms these multi-branches into a single-branch structure, where the identity mapping branch is treated as a 1×1 convolution, and the 1×1 convolution can be transformed into a 3×3 convolution by padding with zeros. Based on the linear additivity of convolution, the resulting 3×3 convolution is obtained by summing the three bias vectors. When trying to use RepConv [37] in different positions of the model architecture, we observed that this method did not yield optimal network performance, indicating that the proposed reparameterisation model is not directly suitable for architectures in the field of depth estimation. Therefore, we have designed a new reparameterisation module that integrates well with existing state-of-the-art depth estimation architectures, which provides solution ideas for achieving high network performance and efficient inference.

The structure reparameterisation technique was initially proposed and applied to VGG [33] architectures by Ding et al. [29], and its core advantage lies in fully leveraging the network performance while accelerating the inference speed of the network architecture. To enhance network performance, the network adopts a multi-branch structure during training, which includes a 3×3 convolution layer branch, a 1×1 convolution layer branch, and an identity mapping branch. During inference, the network transforms these multi-branches into a single-branch structure, where the identity mapping branch is treated as a 1×1 convolution, and the 1×1 convolution can be transformed into a 3×3 convolution by padding with zeros. But given the growth of model parameters, RepVGG [37] is difficult to apply to different types of network architectures. DBB et al. [38] explore six reparameterization methods based on this foundation, which are able to improve the performance of the network model but do not speed up the inference of the model. [29] have used deep convolutions combined with other pointwise convolutions to improve the inference speed of models, reducing the overall number of parameters, but also resulting in a decrease in overall network performance. To the best of our knowledge, the aforementioned reparameterization models are not directly applicable to the architectures in the field of depth estimation. Therefore, we design a new reparameterization module that integrates well with existing state-of-the-art depth estimation architectures, providing a solution for achieving high-performance networks and fast inference.

Proposed framework

Motivation for the design

The encoder in monocular depth estimation network architectures is continuously being improved. Current network architectures focus more on the details of image depth estimation, while also neglecting model size and inference speed. Due to the limited receptive field of the CNN architecture, it is difficult to effectively extract the global information from image. Transformer architecture is known for its powerful extraction of contextual information and also achieves high accuracy, but its unique parallel structure increases model size and limits speed. Mateformer [12] demonstrates the importance of the network architecture in Transformer, and points out that the self-attention module is not all that is needed, testing the plausibility of this in various experiments. Inspired by this, we attempt to design a framework specifically for monocular depth estimation, aiming to achieve both high accuracy and high inference speed. Eventually, we adopt the designed LCKT and RepTM on the encoder-decoder architecture. In LCKT, we employ a large 7×7 convolutional kernel to expand the receptive field. Our proposed RepTM achieves spatial information fusion of token mixer through deep convolution, and channel mixer achieves the inter-channel information interaction through 1×1 dilated convolution and 1×1 projection layer. Compared with multi-head attention, LCKT combined with RepTM can also capture both local and global features, and it is beneficial for accelerating inference. The architecture details is described in detail below.

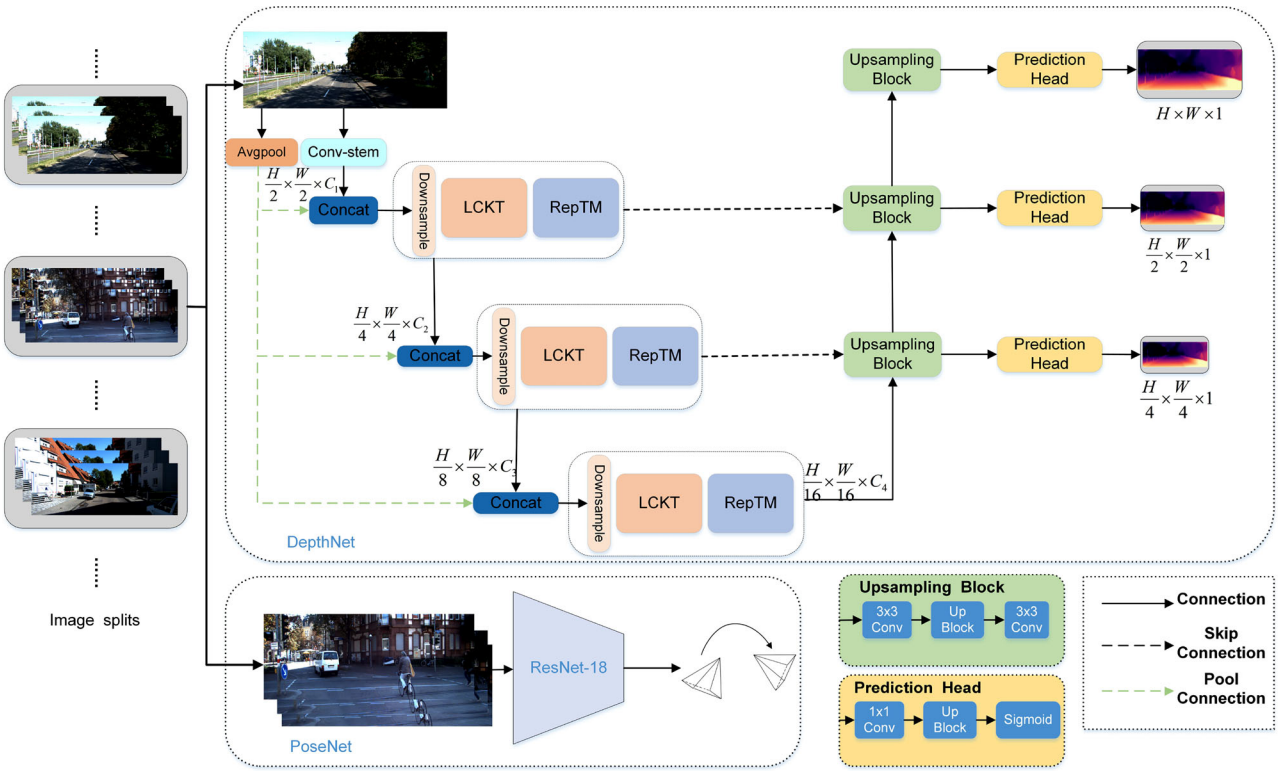

Fig. 2 Overview of our repmono framework. Repmono as a whole is divided into two parts, DepthNet and PoseNet. The depth network encoder uses a large convolutional kernel feature extraction module and a reparameterized token mixer module to extract rich deep features while speeding up inference. PoseNet uses the same as in previous works [23, 31, 39] to estimate the pose between neighboring frames of a monocular image

Deep network architecture

Depth encoder. The encoder-decoder architecture of DepthNet is able to extract features efficiently, as demonstrated in previous work [23, 27]. As shown in Fig. 2, the proposed architecture is divided into four stages. Except for the first stage, all subsequent stages use the same modules. The input image first enters Stage 1, where local features are extracted through a 3×3 convolution with a stride of 2, generating feature maps with dimensions $\frac{H}{2} \times \frac{W}{2} \times C_1$. Stage 2 consists of the downsampling layer, LCKT module, and RepTM module. In this stage, the input consists of the concatenation of the feature maps from the previous stage and the feature maps obtained by average pooling of the original image. This structure aims to compensate for the spatial information loss caused by downsampling, resulting in feature maps of size $\frac{H}{4} \times \frac{W}{4} \times C_2$. The third and fourth stages use the same structure, and their inputs also receive feature maps obtained by average pooling of the original image. The feature maps $\frac{H}{8} \times \frac{W}{8} \times C_3$ and $\frac{H}{16} \times \frac{W}{16} \times C_4$ are generated by downsampling the layers, respectively.

