Self-supervised Coarse-to-fine Monocular Depth Estimation Using Lightweight Attention Module

Yuanzhen Li, Fei Luo, Chunxia Xiao*
School of Computer Science, Wuhan University, Wuhan, Hubei, China
yuanzhen@whu.edu.cn, luofei@whu.edu.cn, cxxiao@whu.edu.cn

Abstract

Self-supervised monocular depth estimation has been widely investigated and applied in previous works. However, existing methods suffer from texture-copy, depth drift, and incomplete structure. It is difficult for normal CNN networks to completely understand the relationship between the object and its surrounding environment. Moreover, it is hard to design the depth smoothness loss to balance depth smoothness and sharpness. To address these issues, we propose a coarse-to-fine method with a normalized convolutional block attention module (NCBAM). In the coarse estimation stage, we incorporate the NCBAM into depth and pose networks to overcome the texture-copy and depth drift problems. Then, we use a new network to refine the coarse depth guided by the color image and produce a structure-preserving depth result in the refinement stage. Our method can produce results competitive with state-of-the-art methods. Comprehensive experiments prove the effectiveness of our two-stage method using the NCBAM.

1. Introduction

Depth information in a 2D image has a wide range of applications, including 3D reconstruction [2, 9, 56, 11, 10], simultaneous localization and mapping (SLAM) [33], shadow removal [8], and so on. Range finding sensors, such as LiDAR, time of flight cameras (TOF), and stereo cameras, are often used to extract depth information. However, it is unrealistic to rely on such expensive or complex sensors in many cases. This has advanced the development of learning-based methods using large datasets [23, 51]. Supervised monocular depth estimation methods have made great progress [19]. However, collecting extensive and high-quality ground truth depth is challenging due to sensor noise and unpredictable complex environmental conditions. Supervised monocular depth estimation thus has limited generalization ability.

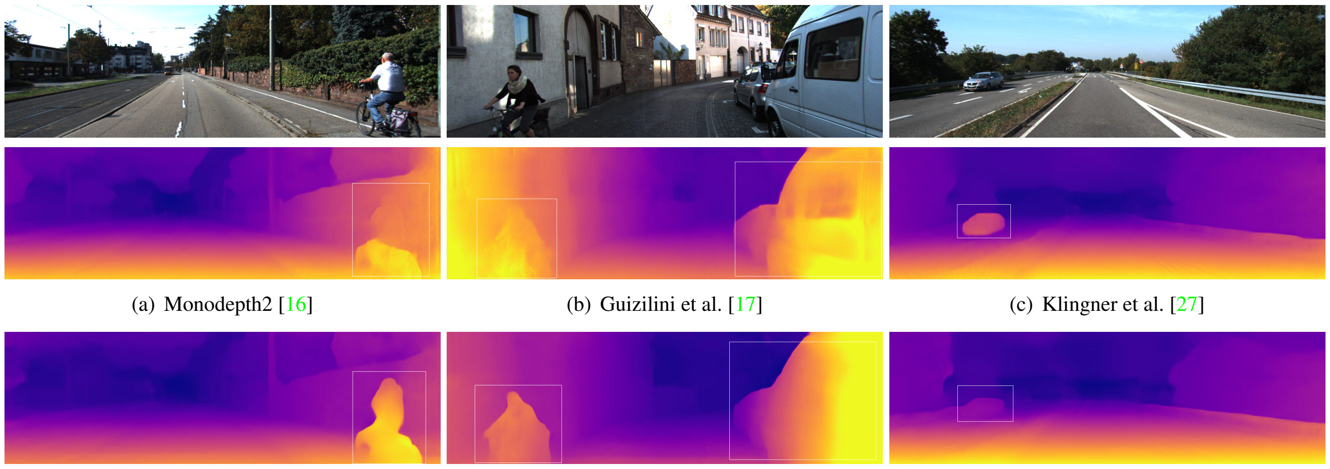
Recently, self-supervised monocular depth estimation approaches have been introduced, trained with stereo image pairs [15], or monocular video sequences [16, 63, 62], and supervised with geometric information. Compared to stereo-based supervision, monocular video is more attractive, as more sequenced frames are available for use as supervision signals. To enhance the performance of depth estimation, many works focus on masking strategies [16, 27, 63], loss functions [16], and multi-task learning [44, 57]. However, existing self-supervised monocular depth methods still suffer from texture-copy, depth drift, and incomplete structure.

Texture-copy in a depth map the result when details of the color image are transferred to the depth map. Monodepth2 [16] upsamples the generated multi-scale depths to the input image resolution and then computes all losses, to partially alleviate the texture-copy phenomenon. Depth drift occurs when object depth largely differs from its surrounding environment in the wrong way. It is caused by the depth network incompletely understanding the spatial correlation between the object and its surrounding environment. Incomplete structures result when in object depths are incompletely predicted, especially for sharp objects in the scene, as the smoothness loss mistakenly eliminates the depth differences of the sharp object. In Fig. 1, we illustrate some typical examples of the above problems in the predicted depth; our predicted depth maps are better than those of the comparator methods.

We propose a coarse-to-fine method with a normalized convolutional block attention module (NCBAM). Our pipeline includes coarse depth estimation and depth refinement, as shown in Fig. 2. Specially, we improve the lightweight CBAM attention module [53], to provide a normalized convolutional block attention module (NCBAM), and then incorporate it into networks to tackle the problems of texture-copy and depth drift. Furthermore, we design a network that uses the corresponding color image as a guide to refine the coarse depth, which can deal with the incomplete structure problem. The coarse depth network and depth refinement network are trained, respectively.

To summarize, this paper presents the following two main contributions:

- we tackle the texture-copy and depth drift problems by


(a) Monodepth2 [16] (b) Guizilini et al. [17] (c) Klingner et al. [27]

(d) Ours (e) Ours (f) Ours

Figure 1. Existing methods suffer from texture-copy, depth drift, and incomplete structure. (a) The depth of the person exhibits an incomplete structure problem in the Monodepth2 [16] output. (b) The depths of the person and the car suffer from a texture-copy problem in the result of Guizilini et al. [17]. (c) The car depth has a drift problem in the output of Klingner et al. [27].

improving the CBAM and incorporating it into depth and pose networks.

- we tackle the incomplete structure problem by designing a new network using the color image as a guide to refine the coarse depth.

2. Related work

2.1. Background

Inferring depth from a single image is an ill-posed problem. However, deep learning has shown its ability to provide acceptable estimation results based on large-scale datasets. In this section, we mainly review related work on self-supervised monocular depth estimation.

Traditionally, structure-from-motion [46] and binocular stereo algorithms [20] have been used to estimate depth from a series of images or stereo image pairs, respectively. Recently, learning-based algorithms have made great progress in monocular depth estimation [6, 7]. Supervised methods train a network model on sparse depth labels provided using RGBD sensors. However, it is not easy to collect high-quality ground truth depths. As an alternative, self-supervised depth estimation has attracted attention, using stereo image pairs [15] or monocular video sequences [16, 59] as training datasets. Self-supervised depth estimation trains the depth estimation model by projecting one view to nearby views based on the predicted depth and minimizing the photometric re-projection loss between the projected image and the target image.

2.2. Self-supervised Stereo Training

Deep3D [54] uses a deep neural network to generate 3D stereo image pairs from 2D images or video frames and uses the photometric re-projection loss to train the depth network on the stereo image pair datasets. This network predicts a probabilistic disparity map for the input image, and the depth-based image rendering layer produces the right image in the context of binocular pairs. Garg et al. [12] proposed a deep neural network to directly estimate the depth and trained loss terms including a photometric re-projection loss and a depth smoothness loss. Monodepth [15] inputs a left image into a depth network and predicts left-right disparities to enforce mutual consistency. This method uses photometric re-projection loss and introduces a left-right disparity consistency loss. Both methods [39] and [1] use generative adversarial networks to train the depth network.

2.3. Self-supervised Monocular Training

Monocular video is more attractive than stereo-based supervision, as more frame sequences are available for use as supervision signals. Self-supervised monocular training needs to estimate the parameters of the depth and pose estimation models. The pose estimation network takes a finite series of frames as input and outputs the relative camera pose. The source frame is warped to the target frame based on the predicted depth and relative camera pose, and then the photometric error between the warped frame and the target frame is used to supervise the model during training [63].

