

Article

MonoLENS: Monocular Lightweight Efficient Network with Separable Convolutions for Self-Supervised Monocular Depth Estimation [†]

Genki Higashiuchi ¹, Tomoyasu Shimada ¹ , Xiangbo Kong ^{2,*}, Haimin Yan ¹ and Hiroyuki Tomiyama ^{1,*}

¹ Graduate School of Science and Engineering, Ritsumeikan University, Kusatsu 525-8577, Shiga, Japan; ri0117vi@ed.ritsumei.ac.jp (G.H.); ri0080xs@ed.ritsumei.ac.jp (T.S.); rm0215pk@ed.ritsumei.ac.jp (H.Y.)

² Department of Intelligent Robotics, Faculty of Information Engineering, Toyama Prefectural University, Imizu 939-0398, Toyama, Japan

* Correspondence: kong@pu-toyama.ac.jp (X.K.); ht@fc.ritsumei.ac.jp (H.T.)

[†] Presented at the Forum on Information Technology, Information Processing Society of Japan, Hokkaido, Japan, 2–5 September 2025.

Abstract

Self-supervised monocular depth estimation is gaining significant attention because it can learn depth from video without needing expensive ground-truth data. However, many self-supervised models remain too heavy for edge devices, and simply shrinking them tends to degrade accuracy. To address this trade-off, we present MonoLENS, an extension of Lite-Mono. MonoLENS follows a design that reduces computation while preserving geometric fidelity (relative depth relations, boundaries, and planar structures). MonoLENS advances Lite-Mono by suppressing computation on paths with low geometric contribution, focusing compute and attention on layers rich in structural cues, and pruning redundant operations in later stages. Our model incorporates two new modules, the DS-Upsampling Block and the MCACoder, along with a simplified encoder. Specifically, the DS-Upsampling Block uses depthwise separable convolutions throughout the decoder, which greatly lowers floating-point operations (FLOPs). Furthermore, the MCACoder applies Multidimensional Collaborative Attention (MCA) to the output of the second encoder stage, helping to make edge details sharper in high-resolution feature maps. Additionally, we simplified the encoder's architecture by reducing the number of blocks in its fourth stage from 10 to 4, which resulted in a further reduction of model parameters. When tested on both the KITTI and Cityscapes benchmarks, MonoLENS achieved leading performance. On the KITTI benchmark, MonoLENS reduced the number of model parameters by 42% (1.8M) compared with Lite-Mono, while simultaneously improving the squared relative error by approximately 4.5%.

Keywords: monocular depth estimation; self-supervised learning; lightweight neural networks; depthwise separable convolutions; multidimensional collaborative attention



Academic Editors: Miguel Angel Patricio and Luis Usero Aragonés

Received: 15 August 2025

Revised: 20 September 2025

Accepted: 22 September 2025

Published: 25 September 2025

Citation: Higashiuchi, G.; Shimada, T.; Kong, X.; Yan, H.; Tomiyama, H. MonoLENS: Monocular Lightweight Efficient Network with Separable Convolutions for Self-Supervised Monocular Depth Estimation. *Appl. Sci.* **2025**, *15*, 10393. <https://doi.org/10.3390/app151910393>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Depth estimation, the process of predicting the distance for each pixel in an image, is fundamental to numerous computer vision applications, including 3D reconstruction [1] and autonomous driving [2]. While state-of-the-art depth sensors like RGB-D cameras, LiDAR, and structured light systems provide highly accurate depth maps [3], they often present significant disadvantages, including high cost, a large form factor, and considerable

power consumption. On the other hand, stereo cameras infer depth through pixel matching. However, this approach demands substantial computational resources and powerful processors. Furthermore, minor temporal or spatial misalignments between the two cameras can lead to error accumulation and degraded performance in practical deployments.

Monocular depth estimation, which infers depth from a single RGB image, offers a cost-effective and easily deployable solution, as it requires no specialized hardware. The pioneering work by Eigen et al. [4], which demonstrated a two-stage Convolutional Neural Network (CNN) for coarse global depth prediction followed by local refinement, spurred rapid advancements in deep-learning-based monocular depth methods. Supervised depth estimation relies heavily on high-quality ground-truth data during training, thereby limiting accuracy due to dataset availability and collection costs. In contrast, self-supervised approaches eliminate the need for ground-truth labels. Most initial self-supervised methods utilized calibrated stereo image pairs for depth learning. Although stereo-based self-supervision can achieve accuracy comparable to supervised methods, it still relies on dual-camera setups, limiting its ability to fully utilize abundant monocular video data.