Large convolution kernel transformer (LCKT). As shown in Fig. 3, the Transformer mainly consists of two parts. One part is a token mixer module based on the self-attention mechanism and the other part is a channel MLP module. Previous research [12] found that performance competitive with the original Transformer could still be maintained across multiple computer vision tasks by replacing the self-attention mechanism in the token mixer with a simpler spatial pooling operation. Therefore, they regarded the self-attention in the token mixer as a specific token mixer, collectively referred to as MetaFormer [12]. To accelerate the inference speed, we replace the token mixer with a simple and efficient SENet [13] module and use batch normalisation across the network to improve stability. Especially in channel MLP module, we introduce depthwise dilated convolution, inspired by the design of the CDC module [9]. By using dilated convolutions with different dilation rates in different stages, feature multi-scale fusion is achieved. Dilated convolutions can be defined by the formula:

$$y[a] = \sum_{w=1}^{W} x[a + r \cdot k]h[w] \tag{1}$$

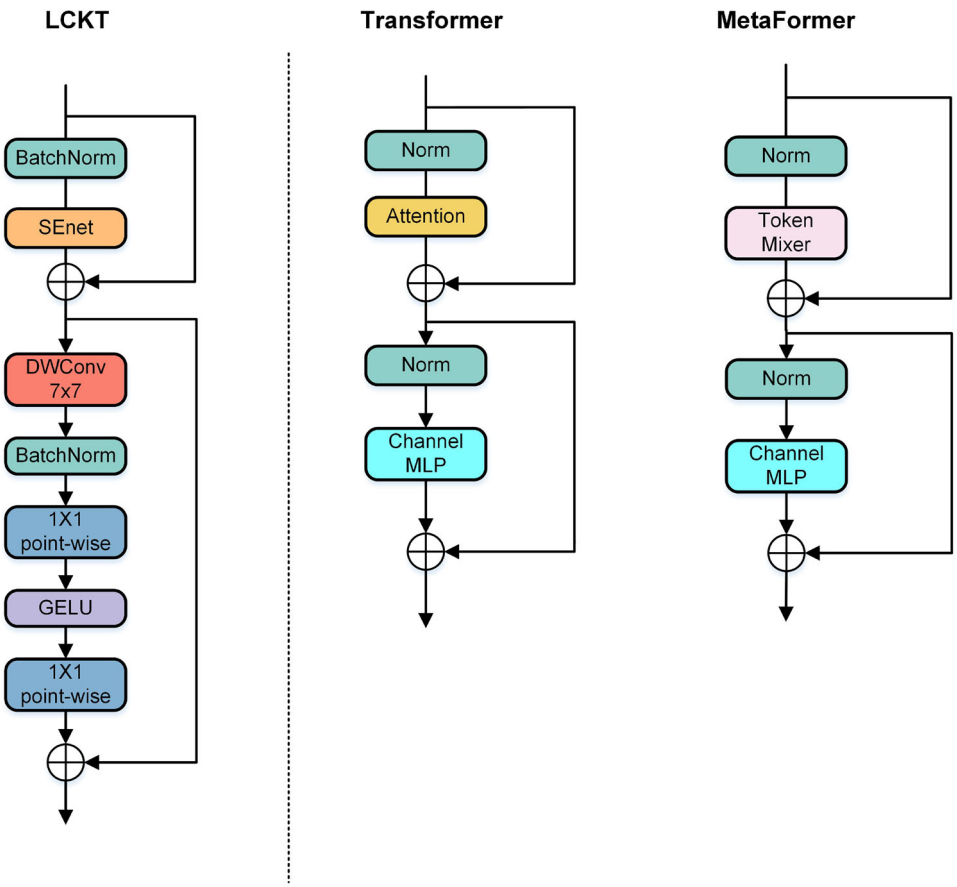

Fig. 3 LCKT design concept. The token mixer of LCKT is senet and the channel MLP module consists of 7 × 7 large convolution kernel depth convolution, batch normalization module (BatchNorm), 1 × 1 point-by-point convolution and GELU activation function

where the input is x[a], $h[w]$ is a filter of length k, and r represents the dilation rate, without changing the size of the convolution kernel when r=1. [31] proposes that using deep large convolutions in the network is competitive with the use of a self-attention variant, but leads to a modest increase in inference. We change the initial convolution kernel from 3×3 to 7×7. Although dilated convolutions can increase the receptive field through the dilation rates, their computation is based on sparse sampling. After dilation, the convolution kernel loses continuity, and the number of sparse samples after dilation of a large convolutional kernel is significantly greater than that of a small convolution kernels, which may affect the continuity of information. In order to fuse the features efficiently in the depth direction and considering the importance of information continuity, we use point-by-point convolution to perform convolution operation on each pixel of the feature map, and then perform a weighted combination in the depth direction. This design aims to work in tandem with large kernels to optimize model performance. Experimental results show that using 7 × 7 large convolution kernel can significantly improve the model performance. Compared to a 5 × 5 convolution, the average error decreases by 6.4%, and compared to a 3 × 3 convolution, the average error decreases by 9.4%. This finding emphasizes the potential of using large convolution kernels in improving model performance and robustness.

Reparameterize token mixer (RepTM). As demonstrated in Fig. 4, we replace the token mixer with elements from the lightweight network MobileNet, particularly a 3 × 3 depthwise convolution and a batch normalisation (BatchNorm) layer. The use of this depthwise convolution aims to facilitate the efficient fusion of spatial information within the model. To enhance the flexibility and efficiency of the model, a residual branch and a branch combining 1 × 1 depthwise convolution with BatchNorm layer are concurrently incorporated into the token mixer. According to the linearity of convolution's additivity formula:

$$\begin{aligned} &Conv(x,\omega1)+Conv(x,\omega2)+Conv(x,\omega3)\\ &\quad=Conv(x,(\omega1+\omega2+\omega3)) \end{aligned} \tag{2}$$

Where denotes $Conv(x,\omega)$ the convolution operation with input feature x, convolution kernel ω, where $\omega1$, $\omega2$ and