The method in [63] was the first work that used monocular video to train end-to-end depth and camera pose estimation networks. Mahjourian et al. [34] used a 3D geometry consistency loss to train the model. Godard et al. [16] made the following three innovations. First, they proposed a minimum photometric re-projection loss to address the problem of occluded pixels. Then, they designed an auto-masking

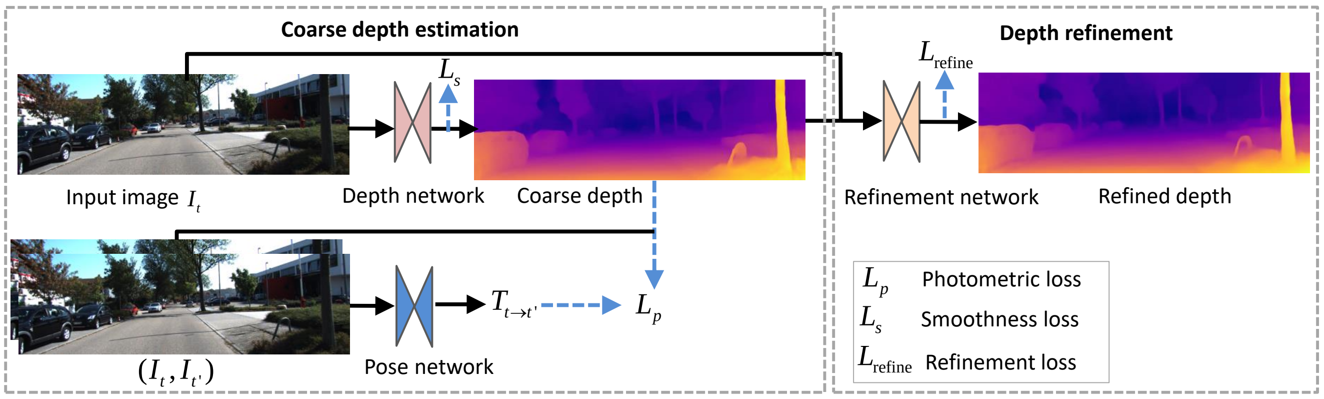


Figure 2. Overview of our method. We use a coarse-to-fine method comprising coarse depth estimation and depth refinement. We train the depth and pose estimation models in the coarse depth estimation stage. In the depth refinement stage, we only train the depth refinement model.

loss to ignore training pixels that violate relative camera motion assumptions. Finally, they upsampled the predicted depth maps to the input resolution and computed all losses to reduce texture-copy artifacts.

Multi-task training strategies are also available to improve the accuracy of depth estimation. Yang et al. [57] constrained the depth to be consistent with the surface normal and image edges. Ying et al. [59] learned depth, optical flow, and pose together and used the predicted depth and optical flow to mask moving objects during training. Zhu et al. [64] used edge consistency between the semantic segmentation and depth map as a supervision signal. Klingner et al. [27] used the learned semantic information to eliminate the influence of moving objects when computing photometric re-projection loss.

Self-attention (Transformer) [48] has improved the performance of natural language processing systems by better handling of long-range dependencies between words. In addition, self-attention has been applied in computer vision tasks such as semantic segmentation [60] and depth estimation [43, 30, 55]. Johnston et al. [22] used the ResNet-101 network to encode the input image and then passed it through a self-attention module [48] to explore contextual information, allowing the inference of similar depth values in discontiguous regions of the input image. However, as the self-attention module requires much memory, they only incorporated it into the encoder output layer.

Attention mechanisms have achieved great success in many visual tasks, such as image classification, object detection, and semantic segmentation [18]. We improve the lightweight attention module CBAM [53], to give a normalized convolutional block attention module (NCBAM). Then, we generalize the NCBAM model in multiple places, including the depth estimation network, relative pose estimation network, and depth refinement network, to improve the accuracy of the depth and pose estimation models.

3. Method

In this section, we give a detailed description of our method (see Fig. 3). First, we introduce the improvements to CBAM to give NCBAM. Then, we describe the coarse depth and pose estimation methods. Finally, we present the depth refinement approach. We use the Monodepth2 [16] network as a baseline.

3.1. NCBAM Attention Module

The convolutional block attention module (CBAM) [53] is a lightweight module, which can aggregate deep features. It sequentially infers attention maps along with two separate sub-modules: channel and spatial, as shown in Fig. 4. The attention maps are multiplied by the input feature map for adaptive feature refinement. The CBAM attention module can learn correlations between the object itself and the surrounding environment. When incorporating the CBAM model into depth estimation networks, it cannot completely solve the problems of texture-copy and depth drift (as shown in Fig. 12 later).

We improve upon the CBAM module in the following ways, in a normalized convolutional block attention module (NCBAM). To reduce the differences between global average pooling and global max-pooling in the channel and spatial attention modules, we convert the input feature to the range $(-1, 1)$ using the tanh function. We use the activation function $\text{softplus}(x) = \log(1 + e^x)$ to replace $\text{relu}(x) = \max(0, x)$ in the shared network of the channel sub-module and the convolution layer of the spatial sub-module. The activation function softplus can be seen as smoothing relu, avoiding excessive neuronal death during training. Experiment comparisons show that the NCBAM module can produce better results than the CBAM module.

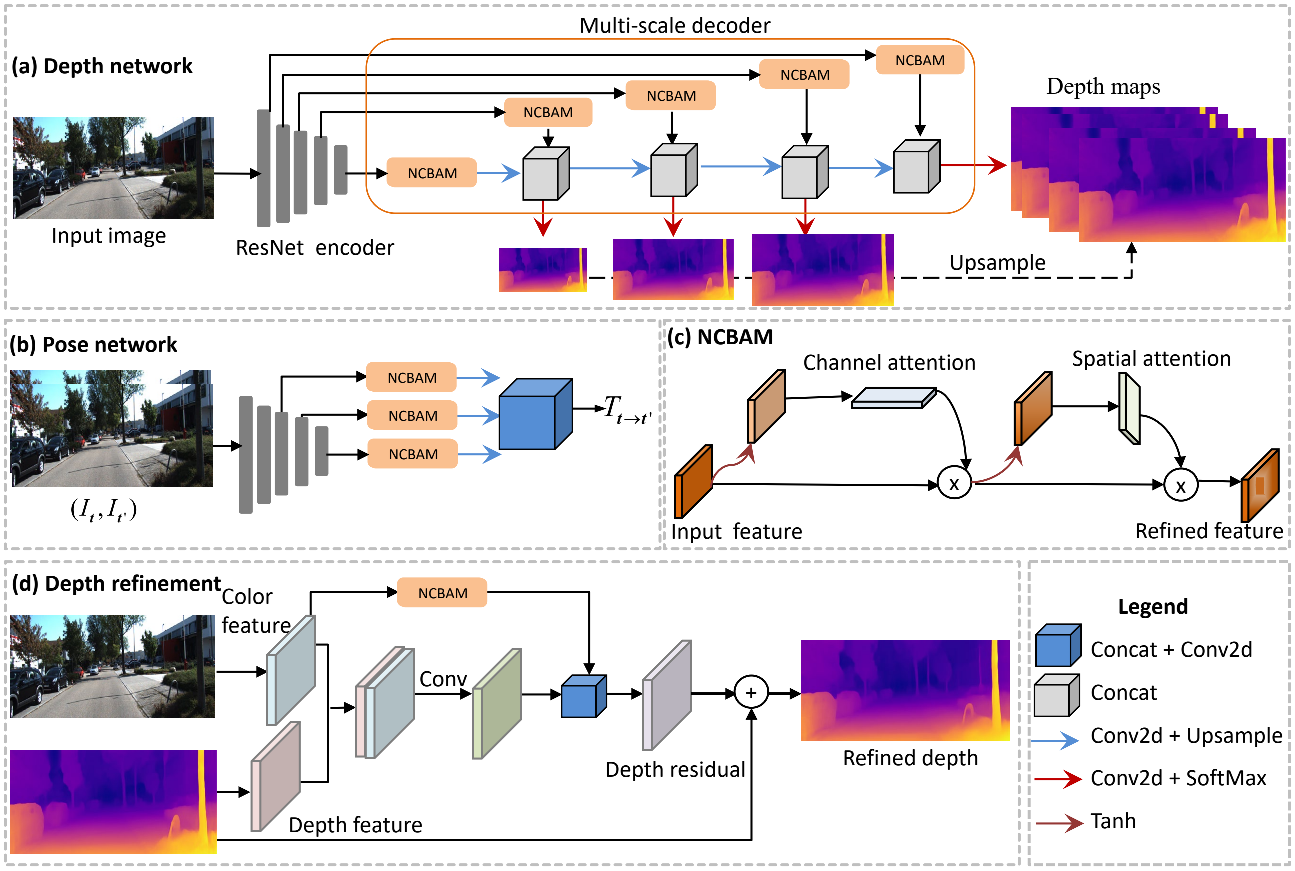


Figure 3. Detailed architecture of our method. (a) Depth network. We incorporate the NCBAM module into the depth estimation network [16], which is a U-Net network, an encoder with residual blocks, and a decoder with skip connections. (b) Pose network. We incorporate the NCBAM module into the standard pose network [16]. (c) NCBAM module. (d) Depth refinement network. This uses the color image as guidance to refine the coarse depth.

3.2. Coarse Depth Estimation

We need to train depth and pose estimation networks simultaneously based on monocular video training. The depth estimation network $f_D : I \to D$ predicts the depth for every pixel in the target image I_t. The pose estimation network f_T: $(I_t, I_{t'}) \to T_{t\to t'}$, predicts the camera transformation relating the target image I_t to the source image $I_{t'}$. Based on the learned depth and pose, we warp the frame into adjacent frames $\{t-1,\ t+1\}$ using the photometric re-projection loss as the optimization objective function. We follow the multi-scale depth output strategy proposed in [15]. First, we input encoder features into deep convolutions and output a low-resolution depth map. Then, we apply three additional stages of upsample-convolution that receive skip connections from the ResNet encoder to generate corresponding resolution depth maps.