Self-supervised monocular depth estimation, trained exclusively on single-camera videos, dramatically reduces data collection costs and enables compatibility with a wide range of applications, such as human–computer interaction [5] and novel view synthesis [6]. Influential works like Monodepth2 [7] further enhanced robustness by incorporating photometric reprojection and auto-masking losses to effectively handle occlusions and dynamic objects. Our proposed MonoLENS adopts this video-based learning framework. However, the pursuit of higher accuracy has made modern depth models deeper and wider. This has increased their model parameters and computational demands. As a result, deploying these intensive models on edge devices remains a significant challenge because edge devices have limited memory and processing capabilities [8]. Accordingly, a design is required that maintains depth estimation accuracy on edge devices while substantially reducing model parameters and inference time.

As shown in Figure 1, we present MonoLENS, a lightweight self-supervised monocular depth estimation framework built upon Lite-Mono [9] that achieves the best balance of model size and performance. This article is a revised and expanded version of [10]. It extends the original model with two novel modules, the DS-Upsampling Block and the MCACoder, and incorporates a simplified encoder. Specifically, the DS-Upsampling Block utilizes depthwise separable convolutions [11] throughout the decoder, significantly reducing both model parameters and FLOPs. And the MCACoder applies an MCA [12] to the output of the second encoder stage to sharpen edge details in high-resolution feature maps. Finally, we streamline the encoder by reducing its fourth stage from 10 blocks to 4, thereby further trimming model parameters and computation. The key contributions of this paper are summarized as follows:

- We introduce MonoLENS, a novel hybrid architecture for lightweight, self-supervised monocular depth estimation. We demonstrate its effectiveness by showing substantial reductions in both model parameters and FLOPs compared with baseline architectures.
- We show that MonoLENS achieves superior accuracy on the KITTI dataset [13] and Cityscapes [14] when compared with much larger competing models.
- The inference time of the proposed method is evaluated on NVIDIA Jetson Orin Nano platforms, demonstrating its favorable trade-off between model complexity and inference speed.

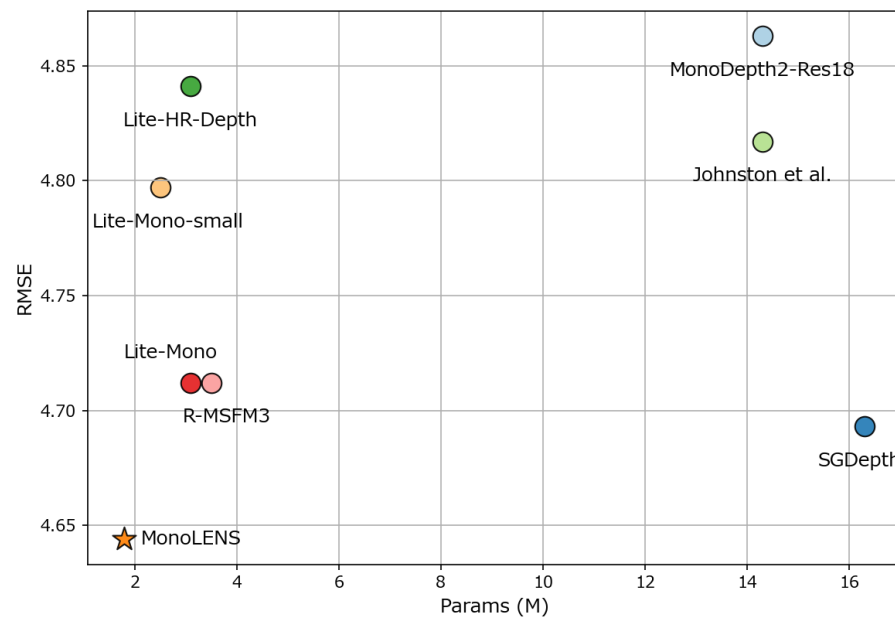


Figure 1. This scatter plot compares model size and depth estimation error (RMSE) for monocular depth methods under 20M parameters. MonoLENS achieves the lowest RMSE with the lowest model parameters [15].