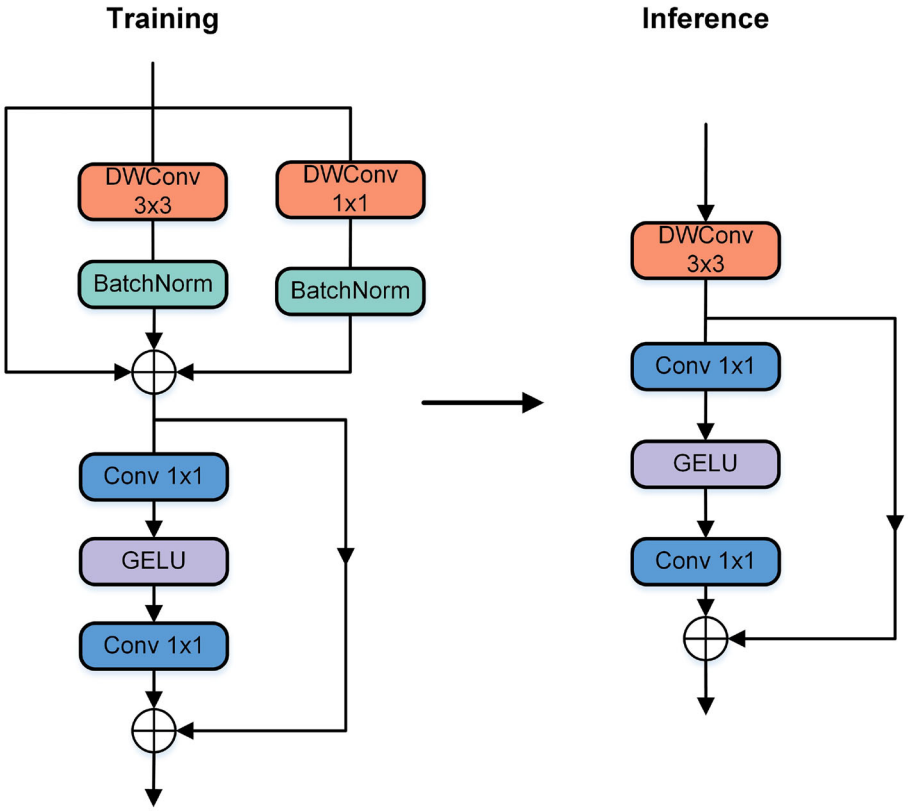

Fig. 4 Equivalent diagram of RepTM module structure. Where the multi-branch structure during training is converted to a single-branch structure during inference

$\omega 3$ are convolution kernels of the same size. This structural parameterization design endows the network enhanced feature extraction capability during the training phase, while the multi-branch structure can be converted into a single 3×3 depthwise convolution during the inference phase. This conversion effectively reduces the computational and storage costs introduced due to the parallel structure of the network, thereby improving the computational efficiency of the model during inference. For channel MLP module, the design primarily uses two 1×1 convolutions coupled with GELU activation function, facilitating inter-channel information exchange and further boosting the model's performance.

Depth encoder. We use a bilinear upsampling technique to increase the spatial dimensionality of the model, which is consistent with the approach used in Lite-Mono [9], and integrate the features of the three phases in the encoder through a convolutional layer. After each upsampling block, we set up the prediction header to output inverse depth maps at full resolution, $\frac{1}{2}$ resolution and $\frac{1}{4}$ resolution for depth prediction at different accuracy levels.

PoseNet. This study follows the framework established in prior works [23, 31, 39], utilizing a pretrained lightweight ResNet18 [28] to construct PoseNet. The system processes spliced colour image pairs $[I, I']$ and accurately estimates the 6-degree-of-freedom relative pose between adjacent frames by means of a three-layered convolutional bit-pose decoder.

Self-supervised learning

Our task is to infer a depth map from a single RGB image alone in the absence of actual depth information. In this process, a depth estimation network generates a depth map D_t based on a given input image I_t. Simultaneously, the pose estimation network handles temporally adjacent images, computing the relative pose $T_{t \longrightarrow t'}$ from the target image I_t to the source image $I_{t'}$ (where t' is the previous or subsequent frame of t). The depth map D_t and the pose $T_{t \longrightarrow t'}$ are used as supervised signals for efficient training and learning.

Photometric consistency loss. Following previous research [23], we model the learning objective with the aim of minimizing the image reconstruction loss Lp between the original image I_t and the synthetic target image $I_{t'}$, Lp is defined:

$$Lp = \min_{t'} F(I_t, I_{t' \longrightarrow t}) \tag{3}$$

where $F()$ in Eq. (3) represents the photometric reconstruction error, $I_{t' \longrightarrow t}$ denotes the warp result from image $I_{t'}$ to image I_t.

$$I_{t' \to t} = I_{t'} \langle \text{proj}(D_t, T_{t' \to t}, K) \rangle \tag{4}$$

where *proj*() represents the two-dimensional coordinates obtained by projecting the depth map D_t onto the image $I_{t'}$, where $\langle\rangle$ is the sampling operator, K is the image-identical camera-internal reference matrix, and we sample the images using differentiable bilinear [40] sampling.

$$F(I_a, I_b) = \frac{\alpha}{2}((1 - \text{SSIM}(I_a, I_b)) + (1 - \alpha)\|I_a - I_b\|) \tag{5}$$

the original image is set to I_a, and the reconstructed image obtained by Eq. (4) is set to I_b, $F(I_a, I_b)$ represents the weighted sum between the Structural Similarity Index Measure (SSIM) [41] and the intensity difference term, where α is set to 0.85 empirically [27].

Minimum photometric loss. For the occlusion phenomenon in the source image, the minimum photometric loss [27] is computed for each pixel among the losses between adjacent frames in both forward and backward directions.

$$L_{\text{SS}}(I_s, I_t) = \min_{i \in [-1,1]} F(I_a, I_b) \tag{6}$$

where I_s denotes the previous or next frame of the target image, $i \in [-1, 1]$ indicates that the image range is forward and backward neighboring frames.

Automatic mask. For dynamic objects in the image, automatic mask [27] is used to filter them out, ensuring that the objects are stationary relative to the camera. Here we denote it by u.

$$u = \min_{i \in [-1,1]} L_{\text{SS}}(I_s, I_t) > \min_{i \in [-1,1]} L_{\text{SS}}(I_a, I_b) \tag{7}$$

Smoothness loss. In addition, in order for the inverse depth map not to shrink arbitrarily, an edge-aware smoothness loss [27, 29] is utilized.

$$L_{\text{smooth}} = |\partial_x d_t^*| \cdot e^{-|\partial_x I|} + |\partial_y d_t^*| \cdot e^{-|\partial_y I|} \tag{8}$$

where ∂_x is the gradient operator in the x direction, ∂_y is the gradient operator in the y direction, and $d_t^* = \frac{d_t}{\hat{d}_t}$ denotes the mean normalised inverse depth.

Final total loss. The final total loss L is composed of the total image reconstruction loss $uL_{\text{SS}}(I_s, I_t)$ and the smoothness loss $\lambda \cdot L_{\text{smooth}}$.

$$L = \frac{1}{3}\sum_{i=1}^{3}(uL_{\text{SS}}(I_s, I_t) + \lambda \cdot L_{\text{smooth}}) \tag{9}$$

where u is the Automatic mask, $L_{\text{SS}}(I_s, I_t)$ is the minimum luminosity loss, λ weights the smoothness term, we set it to 0.001, which are output from three sizes for final fusion to full resolution.

Experiments

Datasets

KITTI. The KITTI dataset [14] is a stereo vision dataset containing 61 scenes, mainly used for stereo imaging studies. It collects images with dimensions of 1242 × 375, which were captured by a stereo camera system on a LiDAR-equipped vehicle. This study is based on previous studies in the field [23, 27, 29] and uses the image segmentation scheme defined by Eigen et al [17], which consists of 39,810 sets of monocular triple images used for training and 4,424 sets of images used for validation. We evaluated the single-view depth performance on a custom test [19] set, using both the original LiDAR data (697 images in total) and the modified real labels [42] (652 images in total).