The NCBAM attention module can aggregate deep features and extract correlations between the object and the surrounding environment. Therefore, we incorporate the NCBAM model into the ResNet multi-scale features and the skip connections associated with decoder layers for the U-Net architecture (SK feature), as shown in Fig. 3(a). Skip connections between encoder and associated decoder layers can keep high-level information in the final depth output. When incorporating the NCBAM module, the depth estimation model can overcome with the texture-copy and depth drift problems.

The output of our depth network is a pixel-wise disparity probability for multiple disparity layers, to give a discrete disparity volume (DDV) [25]. We input the SK feature into a 2D convolutional layer with filters of size 3×3, and output a K channel disparity probability volume $P = \{P_1, \ldots, P_K\}$ with K disparity layers:

$$d_k = d_{\min} + \Delta_d(k-1), \qquad k = 1, \ldots, K, \qquad (1)$$

where $d_{\min}$ and Δ_d are the minimum disparity value and disparity interval. A depth-wise softmax operation processes P to produce an actual probability map for each disparity plane:

$$P^d = \text{softmax}(P). \qquad (2)$$

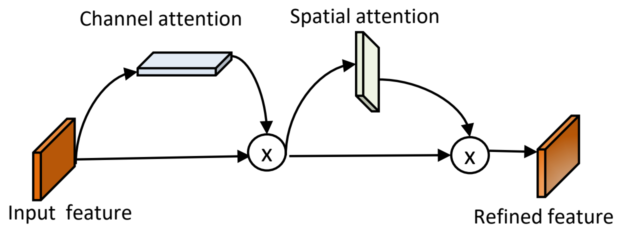


Figure 4. The CBAM attention module [53] has two sequential sub-modules: channel and spatial.

We extract the final disparity as a weighted sum of the disparity probabilities P^d:

$$D = \sum_{k=1}^{K} d_k P_k^d. \tag{3}$$

We use a widely used backbone [16], which takes two images at different time steps as input and learns the relative camera pose transformation $T_{t \to t'} \in SE(3)$ between the images recorded at time steps t and t':

$$T_{t \to t'} = f_T(I_t, I_{t'}). \tag{4}$$

The special Euclidean group $SE(3)$ defines the set of all possible rotations and translations. Such transformations are usually represented by 4×4 matrices. Following [16], we predict the six degrees of freedom pose.

We also incorporate the NCBAM model into the pose network, as shown in Fig. 3(d). The NCBAM model can also learn other related features in the two input images, enhancing the accuracy of the pose model, furthermore increasing the accuracy of the depth estimation model. First, we input a pair of color images to the ResNet-18 network to extract corresponding deep features. Then, we input those deep features into the NCBAM module to learn their correlation (CF features). Finally, we concatenate the CF features and input them to a series of 2D convolution layers to output a single six degrees of freedom relative pose.

3.2.1 Training the Coarse Depth Network

Following [16], training our coarse depth estimation model is mainly based on minimizing per-pixel photometric re-projection loss between the source image $I_{t'}$ and target image I_t, using the learned relative pose $T_{t \to t'}$ and depth D_t. The photometric re-projection loss is defined as:

$$L_p = \mu \min_{t'} \mathrm{pe}(I_t, I_{t' \to t}), \tag{5}$$

where $\mathrm{pe}(.)$ is the photometric reconstruction error; $t' \in \{t-1, t+1\}$: we use the two frames temporally adjacent to I_t as the source frames [16]. μ is a binary mask that filters out stationary points:

$$\mu = [\min_{t'} \mathrm{pe}(I_t, I_{t' \to t}) < \min_{t'} \mathrm{pe}(I_t, I_{t'})], \tag{6}$$

where $[.]$ is the Iverson bracket. The binary mask μ includes pixels where the re-projection error of $I_{t' \to t}$ is lower than the un-warped image $I_{t'}$, indicating that the object is stationary relative to the camera. Re-projection loss minimization significantly reduces artifacts along the object boundaries in the image, leading to better accuracy. The re-projected image is defined as:

$$I_{t' \to t} = I_{t'} \left\langle \mathrm{proj}(\sigma(D_t), T_{t \to t'}, K) \right\rangle, \tag{7}$$

where $\langle . \rangle$ is the sampling operator; $K \in \mathbb{R}^{3 \times 3}$ is the camera intrinsic parameter matrix, identical for all images. We also apply differentiable bilinear sampling [21] to sample the source images. $\mathrm{proj}(.)$ returns 2D coordinates of the projected depths D_t in $I_{t'}$ [4]:

$$\mathrm{proj}(\sigma(D_t), T_{t \to t'}, K) = K T_{t \to t'} D_t(p_t) K^{-1} p_t, \tag{8}$$

where p_t denotes a pixel. The photometric reconstruction error function pe [15] is

$$\mathrm{pe}(I_t, I_{t'}) = \frac{\alpha}{2}(1 - \mathrm{ssim}(I_t, I_{t'})) + (1-\alpha) \left\| I_t - I_{t'} \right\|_1, \tag{9}$$

where $\alpha = 0.85$, and $\mathrm{ssim}()$ is structural similarity measured as in [50] with a 3×3 block filter.

Monodepth2 [16] encourages neighbouring pixels to have similar depths, and uses an edge-aware depth smoothness loss L_s weighted by image gradients to improve predictions around object boundaries:

$$L_s = |\partial_x \hat{D}_t| \ \exp(-|\partial_x I_t|) + |\partial_y \hat{D}_t| \ \exp(-|\partial_y I_t|), \tag{10}$$

where ∂_x, ∂_y are gradient operators in x, y, and $\hat{D}_t = D_t / \overline{D_t}$ is the mean-normalized inverse depth from [49] to discourage shrinking of the estimated depth. The final loss is computed as the weighted sum of masked image photometric re-projection loss L_p and smoothness loss L_s:

$$L_{\text{coarse}} = L_p + \lambda L_s, \tag{11}$$

where λ weights the smoothness term. In our experiments, we set it to 0.01.

3.3. Depth Refinement

3.3.1 Background

In Section 3.2, we incorporate the NCBAM module into the depth estimation network to tackle the problems of texture-copy and depth drift, which can improve the accuracy of the initial depth estimates. However, these estimates are still imperfect. In particular, the depths may exhibit incomplete structure in sharp object regions, as shown in Fig. 7. We design a refinement network that uses the color image as guidance to refine the coarse depth, and deal with the incomplete structure problem.

Table 1. Refinement network architecture. **k** = kernel size, **s** = stride, **d** = kernel dilation, **chns** = number of input and output channels for each layer, **input** = input source of each layer, and + indicates concatenation.

Layer	k	s	d	chns	active	input
conv1_1	3	1	1	3/32	PReLU	I_t
conv1_2	3	1	1	32/32		conv1_1
conv1_3	3	1	2	32/32		conv1_2
conv1_4						NCBAM(conv1_3)
conv2_1	3	1	1	1/32	PReLU	D_t
conv2_2	3	1	1	32/32		conv2_1
conv3_1	3	1	1	64/24	PReLU	conv1_2 + conv1_2
conv3_2	3	1	1	24/24		conv3_1
conv4_1	3	1	2	56/24	PReLU	conv3_2 + conv1_4
conv4_2	3	1	2	24/24		conv4_1
conv5_1	3	1	2	24/24	PReLU	conv4_2
conv5_2	3	1	1	24/1		conv5_1

First, we input the color and corresponding coarse depth images into a series of 2D convolution layers to extract their features. Then, we concatenate the output features and pass them through a 2D convolution to generate the color-depth feature. Meanwhile, we input the color feature into the NCBAM module, allowing the network to learn more about the color feature. Finally, we concatenate the color-depth feature and the refined color feature and pass them through a series of 2D convolution layers to output a depth residual. The depth residual is added to the coarse depth to get the refined depth. Tab. 1 presents a detailed specification of the depth refinement network.

3.3.2 Training the Depth Refinement Model

Depth and normal are two highly correlated entities. Inspired by [42], we design a normal consistency loss for the coarse depth $\widehat{D}$ and refined depth D:

$$L_{\mathrm{n}}(u,\widehat{u}) = \frac{1}{N}\sum_i \left(1 - \frac{<\widehat{u_i}, u_i>}{||\widehat{u_i}||\;||u_i||}\right), \tag{12}$$

where N denotes the number of pixels, i indexes pixels, and $u_i = (\partial_x D_i, \partial_y D_i)$. Angle minimization is performed by maximizing the dot-product.

We also use the photometric re-projection loss in Eq. (5) with the camera pose model trained in the coarse depth estimation work. Here, we use multi-scale structural similarity, MS-SSIM [61]. The photometric reconstruction error function pe is

$$\begin{aligned}\mathrm{pe}'(I_t, I_{t'}) = \;& (1-\alpha)\left\|I_t - I_{t'}\right\|_1 + \\ & \frac{\alpha}{2}(1 - \mathrm{msssim}(I_t, I_{t'}, s)),\end{aligned} \tag{13}$$

where $s = 4$ is the number of scales employed. Here, the photometric re-projection loss is:

$$L_{p'} = \mu \min_{t'} \mathrm{pe}'(I_t, I_{t\to t'}). \tag{14}$$

The final learning objective function of our depth refinement network is:

$$L_{\mathrm{refine}} = L_{\mathrm{p'}} + \gamma_n L_{\mathrm{n}} + \gamma_s L_{\mathrm{s}}, \tag{15}$$

where γ_n and γ_s are hyperparameters to control the significance of the normal term L_n and patch smoothness term L_s. In our experiments, γ_n is set to 10^{-4}, and γ_s is set to 10^{-5}.