2. Related Work

2.1. Deep Learning–Based Monocular Depth Estimation

Estimating depth from a single 2D image is fundamentally challenging, primarily because an infinite number of 3D scenes can correspond to the same 2D projection. In recent years, however, approaches based on deep learning have made significant advancements in addressing this ill-posed problem. Deep learning–based monocular depth estimation methods can be broadly categorized into supervised learning and self-supervised learning.

2.2. Supervised Monocular Depth Estimation

In supervised monocular depth estimation, the network is trained using ground-truth depth maps as its direct supervisory signal. The objective is for the network to learn robust features from the input image and to accurately map RGB values to corresponding depth values. Eigen et al. [4] pioneered this field by combining global coarse-scale and local fine-scale predictions within a multi-scale architecture, introducing a scale-invariant error metric to achieve high-precision depth estimates. Subsequently, Laina et al. [16] adopted the reverse Huber (BerHu) loss for model optimization. Meanwhile, Fu et al. [17] reframed depth estimation as an ordinal regression problem, proposing a two-stage multi-scale network that achieved both high accuracy and fast convergence. More recently, Lee et al. [18] introduced BTS, which incorporated multi-stage local planar guidance layers into the decoder. Another notable contribution came from Bauer et al. [19], who introduced NVS MonoDepth, which integrated a novel consistency constraint into the supervisory signal. Additionally, Bhat et al. [20] introduced AdaBins, featuring a dynamic binning scheme that adapts depth ranges according to scene features. Despite their impressive accuracy, supervised models inherently necessitate expensive ground-truth depth data, which has motivated researchers to explore alternative methods that do not rely on real-world depth labels.

2.3. Self-Supervised Monocular Depth Estimation

Self-supervised depth estimation methods are typically divided into two main categories. One category consists of stereo-matching methods [21,22], and the other consists of

temporal sequence methods. Stereo-matching methods draw inspiration from traditional stereo vision. For self-supervised learning, it is common practice to utilize left-right image pairs for depth estimation [23]. Garg et al. [22] initiated this direction by training on stereo pairs with a reprojection loss, and Godard et al. [21] introduced MonoDepth, which further improved accuracy by enforcing left-right disparity consistency. Poggi et al. [24] presented a method that used a multi-camera setup to reduce occlusion effects. Watson et al. [25] improved photometric losses with depth hints from existing stereo algorithms. Other advanced methods include Gonzalez-Bello [26], who brought in the FAL net to create occlusion masks using a mirror-occlusion module. Zhu et al. [27] introduced EdgeDepth, which combined semantic segmentation. And Peng et al. [28] developed EPDepth, which got better results with edge-based graph filtering.

On the other hand, temporal sequence methods utilize monocular video frames. Zhou et al. [29] introduced SfM-Learner, which jointly learned depth and camera pose. Later, Godard et al. [7] presented Monodepth2, addressing dynamic objects and occlusions through minimum reprojection error and auto-masking. Shu et al. [30] introduced FeatDepth, featuring a feature-distance loss, and Lyu et al. [31] introduced HR Depth, which redesigned skip connections to retain high-resolution features. However, a common limitation of many of these methods is their large model sizes and slow inference speeds, which prevent their deployment in real-time applications like autonomous driving.

2.4. *Lightweight Models for Depth Estimation*

There is a growing demand for monocular depth models capable of real-time execution on resource-constrained hardware. Researchers are actively striving to match or exceed the accuracy of large models while consuming significantly fewer computational resources. For example, Yin et al. [32] enhanced depth accuracy and 3D reconstruction by incorporating a virtual surface-normal term into the loss function. Wofk et al. [33] introduced FastDepth, which combined an encoder–decoder structure with network pruning to dramatically reduce model parameters and inference latency, thereby enabling high-precision real-time estimation on embedded devices. Nekrasov et al. [34] utilized semantic-segmentation training and knowledge distillation to transfer complex large-model structures into a lightweight model without sacrificing accuracy. Hu et al. [35] further boosted performance by fusing distillation with external auxiliary data within a compact real-time network. Sheng et al. [36] introduced the Distribution Alignment Network, which dynamically corrected depth distributions between large and small models, achieving high accuracy at low cost. Nevertheless, these methods often still depend on high-quality ground-truth labels or additional training tasks, which limits their general applicability and increases data-collection costs.