Make3D. The Make3D dataset [15] contains mainly images of outdoor environments and is often used to test the generalisation ability of monocular depth estimation frameworks. We tested the Repmono model using the same image preprocessing steps and evaluation criteria as in [27].

DrivingStereo. The DrivingStereo dataset [16] is a large-scale stereo dataset containing real-world driving scenes. The dataset includes a variety of scenes, and the classification of images under different weather scenarios is provided in the official website, and each class of images consists of 500 frames. We use the DrivingStereo dataset containing four types of image scenes: sunny, rainy, cloudy, and foggy. We use this dataset to evaluate our generalization capability in specific driving scenarios.

Implementation details

In our experiments, the model is implemented based on the Pytorch framework and trained on a server equipped with an NVIDIA A100. Both depth and pose networks are pretrained on ImageNet [27, 34]. The model employs AdamW [34] as the optimizer, with the training period set for 30 epochs, a batch size of 20, and an initial learning rate of 0.0001. For images with resolutions of 640 × 192 and 1024 × 320, the model training was conducted on a single GPU with 40 G of memory. Training the network for 640 × 192 images take

Table 1 Comparison of Repmono with some recent representative methods on the KITTI benchmark using the Eigen split

Method	Data	Resolution	Depth error				Depth accuracy			Model size params (MB)
			Abs Rel	Sq Rel	RMSE	RMSE log	δ_1	δ_2	δ_3	
GeoNet [26]	M	640 × 192	0.149	1.060	5.567	0.226	0.796	0.935	0.975	31.6
DDVO [43]	M	640 × 192	0.151	1.257	5.583	0.228	0.810	0.936	0.974	28.1
Struct2depth [44]	M	640 × 192	0.141	1.026	5.291	0.215	0.816	0.945	0.979	66.8
SGDepth [45]	M+Se	640 × 192	0.113	0.835	4.693	0.191	0.879	0.961	0.981	16.3
Monodepth2 [27]	M	640 × 192	0.115	0.903	4.863	0.193	0.877	0.959	0.981	14.3
R-MSFM3 [46]	M	640 × 192	0.114	0.815	4.712	0.193	0.876	0.959	0.981	3.5
R-MSFM6 [46]	M	640 × 192	0.112	0.806	4.704	0.191	0.878	0.960	0.981	3.8
Lite-Mono-small [9]	M	640 × 192	0.110	0.802	4.671	0.186	0.879	0.961	0.982	2.5
Repmono (ours)	M	640 × 192	0.107	0.804	4.641	0.184	0.885	0.961	0.982	2.31
Monodepth2 [27]	M*	640 × 192	0.132	1.044	5.142	0.210	0.845	0.948	0.977	14.3
R-MSFM3 [46]	M*	640 × 192	0.128	0.965	5.019	0.207	0.853	0.951	0.977	3.5
Lite-Mono-small [9]	M*	640 × 192	0.123	0.919	4.926	0.202	0.859	0.951	0.977	2.5
Repmono (ours)	M*	640 × 192	0.123	0.946	5.017	0.209	0.851	0.947	0.978	2.31
Monodepth2 [27]	M	1024 × 320	0.115	0.882	4.701	0.190	0.879	0.961	0.982	14.3
Sun [47]	M	1024 × 320	0.110	0.791	4.557	0.184	0.887	0.964	0.983	–
R-MSFM3 [46]	M	1024 × 320	0.112	0.773	4.581	0.189	0.879	0.960	0.982	3.5
R-MSFM6 [46]	M	1024 × 320	0.108	0.748	4.470	0.185	0.889	0.963	0.982	3.8
HR-Depth [39]	M	1024 × 320	0.106	0.755	4.472	0.181	0.892	0.966	0.984	14.7
Lite-Mono-small [9]	M	1024 × 320	0.103	0.757	4.449	0.180	0.894	0.964	0.983	2.5
Repmono (ours)	M	1024 × 320	0.103	0.731	4.446	0.178	0.893	0.984	0.983	2.31

The input image sizes are divided into 640×192 and 1024×320, with the best results underlined. For the performance metrics Abs Rel, Sq Rel, RMSE and RMSE log lower is better, for δ_1, δ_2, δ_3 higher result scores are better. In the "Data column", "M": monocular video, "M*": monocular without pretraining, "M+Se ": monocular video + semantic segmentation, all models except M* are pretrained on ImageNet [34]. Depth estimation is scaled using the median finger of the true laser information during testing

approximately 8 h, while for 1024 × 320 images, it take about 12 h and 30 min. In our experiments, we use the same data enhancement strategy as in previous studies [27]. For the final evaluation of the model's effectiveness, we use seven metrics widely used in the field of depth estimation, including Absolute Relative Difference(Abs Rel), Squared Relative Difference(Sq Rel), Root Mean Square Error (RMSE), Root Mean Square Logarithmic Error(RMSE Log), as well as three accuracy metrics ($\delta_1 < 1.25$, $\delta_2 < 1.25^2$, $\delta_3 < 1.25^3$).

KITTI results

We compare our model with existing classic and lightweight models. As shown in Table 1, we show the results of the training method using monocular video (M) and untrained monocular video (M*). For monocular videos (M), we use two sizes: low resolution (640 × 192) and high resolution (1024×320), while the untrained monocular videos are tested only at 640 × 192 resolution. The low-resolution monocular video shows that the overall size of our model is only 2.31M, which is 85% less than Monodepth2 [27] model, 40% less than R-MSFM3 model, and 16% less than Lite-Mono-small model. At the same time, our model surpasses most models in the table in terms of depth accuracy and matches the Lite-Mono-small [9] model, but exceeds Lite-Mono-small [9] in δ_1 accuracy. Figure 5 provides a more intuitive demonstration of the effectiveness of our proposed solution in depth estimation. We select lightweight network R-MSFM [46], classic network Monodepth2 with ResNet18 [28] encoder, and lightweight network Lite-Mono-small [9] with an improved self-attention mechanism in the encoder for comparison. By comparing, our solution performs better in thin structure and low-light overlapping structures. For example, in the first column of the depth map of the highway signage, depth maps from other models show dragging and incomplete structure issues in such thin structures, whereas our model's depth map performs better than other methods. In the third column of the figure is poorly distinguished from the background, our model successfully distinguishes the shape of the figure and the background, which has always been a challenging problem for monocular depth estimation. Additionally, we achieve good results for reflective glass structures. In the second column, unlike other models, our model does not exhibit depth blur or structure loss.

In the fourth column of the depth map, our model has better coherence for the car contours compared to other methods. This is mainly attributed to our use of LCKT module with a large convolutional kernel and RepTM module with reparameterized structure. The large convolution kernel dilated convolution of our LCKT module can obtain rich multi-scale information, and the RepTM module can extract more detailed features, ultimately enabling our solution to estimate depth more accurately in fine details.