4. Results and discussion

This section presents experimental results to verify the effectiveness of our approach. Firstly, we describe the experimental datasets and implementation details. Secondly, we present evaluations of our method on various testing configurations. Thirdly, we perform an ablation study to demonstrate that the NCBAM module can improve the accuracy of the predicted depths. Finally, we apply our predicted depths to novel view synthesis and describe limitations of our depth estimation model.

4.1. Datasets and Evaluation Metrics

4.1.1 Datasets

We trained our overall network model using the standard KITTI benchmark [14]. The KITTI dataset collects rectified stereo pairs of 61 scenes (containing about 42,382 stereo frames) mainly concerned with driving scenarios. The image size is 1242×375 pixels. We follow the Eigen split test dataset [6]. It contains 39,810 monocular training sequences consisting of three frames, 4,424 validation sequences, and 697 for evaluation. Following previous work [63], we remove static frames before training and only evaluate depths up to a fixed range of 80 m per standard practice [16]. We use the same intrinsic parameters for all images; we set the camera principal point to the image center and the focal length to the average of all focal lengths in KITTI.

4.1.2 Training dataset augmentation

For the training dataset, we resized all images to a standard resolution (640×192), or high resolution (1024×320). We augmented the training dataset with horizontal flips, and 50% were processed by random adjustments to contrast ± 0.2, saturation ± 0.2, hue ± 0.1, and brightness ± 0.2. The additional color images were only used as depth and pose network input, but the original color images were used to compute the training loss.

Table 2. Comparisons to state-of-the-art methods on the KITTI test dataset [6]. In the Train column: S = self-supervised stereo pair supervision, M = self-supervised monocular video supervision. The best results in each category are in bold.

Method	Train	Memory(MB)	Error (lower is better)				Accuracy (higher is better)		
			Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Zhou et al. [63]	M	126	0.183	1.595	6.709	0.270	0.734	0.902	0.959
Yang et al. [58]	M	-	0.182	1.481	6.501	0.267	0.725	0.906	0.963
Mahjourian et al. [34]	M	126	0.163	1.240	6.220	0.250	0.762	0.916	0.968
GeoNet [59]	M	229	0.149	1.060	5.567	0.226	0.796	0.935	0.975
DDVO [49]	M	142	0.151	1.257	5.583	0.228	0.810	0.936	0.974
DF-Net [65]	M	-	0.150	1.124	5.507	0.223	0.806	0.933	0.973
LEGO [57]	M	211.35	0.162	1.352	6.276	0.252	-	-	-
Ranjan et al. [44]	M	213.471	0.148	1.149	5.464	0.226	0.815	0.935	0.973
EPC++ [32]	M	146.1	0.141	1.029	5.350	0.216	0.816	0.941	0.976
Struct2depth [3]	M	66.82	0.141	1.026	5.291	0.215	0.816	0.945	0.979
Monodepth2 [16]	M	66.72	0.115	0.903	4.863	0.193	0.877	0.959	0.981
Klingner [27]	M	110.15	0.113	0.835	4.693	0.191	0.879	0.961	0.981
Guizilini et al. [17]	M	106.38	0.120	1.018	5.136	0.198	0.865	0.955	0.980
Johnston et al. [22]	M	250.74	0.110	0.872	4.714	0.189	0.878	0.958	0.980
Ours	M	69.13	0.101	0.811	4.674	0.179	**0.890**	0.962	**0.983**
Ours-refine	M	70.58	**0.098**	**0.810**	**4.672**	**0.177**	**0.890**	**0.964**	**0.983**
Garg et al. [13]	S	23	0.152	1.226	5.849	0.246	0.784	0.921	0.967
Monodepth R50 [15]	S	668	0.133	1.142	5.533	0.230	0.830	0.936	0.970
StrAT [35]	S	-	0.128	1.019	5.403	0.227	0.827	0.935	0.971
Poggi et al. (VGG) [40]	S	902	0.119	1.201	5.888	0.208	0.844	0.941	0.978
SuperDepth [38]	S	-	0.112	0.875	4.958	0.207	0.852	0.947	0.977
Monodepth2 [16]	S	61.7	**0.109**	0.873	4.960	0.209	**0.864**	0.948	0.975
Watson et al. [52]	S	132.14	0.111	0.912	4.977	0.205	0.862	0.950	0.977
MonoResMatch [47]	S	487	0.111	**0.867**	**4.714**	**0.199**	**0.864**	**0.954**	**0.979**
UnDeepVO [29]	MS	-	0.183	1.730	6.571	0.268	-	-	-
EPC++ [32]	MS	146.1	0.128	0.936	5.011	0.209	0.831	0.945	0.979
Monodepth2 [16]	MS	66.72	**0.106**	0.818	**4.750**	**0.196**	**0.874**	**0.957**	0.979
WaveletMonodepth [41]	MS	-	0.109	**0.814**	4.808	0.198	0.868	0.955	**0.980**

Table 3. Comparisons to Monodepth2 [16] and Johnston et al. [22], with encoder network ResNet-18, ResNet-50, and ResNet-101, respectively. High denotes the input image resolution is 1024×320.

encoder	Method	Error (lower is better)				Accuracy (higher is better)		
		Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
ResNet-18	Monodepth2 [16]	0.115	0.903	4.863	0.193	0.877	0.959	0.981
	Johnston et al. [22]	0.110	0.872	4.714	0.189	0.878	0.958	0.980
	Ours	**0.101**	**0.811**	**4.674**	**0.179**	**0.890**	**0.962**	**0.983**
	Monodepth2 [16] (High)	0.115	0.882	4.701	0.190	0.879	0.961	0.982
	Ours (High)	**0.101**	**0.807**	**4.635**	**0.173**	**0.891**	**0.962**	**0.983**
ResNet-50	Monodepth2 [16]	0.110	0.831	4.642	0.187	0.883	0.962	0.982
	Ours	**0.098**	**0.795**	**4.631**	**0.171**	**0.890**	**0.963**	**0.983**
ResNet-101	Johnston et al. [22]	0.110	0.872	4.714	0.189	0.878	0.958	0.980
	Ours	**0.097**	**0.788**	**4.623**	**0.170**	**0.892**	**0.963**	**0.984**

4.1.3 Depth evaluation metrics

To evaluate the depth estimation model, we used four error metrics and three accuracy metrics as in [7]. The four error metrics measure the difference between predicted depth and ground-truth depth, namely the absolute relative error (Abs Rel), the squared relative error (Sq Rel), the root mean square error (RMSE), and the logarithmic root mean square error (RMSE log). The three accuracy metrics give the frac-

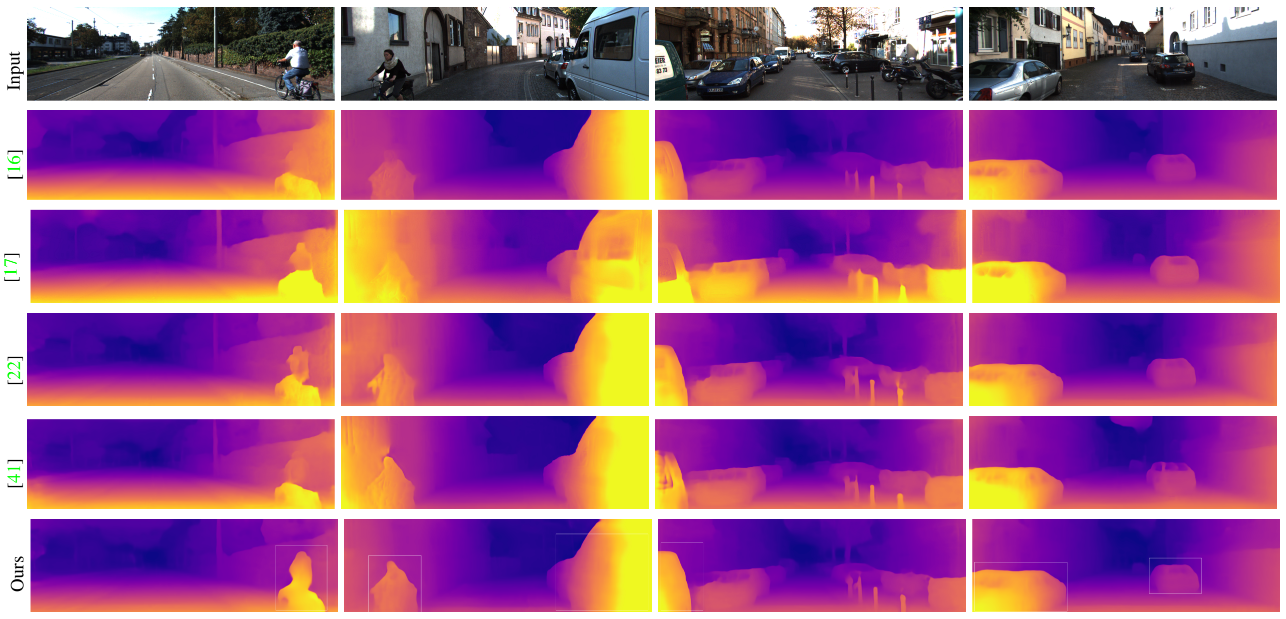


Figure 5. Comparisons to state-of-the-art self-supervised monocular depth estimation methods: Monodepth2 [16], Guizilini et al. [17], Johnston et al. [22], and WaveletMonodepth [41] using examples from the Eigen split test dataset [6].

tion δ of predicted depths in an image whose ratio and inverse ratio to the ground truth are within the thresholds 1.25, 1.25^2, and 1.25^3.