Consequently, lightweight self-supervised models have emerged as a promising direction. For instance, Zhou et al. [37] introduced R-MSFM, which utilizes the first three stages of ResNet-18 [38] as its backbone and maintains multi-scale learning through a feature-modulation module, significantly reducing model parameters. Hoang et al. [39] introduced PydNet, which relies solely on photometric reprojection error from stereo pairs as its supervisory signal, resulting in an extremely compact network capable of running on a CPU. Moreover, purely CNN-based models, constrained by limited receptive fields, can struggle to capture long-range dependencies and global context. Consequently, hybrid architectures that combine lightweight Transformers with CNNs have been proposed to pair CNNs' efficient local inductive biases with Transformers' global modeling capacity. For example, Varma et al. [40] introduced MT-SFM-Learner, a hybrid architecture that combines the local feature extraction of CNNs with the global context capture of Transformers. This approach demonstrated that Transformer-based depth estimation is more robust to image

degradation and adversarial attacks than CNNs, but it also highlighted the efficiency challenges due to the high computational cost of its Transformer components. Another instance is Zhao et al. [41], who introduced MonoViT, which employed MPViT [42] as its encoder to achieve state-of-the-art accuracy, but multiple parallel blocks can still lead to slower inference. Zhang et al. [9] introduced Lite-Mono, which integrated dilated convolutions into convolutional layers to create a hybrid CNN–Transformer model that enhances feature extraction without sacrificing efficiency. However, the Multi-Head Self-Attention (MSHA) modules used in Lite-Mono still pose a bottleneck for fast inference, thus motivating the need for more efficient architectural designs [43].

To address these issues, we aim to develop an even more efficient monocular depth estimation framework that achieves reduced inference time and a lightweight model without compromising high accuracy.

3. Proposed Method

This section provides a comprehensive overview of the proposed MonoLENS framework. We first detail its three core design innovations. These are the DS-Upsampling Block, the MCACoder, and our encoder-depth reduction strategy. Following these motivational insights, we describe the full model architecture and then detail each of its components.

3.1. Design Motivation

3.1.1. Decoder Efficiency: The DS-Upsampling Block

Current monocular depth estimation networks tend to prioritize the fine details of depth prediction, often neglecting model size and inference speed [43]. Even in models like Lite-Mono [9], which are designed to be efficient, their decoders commonly stack many 3×3 convolutions to produce fine-grained depth maps. However, this approach dramatically increases FLOPs, making real-time inference on edge devices challenging.

To overcome these issues, we introduce the DS-Upsampling Block that replaces conventional 3×3 convolutions with depthwise separable convolutions, as popularized by MobileNet [44]. In this block, depthwise convolutions efficiently extract local features within each channel, and 1×1 pointwise convolutions then fuse information across channels. This design, therefore, drastically reduces FLOPs compared with the traditional approach, while simultaneously preserving local precision and enabling real-time inference in resource-constrained environments.

3.1.2. Skip Connection Refinement: The MCACoder

Existing lightweight monocular depth estimation models, especially architectures like Lite-Mono [9], are designed to optimize inference speed and computational efficiency. However, this lightweight design often faces challenges in preserving detailed information, particularly edges and fine textures in high-resolution images. This is because aggressive downsampling in the encoder and simple concatenation of features in skip connections to the decoder can lead to the loss or degradation of these crucial high-frequency components. For instance, downsampling is known to reduce spatial resolution, causing the loss of fine structures and details at object boundaries [45,46].

To overcome these issues, we introduce the MCACoder to refine high-resolution skip connection features. While powerful channel-spatial attention mechanisms, such as CBAM [47], can enhance performance, their reliance on 2D convolutions on large feature maps often results in substantial computational costs. Instead, our module applies MCA to the high-resolution features from the second encoder stage. By summarizing statistics through average and standard deviation pooling and generating gates via lightweight $1 \times k$ 1D convolutions, it learns attention along the height, width, and channel axes with

minimal extra model parameters. Applied before skip connections, this module accurately preserves edge and texture details.

3.1.3. Encoder Optimization: Fourth Stage Compression

While increasing the number of blocks in the encoder, particularly by adding more layers, is crucial for capturing broader contextual information and improving depth prediction accuracy [4], this architectural growth inevitably leads to a significant increase in model size and FLOPs. Consequently, the inference speed is severely slowed down, making real-time applications such as autonomous driving and robotics challenging. This performance bottleneck necessitates a novel approach to encoder design that can achieve a balance between robust feature extraction and computational efficiency.