Make3D results

We test our model experimentally on Make3D dataset [15] for generalisability, Our model was trained using 640×192 resolution images from the KITTI dataset [14], followed by generalization tests on Make3D [15]. For fairness, we not perform any fine-tuning on Make3D [15], and evaluation is carried out using the criteria proposed in [27]. Table 3 presents the comparison between Repmono and other lightweight models, the comparison shows that Repmono has the smallest number of parameters among all lightweight models, and performs best in all four metrics. Figure 6 shows the superiority of our scheme and that our model can model objects at different scales more accurately. Figure 5 specifically illustrates the advantages of Repmono over other lightweight networks. We select objects at varying distances in three categories of images for comparison. In the third column, the grass represents objects at a relatively close distance, in the first column, the large tree represents objects at a medium distance, and in the second column, the small tree represents objects at a relatively far distance. It can be seen that for objects at different distances in the images, the depth maps produced by Repmono exhibit the best object integrity (Table 2).

DrivingStereo results

To evaluate the generalization capability of Repmono in specific road scenarios, we test it under four weather conditions from the DrivingStereo dataset [16]. We still choose lightweight models for comparison, and all models are chosen to be trained using 1024 × 320 resolution images from the KITTI dataset [14], and then tested in sunny, cloudy, foggy, and rainy road scenarios. For fairness, all models are not fine-tuned. Table 3 shows the evaluation results of Repmono and other lightweight models under the four weather roads, and it can be seen that compared to other lightweight models, Repmono has the best performance metrics in sunny, cloudy, and foggy days, although it has the smallest number of parameters. Figure 7 specifically shows the depth maps of Repmono and other lightweight models in four weather scenarios. For close-range objects, Repmono's effect in foggy days is significantly better than that in rainy days, Repmono has the best depth maps of medium-range objects in cloudy scenarios, and it also has the best depth maps of long-range objects in sunny scenarios.

Speed of reasoning

Testing the inference speed of lightweight architectures is crucial. We conduct inference time test on NVIDIA A100 and NVIDIA 1650 for our model and other lightweight models. To ensure fairness in testing, we standardize the input dimensions to (3, 640, 192) and omit the data preprocessing before entering the model. The inference time is defined as the duration from when the data enters the encoder to when it exits the decoder. The model undergoes a "warm-up" phase before GPU inference to transition it into a working state, thus avoiding errors that could arise if the GPU shifts into power-saving mode during testing. In order to overcome the asynchronous execution of the GPU that leads to inference errors, we set the timer at the time when the GPU receives the data to start the computation and synchronise the CPU and the GPU using the synchronize function of the pytorch library. The synchronize function ensures that the computation between the CPU and GPU is properly coordinated. According to the test results in Figs. 7 and 8, our model demonstrates excellent inference performance on both GPU devices. On NVIDIA 1650, when batch size is set to 1 and 2, our model exhibits the fastest inference speed, 81% faster compared to Monovit-small [7], 60% faster compared to Monodepth2-50 [27], and 27% faster compared to Lite-Mono-small [9]. On NVIDIA A100, our model achieves the fastest inference speed when the batch size is 8. At batch sizes 16 and 32, our model's inference speed is comparable to the lightweight network R-MSFM3 [46], which demonstrates that our model maintains a fast inference speed when dealing with larger throughput (Fig. 9).

Ablation experiments on model structures

To evaluate the effectiveness of the model, we execute a series of ablation experiments focused on validating the importance of the LCKT module and RepTM module in the architecture. These experiments are performed without ImageNet [34] pre-training and tested on the KITTI dataset [14] by implementing control variables in the model. From our experiments, when only the LCKT module is used, there is an approximate increase of 2MB parameters and about a 5 ms slowdown in inference speed compared to the baseline model. However, there are improvements in key metrics such as Absolute Relative difference (Abs Rel), Squared Relative difference (Sq Rel), and Root Mean Squared Error (RMSE). When the model integrated both LCKT and RepTM modules, utilizing 7 × 7 convolutions for LCKT, the model's performance sees the most significant improvement. The

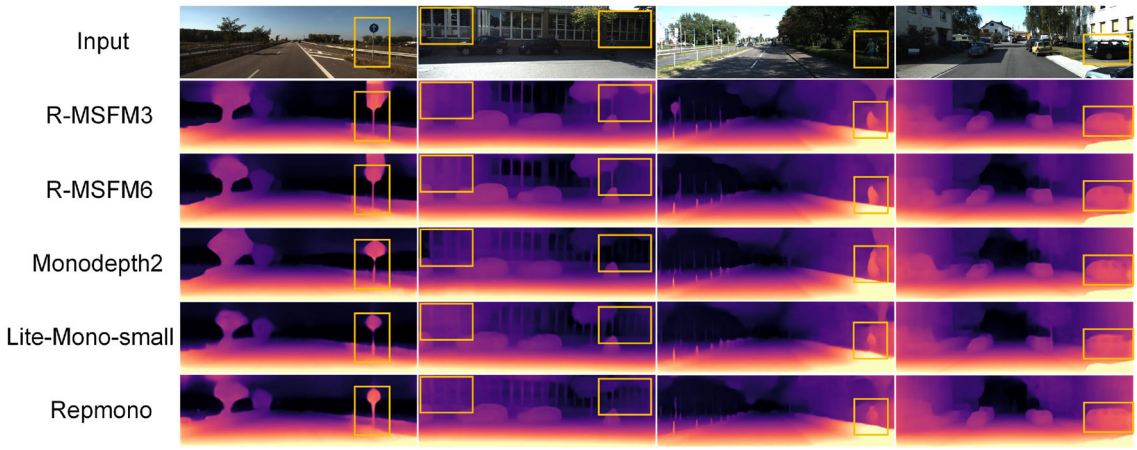

Fig. 5 Qualitative results on the KITTI [14] Eigen Split. Comparing our model with some depth maps generated by Monodepth2 [27], R-MSFM3 [46], R-MSFM6 [46], and Lite-Mono-small [9], it can be seen that Repmono better predicts thin objects, reflective mirrors, and low discrimination objects

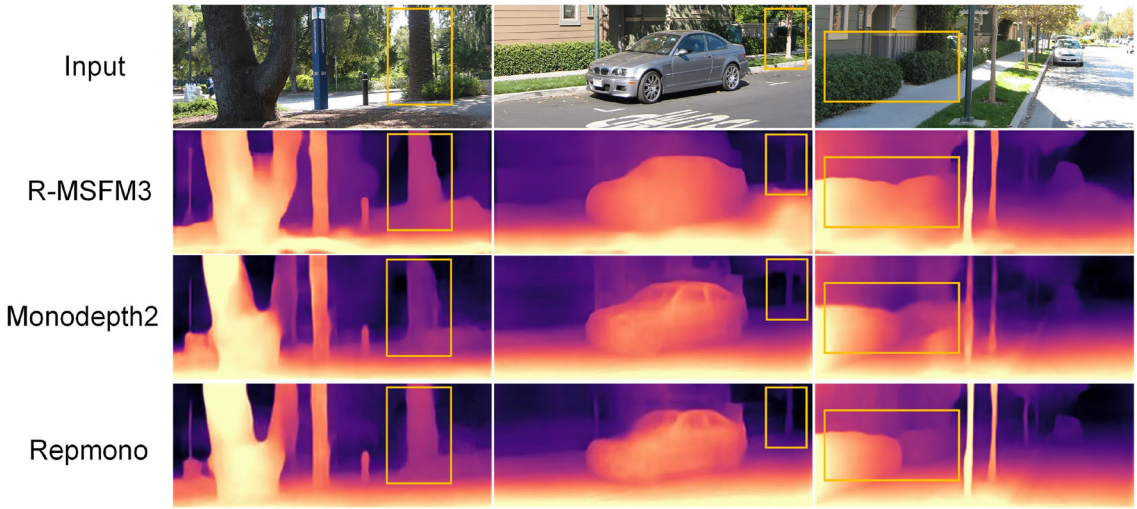

Fig. 6 Qualitative results on the Make3D dataset [15]. Comparing Repmono with Monodepth2 [27] and R-MSFM3 [46], Repmono generalises best