4.2. Implementation Details

In our experiments, we set the number of disparity layers $K = 98$, the minimum disparity value $d_{\min} = 10^{-5}$, and disparity interval $\Delta_d = 0.01$. We used the PyTorch framework [37] to implement our work and trained on a single Nvidia 2080Ti. We used ResNet-18, ResNet-50, and ResNet-101 as the encoders for the depth estimation network. For coarse depth estimation, we used the Adam optimizer [26] with $\alpha = 10^{-4}$, $\beta_1 = 0.9$, and $\beta_2 = 0.999$, training for 25 epochs with a batch size of 8. For depth refinement, $\alpha = 10^{-5}$, $\beta_1 = 0.9$, and $\beta_2 = 0.999$, training for 13 epochs with a batch size of 6. Because the ResNet-101 network needs more memory, the batch size was set to 4 in coarse depth estimation, and we did not train it in the depth refinement work.

4.3. Depth Evaluation on KITTI Dataset

We evaluated our depth estimation model on the Eigen split test dataset [6]. Tab. 2 presents the results, which demonstrate that our method is better than existing methods in terms of the seven evaluation metrics. A qualitative evaluation of our coarse depth estimation model is provided in Fig. 5, showing a comparison to results generated by the methods in [16, 17, 22, 41]. Unlike those methods, our estimated depth maps have complete structures for objects, such as the human body. The estimated depth maps from the comparator methods exhibit texture-copy phenomena in areas such as the car, but our approach overcomes these problems. The depth evaluation results verify that the NCBAM module effectively estimates monocular depth.

We also have qualitatively evaluated our coarse depth estimation model on the Cityscapes test dataset [5]: see Fig. 6. The Cityscapes and KITTI datasets have some differences. Our model only trained on the KITTI dataset was used to predict depths on the Cityscapes dataset. Our predicted depth maps are better than those from the methods in [16, 17, 22, 41], whose predicted depths exhibit texture-copy, depth drift, and incomplete structure problems, in areas such as cars, persons, and landmarks.

The benefits of the depth refinement model are shown in Tab. 2 and Fig. 7. Compared to the coarse depth estimation model, the results of the depth refinement model are further improved. In Fig. 7, we show qualitative results of the coarse and refined depth estimation models. Refined depths provide better results on thin structures such as poles. Tab. 2 and Fig. 7 show that our depth refinement network is effective, and can refine the coarse depth.

We have compared our method to Monodepth2 [16] and Johnston et al. [22] using a variety of encoder networks, including ResNet-18, ResNet-50, and ResNet-101. Tab. 3 shows the quantitative results. Our results are quantitatively better than those of these two methods. Fig. 8 shows a qualitative comparison between our method and Monodepth2 [16] with the ResNet-18 encoder and input image size of 1024×320. Unlike Monodepth2 [16], our depth estimation model can deal with the texture-copy problem and produce a clear depth for a sharp object in the image. Fig. 9

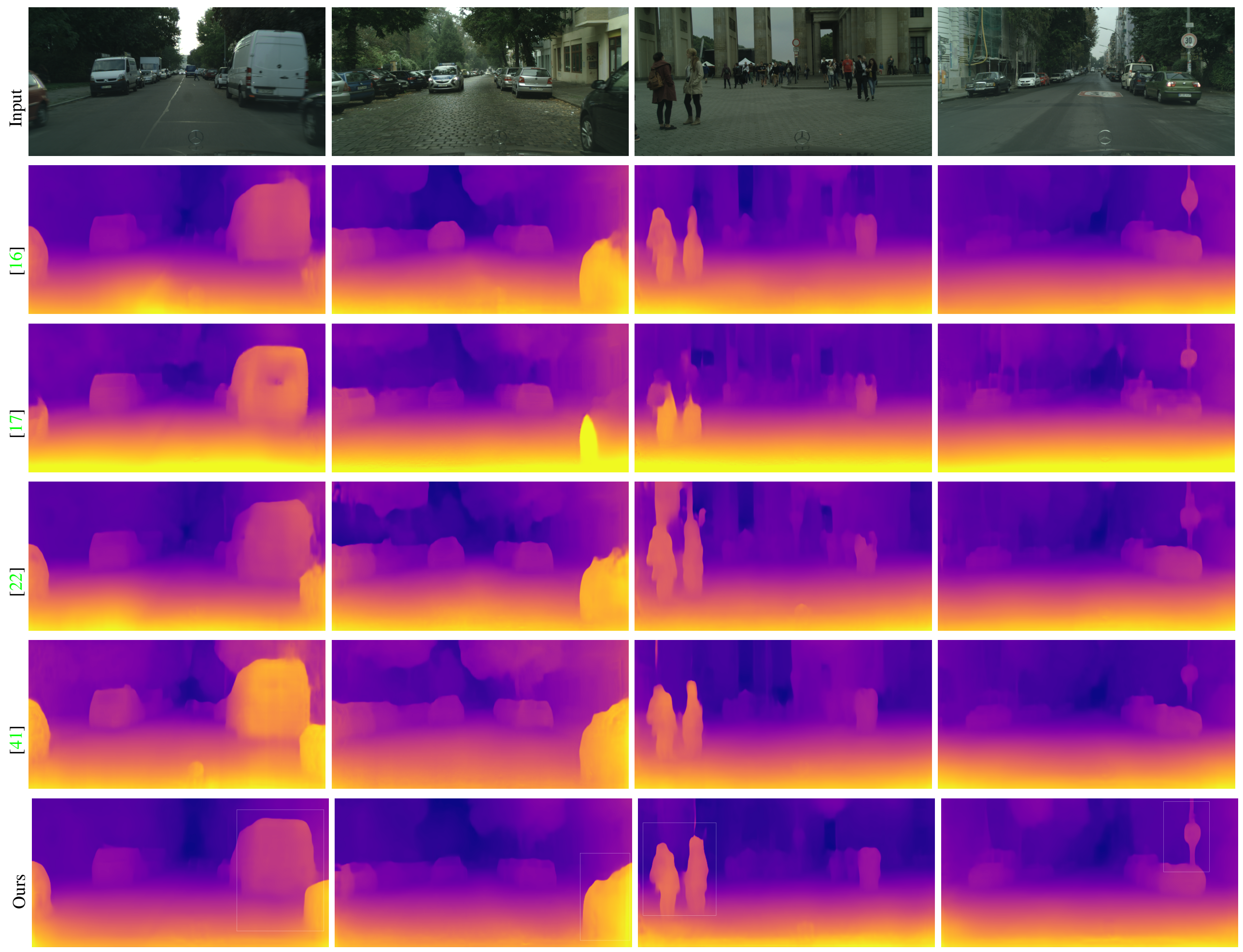


Figure 6. Comparisons to existing self-supervised monocular depth estimation methods: Monodepth2 [16], Guizilini et al. [17], Johnston et al. [22], and WaveletMonodepth [41], using the Cityscapes test dataset [5].

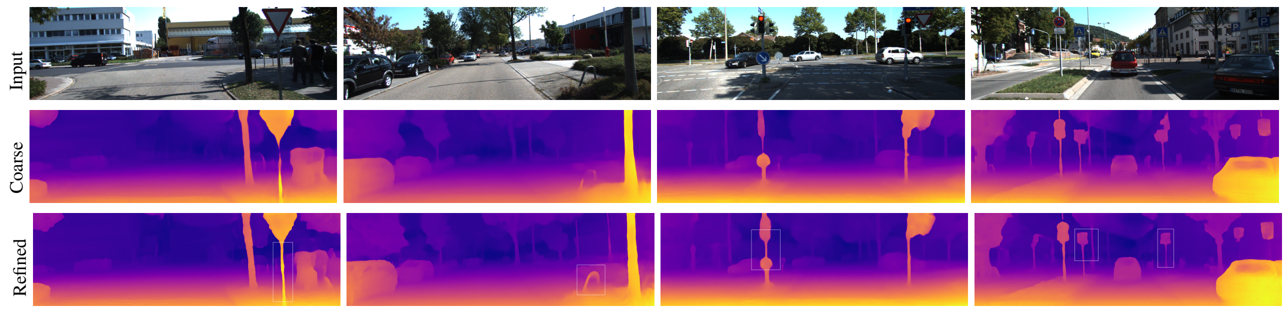


Figure 7. Comparisons between the coarse depth and refined depth. Some depth maps of sharp objects were refined using the depth refinement network.

shows a comparisons between our method and the results of Johnston et al. [22] using the ResNet-101 encoder network. Our approach can predict depths of delicate structures.