Table 2 Comparison of the proposed Repmono with other lightweight models on the Make3D dataset [15]

Method	Abs Rel	Sq Rel	RMSE	RMSE log	Params (MB)
DDVO [43]	0.387	4.720	8.09	0.204	28.1
Monodepth2 [27]	0.322	3.589	7.417	0.163	14.3
R-MSFM3 [46]	0.334	3.285	7.212	0.169	3.5
Repmono (ours)	<u>0.313</u>	<u>3.114</u>	<u>7.062</u>	<u>0.153</u>	<u>2.31</u>

The best results are <u>underlined</u>. Where all models are trained on KITTI [14] with an input image of 640 × 192

model's performance sees the most significant improvement: the parameters is 2.31M, inference speed is 8.89 ms, Abs Rel is reduced to 0.123, Sq Rel to 0.946, and RMSE to 5.017, while exhibiting optimal performance on accuracy metrics $\delta_1 < 1.25$, $\delta_2 < 1.25^2$, $\delta_3 < 1.25^3$. Comparing with models using 3×3 and 5×5 convolutional kernels, although inference speed is faster, they perform poorly on accuracy metrics, validating the effectiveness of 7×7 convolutions in enhancing model performance. Overall, a robust balance is achieved among model size, speed, and accuracy (Table 4).

Table 3 Comparison of Repmono with other lightweight models on the DrivingStereo Dataset [16]

Weather	Method	Depth error				Depth accuracy			Params (MB)
		Abs Rel	Sq Rel	RMSE	RMSE log	δ_1	δ_2	δ_3	
Rainy	Monodepth2	0.238	3.328	11.231	0.308	0.622	0.864	0.958	14.3
	R-MSFM6	0.231	2.986	10.632	0.291	0.646	0.865	0.964	3.5
	Lite-mono-small	0.194	2.013	8.975	0.277	0.712	0.912	0.979	2.5
	Repmono (ours)	0.212	2.231	8.893	0.263	0.703	0.907	0.969	2.31
Foggy	Monodepth2	0.118	1.387	7.532	0.175	0.857	0.963	0.983	14.3
	R-MSFM6	0.119	1.435	7.632	0.179	0.838	0.953	0.979	3.5
	Lite-mono-small	0.105	1.242	7.201	0.163	0.878	0.969	0.986	2.5
	Repmono (ours)	0.106	1.119	6.962	0.153	0.884	0.972	0.991	2.31
Cloudy	Monodepth2	0.152	1.788	6.856	0.205	0.819	0.948	0.981	14.3
	R-MSFM6	0.149	1.671	6.712	0.206	0.821	0.946	0.982	3.5
	Lite-mono-small	0.139	1.532	6.421	0.196	0.832	0.951	0.984	2.5
	Repmono (ours)	0.135	1.462	6.121	0.187	0.838	0.952	0.984	2.31
Sunny	Monodepth2	0.142	1.671	6.639	0.214	0.819	0.947	0.979	14.3
	R-MSFM6	0.151	1.721	7.094	0.248	0.812	0.942	0.978	3.5
	Lite-mono-small	0.137	1.519	6.392	0.196	0.837	0.951	0.982	2.5
	Repmono (ours)	0.131	1.479	6.216	0.192	0.839	0.952	0.982	2.31

The best results are underlined. Where all models are trained on KITTI [14] with an input image of 1024 × 320

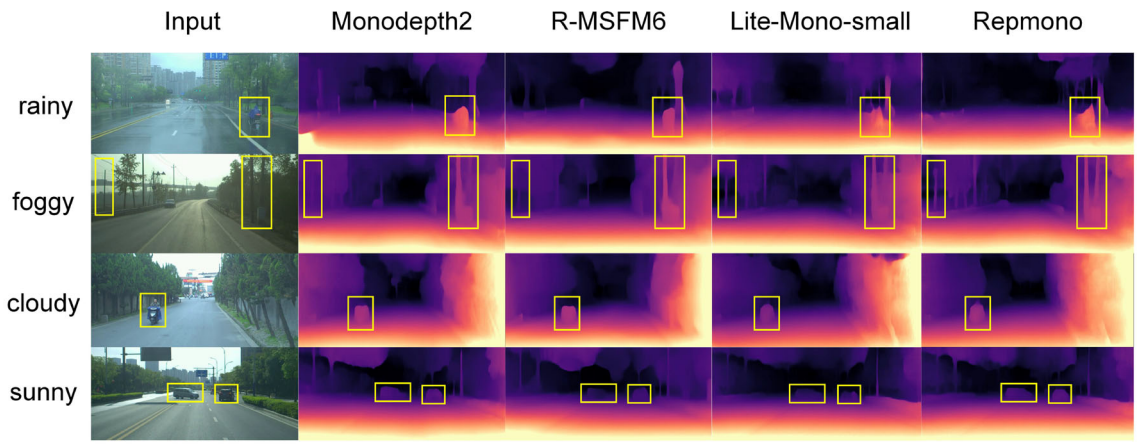

Fig. 7 Qualitative results on the DrivingStereo Dataset [16]. Compared to other lightweight models, Repmono shows superior performance under cloudy, foggy, and sunny conditions

Table 4 Model structure ablation experiments

Architecture	Kernel size			Params	Speed	Abs Rel	Sq Rel	RMSE	RMSE log	δ_1	δ_2	δ_3
	3×3	5×5	7×7	(M)	(ms)							
Baseline				0.12	3.16	0.156	1.281	5.323	0.221	0.834	0.940	0.973
Baseline+LCKT	✓	×	×	2.13	8.18	0.138	1.081	5.112	0.216	0.846	0.942	0.976
Baseline+LCKT	×	✓	×	2.15	8.32	0.134	1.009	5.151	0.211	0.839	0.945	0.977
Baseline+LCKT	×	×	✓	2.18	8.31	0.135	1.032	5.192	0.214	0.843	0.944	0.976
Baseline+LCKT+RepTM	✓	×	×	2.25	8.53	0.136	1.089	5.122	0.221	0.846	0.945	0.976
Baseline+LCKT+RepTM	×	✓	×	2.28	8.75	0.133	1.054	5.195	0.214	0.845	0.943	0.975
Baseline+LCKT+RepTM	×	×	✓	2.31	8.89	0.123	0.946	5.017	0.209	0.850	0.947	0.978

The best results are underlined. Where all models were trained on KITTI [14] with an input image of 640 × 192