4.4. Depth Evaluation on Make3D Dataset

In Tab. 4, we provide quantitative evaluation results on the Make3D dataset [45] using our models trained on the KITTI dataset. We used the same testing protocol as Monodepth2 [16] and the evaluation criteria from Monodepth [15]. For the Make3D dataset, we evaluated on a center crop of 2×1 ratio and used median scaling [16] because the ground truth depth and corresponding input image were not well aligned. Tab. 4 shows that our method can produce better results than previous self-supervision

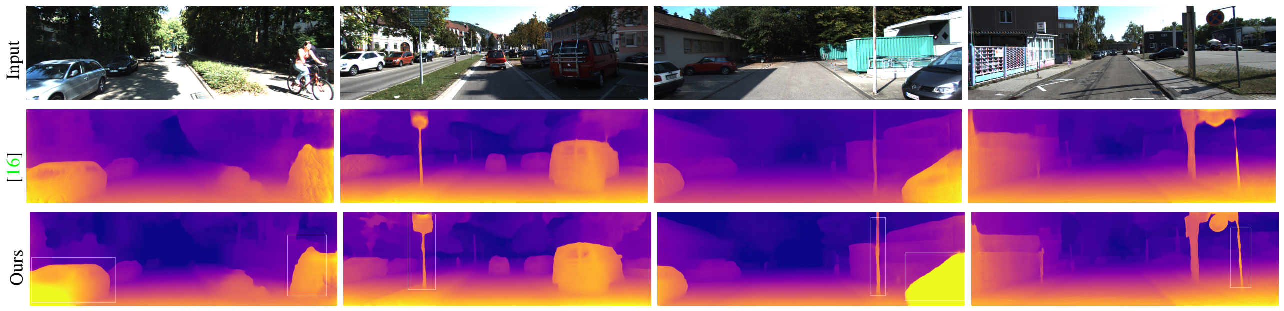


Figure 8. Comparisons to Monodepth2 [16] on an image with 1024×320 resolution. Our method can overcome the texture-copy problem and produce clear depths for sharp objects in the image.

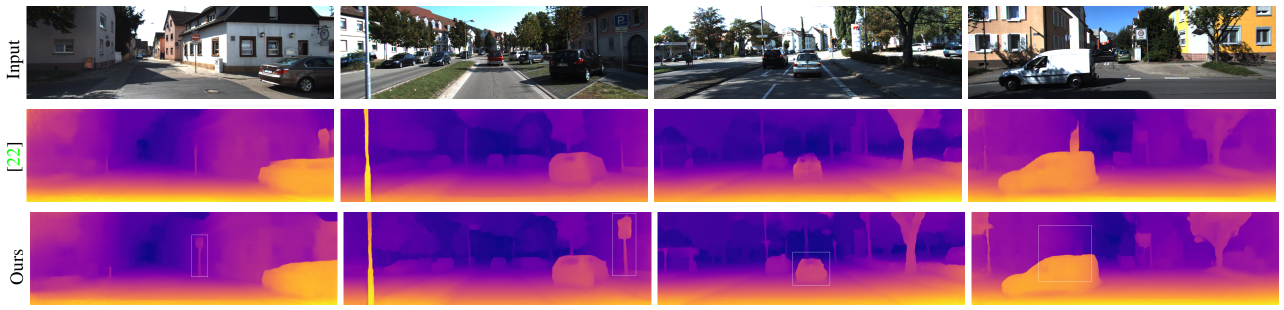


Figure 9. Comparison to results from Johnston et al. [22] using the ResNet-101 encoder network. Our method can estimate accurate depths of thin structures.

Table 4. Evaluation results on the Make3D test dataset [45].

Method	Train	Error (lower is better)			
		Abs Rel	Sq Rel	RMSE	$\log_{10}$
Karsch [24]	D	0.425	4.948	8.290	0.151
Liu [31]	D	0.476	6.611	10.030	0.167
Laina [28]	D	**0.205**	**1.724**	**5.578**	**0.086**
Monodepth [15]	S	0.544	10.94	11.760	0.193
Zhou [63]	M	0.383	5.321	10.470	0.478
DDVO [49]	M	0.387	4.720	8.090	0.204
Monodepth2 [16]	M	0.322	3.589	7.417	0.163
Johnston [22]	M	0.306	3.100	7.126	0.160
Ours	M	0.285	2.798	6.950	0.147
Ours-refine	M	**0.283**	**2.974**	**6.949**	**0.146**

Table 5. Odometry evaluation results on testing sequences 9 and 10 of the KITTI odometry dataset. Results are average absolute trajectory error and standard deviation in meters.

Method	Seq. 09	Seq. 10	Frames
ORB-Slam [36]	0.014 ± 0.008	0.012 ± 0.001	-
DDVO [49]	0.045 ± 0.108	0.033 ± 0.074	3
Zhou [63]	0.050 ± 0.039	0.034 ± 0.028	$5 \rightarrow 2$
Zhou [63]	0.021 ± 0.017	0.020 ± 0.015	5
Mahjourian [34]	0.013 ± 0.010	0.012 ± 0.011	3
GeoNet [59]	0.012 ± 0.007	0.012 ± 0.009	5
Ranjian [44]	$\mathbf{0.012 \pm 0.007}$	$\mathbf{0.012 \pm 0.008}$	5
Monodepth2 [16]	0.017 ± 0.008	0.015 ± 0.010	2
Ours	$\mathbf{0.015 \pm 0.001}$	$\mathbf{0.012 \pm 0.004}$	2

methods. Fig. 10 shows our depth prediction results on the Make3D test dataset. These results demonstrate that the estimated depth is credible even though depth estimation model training did not use the Make3D dataset.

4.5. Odometry Evaluation

We evaluated the pose model to validate the effectiveness of the NCBAM model on pose. Following Monodepth2 [16], the training set of our pose network model was sequences 0–8 of the KITTI odometry dataset [14], and the test dataset was sequences 9 and 10. Generally, the pose network takes five frames as input [63, 44] and predicts transformations. Our pose network baseline is Monodepth2 [16]: the input to the pose network comprises two frame images, and the output is a relative pose transformation between that pair of frames. To evaluate the two-frame model on the five-frame test sequences, Monodepth2 [16] makes separate predictions for each of the four pairs of frame transformations for each set of five frames and combines them to form local trajectories. We follow the Monodepth2 [16] testing protocol to evaluate our pose network. Here, our pose model was trained for 11 epochs. Evaluation results are shown in Tab. 5: the accuracy of our pose model is better than that of Monodepth2 [16], and show that the NCBAM can enhance the results of the pose model.

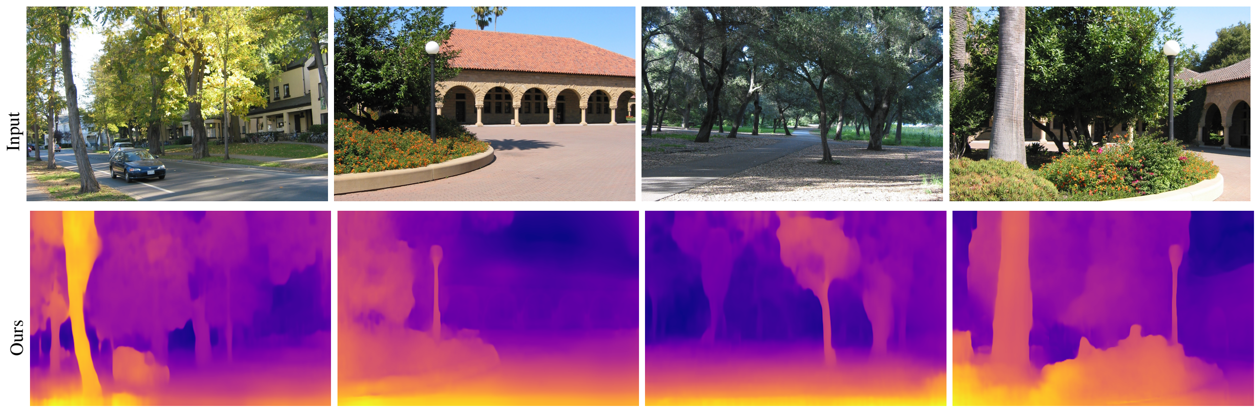


Figure 10. Examples of our approach on the Make3D dataset [45].