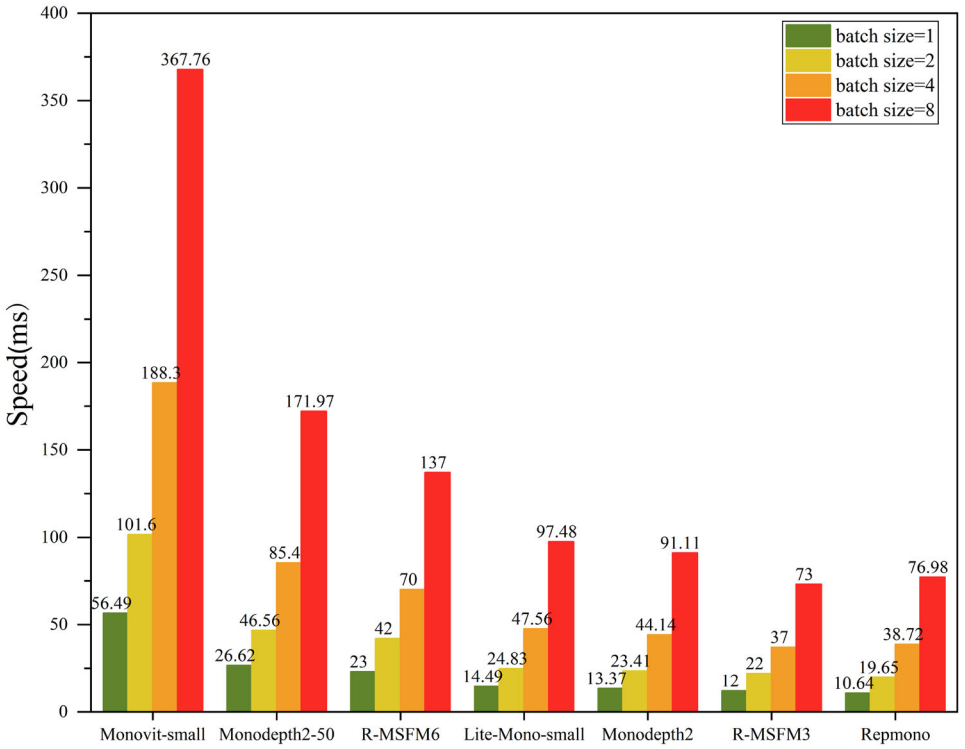

Fig. 8 Comparison of the inference speed of different models on 1650 device. It can be seen that our model inference is significantly faster than the other models, and is fastest when Batch size is set to 1 and 2

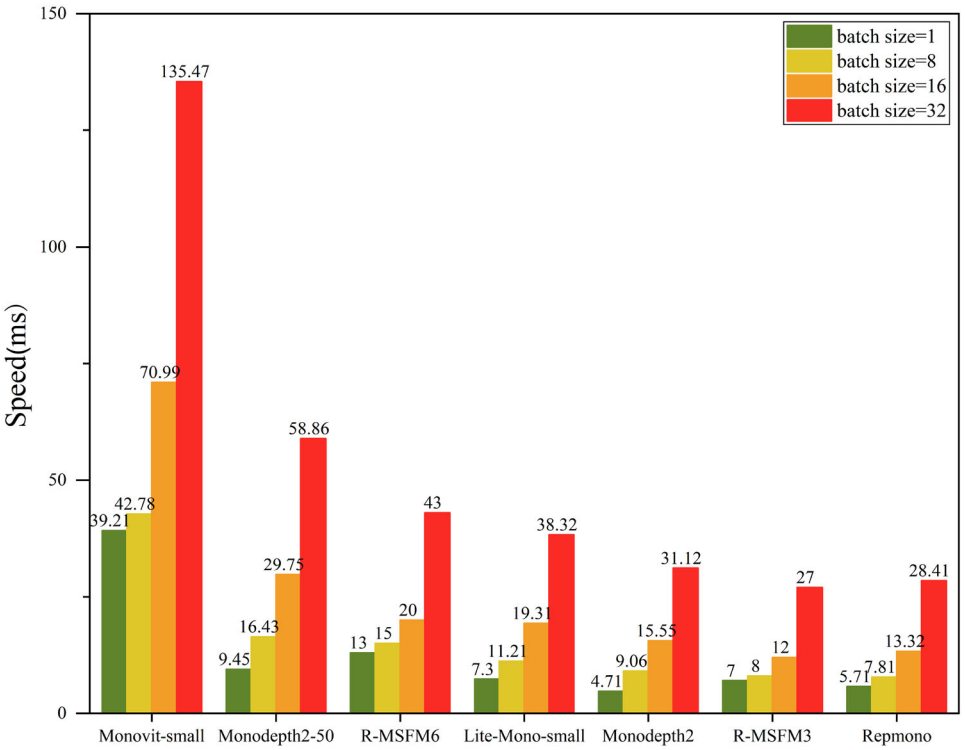

Fig. 9 Comparison of the inference speed of different models on A100 device. It can be seen that the inference speed of our model is not much different from that of Monodepth2 [27] and R-MSFM3 [46], and is significantly faster than that of Monovit-small and Monodepth-50, and is fastest when Batch size is set to 8

Conclusion

This paper introduces a lightweight self-supervised monocular depth estimation model named Repmono, and its two supporting modules, LCKT and RepTM. This approach effectively combines large convolutions and reparameterization structures, enabling Repmono to achieve high performance and fast inference without relying on core Transformer modules. Compared to the classical model Monodepth2, Repmono reduces the number of parameters by 83.8% and speeds up inference by 60.1%. Repmono alleviates the issues of high complexity and slower inference speeds associated with the use of Transformer structures in depth estimation networks, although there is still room for improvement in the network's accuracy. Our approach is extensively tested on the KITTI dataset to validate its effectiveness and demonstrated good generalization on the Make3D dataset and Drivingstereo dataset. We hope our proposed method can contribute to the advancement of self-supervised monocular depth estimation and inspire ideas for subsequent research.

Acknowledgements This paper is partially supported by the Natural Science Foundation of Fujian Province (2023J011437), Education and Research Foundation of Fujian Province (JAT200464) and Fujian Province Key Laboratory of Green Intelligent Cleaning Technology and Equipment.

Data availability The generated data in this study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Menze M, Geiger A (2015) Object scene flow for autonomous vehicles. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3061–3070
2. DeSouza GN, Kak AC (2002) Vision for mobile robot navigation: a survey. IEEE Trans Pattern Anal Mach Intell 24(2):237–267
3. Xu G, Khan A, Moshayedi AJ, Zhang X, Shuxin Y (2022) The object detection, perspective and obstacles in robotic: a review. EAI Endorsed Trans AI Robot 1:7–15. https://doi.org/10.4108/airo.v1i1.2709
4. Newcombe RA, Lovegrove SJ, Davison AJ (2011) DTAM: dense tracking and mapping in real-time. In: 2011 International conference on computer vision. IEEE, pp 2320–2327
5. Zhao C, Sun Q, Zhang C, Tang Y, Qian F (2020) Monocular depth estimation based on deep learning: an overview. Sci China Technol Sci 63(9):1612–1627
6. Dai Z, Cai B, Lin Y, Chen J (2021) UP-DETR: unsupervised pre-training for object detection with transformers. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 1601–1610
7. Zhao C, Zhang Y, Poggi M, Tosi F, Guo X, Zhu Z, Huang G, Tang Y, Mattoccia S (2022) Monovit: self-supervised monocular depth estimation with a vision transformer. In: 2022 International conference on 3D vision (3DV). IEEE, pp 668–678
8. Varma A, Chawla H, Zonooz B, Arani E (2022) Transformers in self-supervised monocular depth estimation with unknown camera intrinsics. CoRR **abs/2202.03131**. arXiv:2202.03131
9. Zhang N, Nex F, Vosselman G, Kerle N (2023) Lite-mono: a lightweight CNN and transformer architecture for self-supervised monocular depth estimation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 18537–18546
10. Moshayedi AJ, Khan AS, Yang S, Zanjani SM (2022) Personal image classifier based handy pipe defect recognizer (hpd): Design and test. In: 2022 7th International conference on intelligent computing and signal processing (ICSP), pp 1721–1728. https://doi.org/10.1109/ICSP54964.2022.9778676
11. Moshayedi AJ, Uddin NMI, Khan AS, Zhu J, Emadi Andani M (2023) Designing and developing a vision-based system to investigate the emotional effects of news on short sleep at noon: an experimental case study. Sensors. https://doi.org/10.3390/s23208422
12. Yu W, Luo M, Zhou P, Si C, Zhou Y, Wang X, Feng J, Yan S (2022) Metaformer is actually what you need for vision. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 10819–10829
13. Hu J, Shen L, Sun G (2018) Squeeze-and-excitation networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 7132–7141
14. Geiger A, Lenz P, Stiller C, Urtasun R (2013) Vision meets robotics: the Kitti dataset. Int J Robot Res 32(11):1231–1237
15. Saxena A, Sun M, Ng AY (2008) Make3D: learning 3D scene structure from a single still image. IEEE Trans Pattern Anal Mach Intell 31(5):824–840
16. Yang G, Song X, Huang C, Deng Z, Shi J, Zhou B (2019) Drivingstereo: a large-scale dataset for stereo matching in autonomous driving scenarios. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 899–908
17. Eigen D, Puhrsch C, Fergus R (2014) Depth map prediction from a single image using a multi-scale deep network. Adv Neural Inf Process Syst 27
18. Li B, Shen C, Dai Y, Van Den Hengel A, He M (2015) Depth and surface normal estimation from monocular images using regression on deep features and hierarchical CRFs. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1119–1127
19. Liu F, Shen C, Lin G, Reid I (2015) Learning depth from single monocular images using deep convolutional neural fields. IEEE Trans Pattern Anal Mach Intell 38(10):2024–2039
20. Xu D, Ricci E, Ouyang W, Wang X, Sebe N (2017) Multi-scale continuous CRFs as sequential deep networks for monocular depth

estimation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 5354–5362
21. Garg R, Bg VK, Carneiro G, Reid I (2016) Unsupervised CNN for single view depth estimation: Geometry to the rescue. In: Computer vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VIII 14. Springer, pp 740–756
22. Godard C, Mac Aodha O, Brostow GJ (2017) Unsupervised monocular depth estimation with left-right consistency. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 270–279
23. Zhou T, Brown M, Snavely N, Lowe DG (2017) Unsupervised learning of depth and ego-motion from video. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1851–1858
24. Vijayanarasimhan S, Ricco S, Schmid C, Sukthankar R, Fragkiadaki K (2017) Sfm-net: learning of structure and motion from video. arXiv preprint arXiv:1704.07804
25. Guizilini V, Hou R, Li J, Ambrus R, Gaidon A (2020) Semantically-guided representation learning for self-supervised monocular depth. arXiv preprint arXiv:2002.12319
26. Yin Z, Shi J (2018) Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1983–1992
27. Godard C, Mac Aodha O, Firman M, Brostow GJ (2019) Digging into self-supervised monocular depth estimation. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 3828–3838
28. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
29. Zhou H, Greenwood D, Taylor S (2021) Self-supervised monocular depth estimation with internal feature fusion. arXiv preprint arXiv:2110.09482
30. Guizilini V, Ambrus R, Pillai S, Raventos A, Gaidon A (2020) 3d packing for self-supervised monocular depth estimation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 2485–2494
31. Yan J, Zhao H, Bu P, Jin Y (2021) Channel-wise attention-based network for self-supervised monocular depth estimation. In: 2021 International conference on 3D vision (3DV). IEEE, pp 464–473
32. Li Y, Luo F, Li W, Zheng S, Wu H-H, Xiao C (2021) Self-supervised monocular depth estimation based on image texture detail enhancement. Vis Comput 37(9):2567–2580
33. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556
34. Loshchilov I, Hutter F (2017) Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101
35. Hoang V-T, Jo K-H (2019) Pydnet: an efficient CNN architecture with pyramid depthwise convolution kernels. In: 2019 International conference on system science and engineering (ICSSE). IEEE, pp 154–158
36. Wofk D, Ma F, Yang T-J, Karaman S, Sze V (2019) Fastdepth: fast monocular depth estimation on embedded systems. In: 2019 International conference on robotics and automation (ICRA). IEEE, pp 6101–6108
37. Ding X, Zhang X, Ma N, Han J, Ding G, Sun, J (2021) Repvgg: Making vgg-style convnets great again. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 13733–13742
38. Ding X, Zhang X, Han J, Ding G (2021) Diverse branch block: building a convolution as an inception-like unit. CoRR **abs/2103.13425** arXiv:2103.13425
39. Lyu X, Liu L, Wang M, Kong X, Liu L, Liu Y, Chen X, Yuan Y (2021) Hr-depth: high resolution self-supervised monocular depth estimation. In: Proceedings of the AAAI conference on artificial intelligence, vol 35, pp 2294–2301
40. Jaderberg M, Simonyan K, Zisserman A et al (2015) Spatial transformer networks. Adv Neural Inf Process Syst 28
41. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP (2004) Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process 13(4):600–612
42. Uhrig J, Schneider N, Schneider L, Frnke U, Brox T, Geiger A (2017) Sparsity invariant CNNs. In: 2017 International conference on 3D vision (3DV). IEEE, pp 11–20
43. Wang C, Buenaposada JM, Zhu R, Lucey S (2018) Learning depth from monocular videos using direct methods. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2022–2030
44. Casser V, Pirk S, Mahjourian R, Angelova A (2019) Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. In: Proceedings of the AAAI conference on artificial intelligence, vol 33, pp 8001–8008
45. Klingner M, Termöhlen J-A, Mikolajczyk J, Fingscheidt T (2020) Self-supervised monocular depth estimation: solving the dynamic object problem by semantic guidance. In: Computer vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16. Springer, pp 582–600
46. Zhou Z, Fan X, Shi P, Xin Y (2021) R-msfm: recurrent multi-scale feature modulation for monocular depth estimating. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 12777–12786
47. Sun Q, Tang Y, Zhang C, Zhao C, Qian F, Kurths J (2021) Unsupervised estimation of monocular depth and VO in dynamic environments via hybrid masks. IEEE Trans Neural Netw Learn Syst 33(5):2023–2033