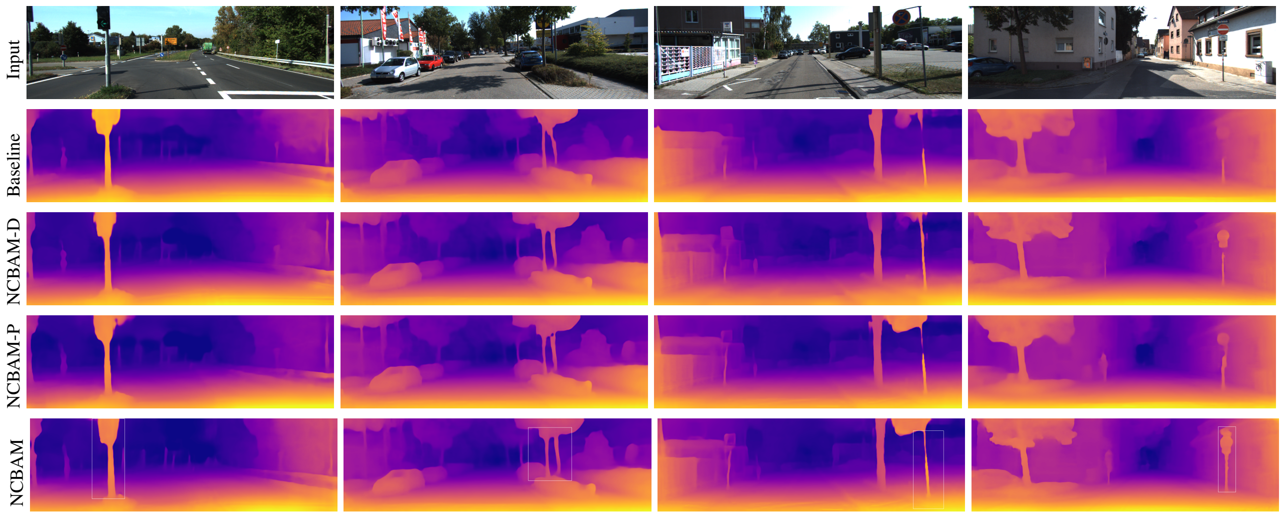


Figure 11. Ablation study. Incorporating NCBAM into both the depth and pose networks produces the best results.

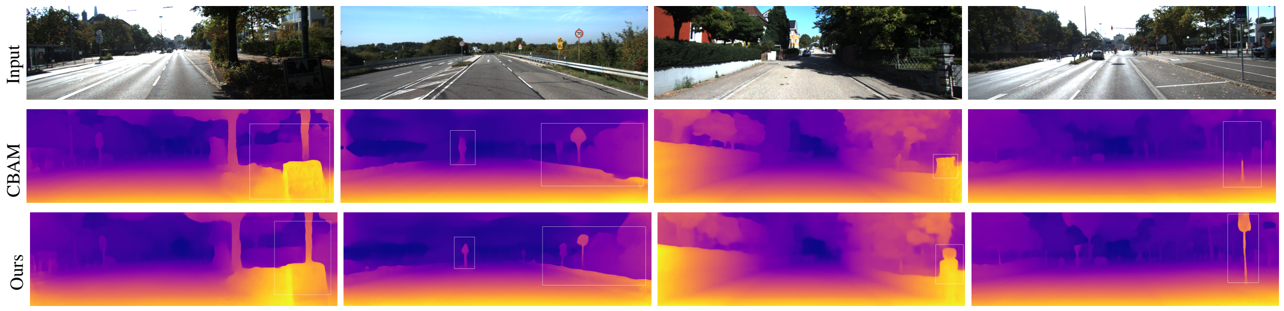


Figure 12. Comparison of CBAM and NCBAM. Incorporating NCBAM into the depth and pose networks produces better results than CBAM.

4.6. Ablation Study

To better validate the effectiveness of NCBAM in coarse depth estimation, we have performed an ablation study. Tab. 6 shows the results. We start from the baseline Monodepth2 [16] + DDV with ResNet-18 encoder network (1st row). Then, we incorporate the NCBAM module into the depth estimation network (2nd row), pose estimation network (3rd row), and depth and pose estimation networks (4th row), respectively. In each part of our method NCBAM improves evaluation measures. The results are significantly improved when adding the NCBAM module to both depth and pose networks. For comparison, we incorporate the CBAM module into the depth + pose network (5th row). The depth estimation model with NCBAM in row 4 is better than the one with CBAM on the seven evaluation metrics.

Table 6. **Ablation study**. Evaluation of the depth estimation model with CBAM and NCBAM on the Eigen split test dataset [6]. First, we evaluate the performance of the NCBAM module used in the depth network (NCBAM-D) and pose network (NCBAM-P). The baseline is Monodepth2 [16] with ResNet-18 + DDV. Then, we compare the effectiveness of the CBAM and NCBAM attention modules. The best results are marked in bold.

Method	Error (lower is better)				Accuracy (higher is better)		
	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Baseline	0.112	0.893	4.843	0.190	0.879	0.961	0.982
Baseline + NCBAM-D	0.104	0.835	4.705	0.182	0.884	0.960	0.982
Baseline + NCBAM-P	0.106	0.838	4.709	0.185	0.883	0.961	0.981
Baseline + (full)	**0.101**	**0.811**	**4.674**	**0.179**	**0.890**	**0.962**	**0.983**
Baseline + (full), CBAM	0.103	0.812	4.678	0.181	**0.890**	0.960	0.981

Table 7. Improvements due to two improvements in CBAM. The best results are marked in bold.

Method	Error (lower is better)				Accuracy (higher is better)		
	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
CBAM	0.103	0.812	4.678	0.181	**0.890**	0.960	0.981
CBAM_Tanh	0.102	**0.811**	4.676	0.180	**0.890**	0.960	0.982
CBAM_softplus	0.103	**0.811**	4.675	**0.179**	0.889	0.961	0.982
NCBAM	**0.101**	**0.811**	**4.674**	**0.179**	**0.890**	**0.962**	**0.983**
CBAM_sigmoid	0.104	0.812	4.677	0.183	0.890	0.959	0.981
CBAM_sigmoid_softplus	0.103	0.812	4.676	0.181	0.891	0.961	0.982

NCBAM can better aggregate deep features and extract correlations between the object and surrounding environment to improve depth estimation accuracy.

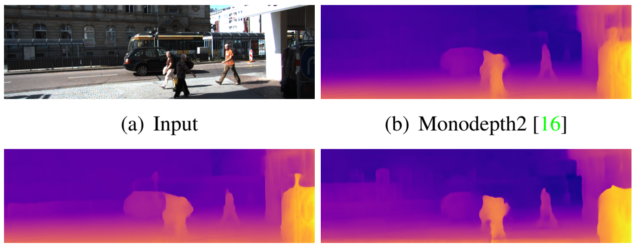
Fig. 11 shows qualitative results of the ablation study. The best results are those in which we incorporate NCBAM into depth and pose networks. In Fig. 12, we compare depth estimation models respectively with NCBAM and CBAM modules. Better results are provided by the version with NCBAM. Tab. 6, Fig. 11, and Fig. 12 show that the proposed NCBAM model is effective.

In Tab. 7, we illustrate the effectiveness of the improvements made to CBAM. The first converts the input feature to the range $(-1, 1)$ using the tanh function in the channel and spatial attention modules (CBAM_Tanh). The second is to use the activation function softplus instead of relu in the shared network of the channel sub-module and the convolution layer of the spatial sub-module (CBAM_softplus). We can see that both changes to CBAM improve depth estimation, and the best results are those when we utilize both improvements simultaneously, i.e. NCBAM.

The normalization sigmoid activation $\text{sigmoid}(x) = 1/1 + e^{-x}$ maps the input to the range (0,1). Tab. 7 compares use of tanh and sigmoid functions for normalization, and shows that tanh function produces better results.

4.7. Limitations

Although our method can overcome the above three targeted problems, our approach also has some limitations in common with other methods. One is that it cannot effectively predict the depths of moving objects. Fig. 13 presents an example generated by our approach and other state-of-the-art self-supervised monocular depth estimation methods. Unfortunately, all methods fail to predict the person's depth since the training set KITTI dataset is collected in scenes nearly completely lacking in humans.


(a) Input (b) Monodepth2 [16]

(c) Klingner et al. [27] (d) Ours

Figure 13. Limitation. Our approach and other state-of-the-art methods fail to predict the depths of moving persons.

5. Conclusion

In this paper, we have presented a new self-supervised monocular depth estimation method. Previous methods typically produce predicted depth maps with incomplete structures, texture-copy issues, and depth drift problems. We improved the attention model CBAM, to provide NCBAM, incorporated it into networks, and proposed a coarse-to-fine approach to address the above problems. We have performed extensive experiments to compare our method to state-of-the-art methods which validate its effectiveness. In future, we will further investigate depth estimation in complex scenes containing motion objects.

Acknowledgments
This work is partially supported by the Key Technological Innovation Projects of Hubei Province (2018AAA062), NSFC (61972298), Wuhan University-Huawei GeoInformatics Innovation Lab. Chunxia Xiao is the corresponding author.

References

[1] F. Aleotti, F. Tosi, M. Poggi, and S. Mattoccia. Generative adversarial networks for unsupervised monocular depth prediction. In *ECCV*, 2019. 2

[2] Y. Cao, L. Kobbelt, and S. Hu. Real-time high-accuracy three-dimensional reconstruction with consumer rgb-d cameras. *ACM Transactions on Graphics*, 37(5):1–16, 2018. 1

[3] V. Casser, S. Pirk, R. Mahjourian, and A. Angelova. Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. In *AAAI*, 2019. 7

[4] Y. Chen, C. Schmid, and C. Sminchisescu. Self-supervised learning with geometric constraints in monocular video connecting flow, depth, and camera. In *ICCV*, 2019. 5

[5] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 8, 9

[6] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, 2015. 2, 6, 7, 8, 12

[7] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NeurIPS*, 2014. 2, 7

[8] X. Fan, W. Wu, L. Zhang, Q. Yan, G. Fu, Z. Chen, C. Long, and C. Xiao. Shading-aware shadow detection and removal from a single image. *The Visual Computer*, 36(10-12):2175–2188, 2020. 1

[9] Y. Fu, Q. Yan, J. Liao, and C. Xiao. Joint texture and geometry optimization for rgb-d reconstruction. In *CVPR*, 2020. 1

[10] Y. Fu, Q. Yan, J. Liao, H. Zhou, J. Tang, and C. Xiao. Seamless texture optimization for rgb-d reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 2021. 1

[11] Y. Fu, Q. Yan, L. Yang, J. Liao, and C. Xiao. Texture mapping for 3d reconstruction with rgb-d sensor. In *CVPR*, pages 4645–4653, 2018. 1

[12] R. Garg, B. G. VijayKumar, G. Carneiro, and I. Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *ECCV*, 2016. 2

[13] R. Garg, B. G. VijayKumar, G. Carneiro, and I. Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *ECCV*, 2016. 7

[14] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 6, 10

[15] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017. 1, 2, 4, 5, 7, 9, 10

[16] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow. Digging into self-supervised monocular depth estimation. In *ICCV*, 2019. 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12

[17] V. Guizilini, R. Ambrus, S. Pillai, A. Raventos, and A. Gaidon. 3d packing for self-supervised monocular depth estimation. In *CVPR*, 2020. 2, 7, 8, 9

[18] M. Guo, T. Xu, J. Liu, Z. Liu, P. Jiang, T. Mu, S. Zhang, R. R. Martin, M. Cheng, and S. Hu. Attention mechanisms in computer vision: A survey. *Computational Visual Media*, 8(35), 2022. 3

[19] X. Guo, H. Li, S. Yi, J. Ren, and X. Wang. Learning monocular depth by distilling cross-domain stereo networks. In *ECCV*, 2018. 1

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

[21] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks. In *NeurIPS*, 2015. 5

[22] A. Johnston and G. Carneiro. Self-supervised monocular trained depth estimation using self-attention and discrete disparity volume. In *CVPR*, 2020. 3, 7, 8, 9, 10

[23] K. Karsch, C. Liu, and S. B. Kang. Depth transfer: Depth extraction from videos using nonparametric sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2144–2158, 2014. 1

[24] K. Karsch, C. Liu, and S. B. Kang. Depthtransfer: Depth extraction from video using non-parametric sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2144–2158, 2014. 10

[25] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach, and A. Bry. End-to-end learning of geometry and context for deep stereo regression. In *ICCV*, 2017. 4

[26] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 8

[27] M. Klingner, J. A. Termöhlen, J. Mikolajczyk, and T. Fingscheidt. Self-supervised monocular depth estimation: Solving the dynamic object problem by semantic guidance. In *ECCV*, 2020. 1, 2, 3, 7, 12

[28] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. Deeper depth prediction with fully convolutional residual networks. In *3DV*, 2016. 10

[29] R. Li, S. Wang, Z. Long, and D. Gu. Undeepvo: Monocular visual odometry through unsupervised deep learning. In *ICRA*, 2018. 7

[30] Z. Li, X. Liu, N. Drenkow, A. Ding, F. X. Creighton, R. H. Taylor, and M. Unberath. Revisiting stereo depth estimation from a sequence-to-sequence perspective with transformers. In *ICCV*, 2021. 3

[31] M. Liu, M. Salzmann, and X. He. Discrete-continuous depth estimation from a single image. In *CVPR*, 2014. 10

[32] C. Luo, Z. Yang, P. Wang, Y. Wang, W. Xu, R. Nevatia, and A. Yuille. Every pixel counts ++: Joint learning of geometry

and motion with 3d holistic understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(10), 2020. 7

[33] H. Luo, Y. Gao, Y. Wu, C. Liao, X. Yang, and K. Cheng. Real-time dense monocular slam with online adapted depth prediction network. *IEEE Transactions on Multimedia*, 21(2):470–483, 2019. 1

[34] R. Mahjourian, M. Wicke, and A. Angelova. Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints. In *CVPR*, 2018. 2, 7, 10

[35] I. Mehta, P. Sakurikar, and P. J. Narayanan. Structured adversarial training for unsupervised monocular depth estimation. In *3DV*, 2018. 7

[36] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. Orb-slam: A versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015. 10

[37] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *NeurIPS*, 2017. 8

[38] S. Pillai, R. Ambrus, and A. Gaidon. Superdepth: Self-supervised, super-resolved monocular depth estimation. In *ICRA*, 2019. 7

[39] A. Pilzer, D. Xu, M. M. Puscas, E. Ricci, and N. Sebe. Unsupervised adversarial depth estimation using cycled generative networks. In *3DV*, 2018. 2

[40] M. Poggi, F. Tosi, and S. Mattoccia. Learning monocular depth estimation with unsupervised trinocular assumptions. In *3DV*, 2018. 7

[41] M. Ramamonjisoa, M. Firman, J. Watson, V. Lepetit, and D. Turmukhambetov. Single image depth prediction with wavelet decomposition. In *CVPR*, 2021. 7, 8, 9

[42] M. Ramamonjisoa and V. Lepetit. Sharpnet: Fast and accurate recovery of occluding contours in monocular depth estimation. In *ICCV*, 2019. 6

[43] R. Ranftl, A. Bochkovskiy, and V. Koltun. Vision transformers for dense prediction. In *ICCV*, pages 12179–12188, 2021. 3

[44] A. Ranjan, V. Jampani, L. Balles, K. Kim, D. Sun, J. Wulff, and M. J. Black. Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In *CVPR*, 2019. 1, 7, 10

[45] A. Saxena, M. Sun, and A. Y. Ng. Make3d: Learning 3d scene structure from a single still image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5):824–840, 2009. 9, 10, 11

[46] J. L. Schönberger and J. M. Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 2

[47] F. Tosi, F. Aleotti, M. Poggi, and S. Mattoccia. Learning monocular depth estimation infusing traditional stereo knowledge. In *CVPR*, 2019. 7

[48] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 3

[49] C. Wang, J. M. Buenaposada, R. Zhu, and S. Lucey. Learning depth from monocular videos using direct methods. In *CVPR*, 2018. 5, 7, 10

[50] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 5

[51] J. Watson, O. M. Aodha, D. Turmukhambetov, G. J. Brostow, and M. Firman. Learning stereo from single images. In *ECCV*, 2020. 1

[52] J. Watson, M. Firman, G. Brostow, and D. Turmukhambetov. Self-supervised monocular depth hints. In *ICCV*, 2019. 7

[53] S. Woo, J. Park, J. Y. Lee, and I. S. Kweon. Cbam: Convolutional block attention module. In *ECCV*, 2018. 1, 3, 5

[54] J. Xie, R. B. Girshick, and A. Farhadi. Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks. In *ECCV*, 2016. 2

[55] G. Yang, H. Tang, M. Ding, N. Sebe, and E. Ricci. Transformer-based attention networks for continuous pixel-wise prediction. In *ICCV*, 2021. 3

[56] L. Yang, Q. Yan, Y. Fu, and C. Xiao. Surface Reconstruction via Fusing Sparse-Sequence of Depth Images. *IEEE Transactions on Visualization and Computer Graphics*, 24(2):1190–1203, 2018. 1

[57] Z. Yang, P. Wang, Y. Wang, W. Xu, and R. Nevatia. Lego: Learning edge with geometry all at once by watching videos. In *CVPR*, 2018. 1, 3, 7

[58] Z. Yang, P. Wang, W. Xu, L. Zhao, and R. Nevatia. Unsupervised learning of geometry from videos with edge-aware depth-normal consistency. In *AAAI*, 2018. 7

[59] Z. Yin and J. Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *CVPR*, 2018. 2, 3, 7, 10

[60] Y. Yuan, L. Huang, J. Guo, C. Zhang, X. Chen, and J. Wang. Ocnet: Object context network for scene parsing. *International Journal of Computer Vision*, 129(407):23752398, 2021. 3

[61] H. Zhao, O. Gallo, I. Frosio, and J. Kautz. Loss functions for image restoration with neural networks. *IEEE Transactions on Computational Imaging*, 3(1):47–57, 2017. 6

[62] W. Zhao, S. Liu, Y. Shu, and Y. Liu. Towards better generalization: Joint depth-pose learning without posenet. In *CVPR*, pages 9148–9158, 2020. 1

[63] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017. 1, 2, 6, 7, 10

[64] S. Zhu, G. Brazil, and X. Liu. The edge of depth: Explicit constraints between segmentation and depth. In *CVPR*, 2020. 3

[65] Y. Zou, Z. Luo, and J. Huang. Df-net: Unsupervised joint learning of depth and flow using cross-task consistency. In *ECCV*, 2018. 7