To address this challenge, we structurally reduce the encoder's fourth stage. Our approach quantitatively evaluates each layer's contribution to both depth estimation errors and computational efficiency, the latter being measured by FLOPs and model parameters. We then identify layers with overlapping receptive fields and remove six blocks that show a minimal contribution to accuracy while consuming significant resources. This process allows us to compress the fourth stage from 10 down blocks to 4, effectively balancing performance and efficiency.

3.2. Model Architecture

Inspired by Lite-Mono [9], we propose MonoLENS, a more lightweight depth estimation framework. This framework consists of two main components, as shown in Figure 2. These are the DepthNet, which is the depth estimation network, and the PoseNet, which is the camera motion estimation network. The goal of this self-supervised learning approach is to jointly train these two networks by measuring and optimizing the reconstruction error of the target images.

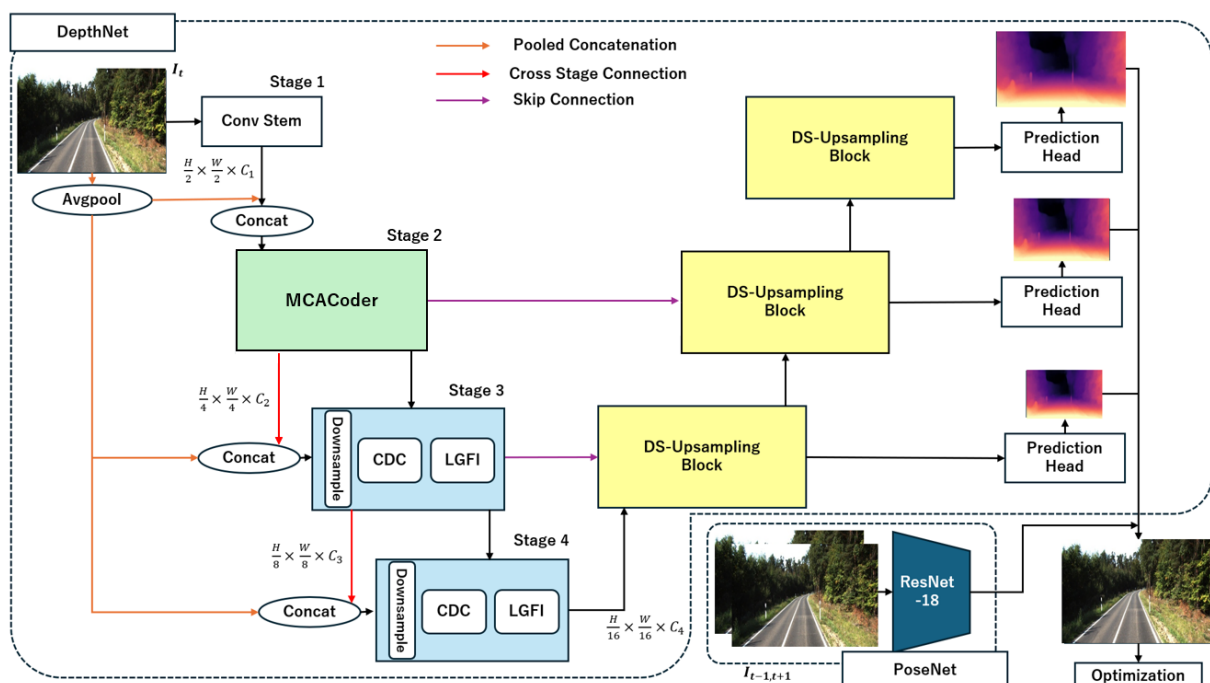


Figure 2. Overview of the proposed MonoLENS [10].

3.3. Encoder

The depth encoder aggregates multi-scale features across four stages. The input image with size $H \times W \times 3$ is first fed into a Conv Stem, where the image is downsampled by a 3×3 convolution. Following two additional 3×3 convolutions with stride = 1 for local feature extraction, feature maps of size $\frac{H}{2} \times \frac{W}{2} \times C_1$ are obtained.

Subsequently, this feature map of size $\frac{H}{2} \times \frac{W}{2} \times C_1$ is again downsampled by a 3×3 convolution with a stride of 2, constructing a feature map of size $\frac{H}{4} \times \frac{W}{4} \times C_2$. Following this, as in Figure 3a, Consecutive Dilated Convolutions (CDC) and Local-Global Features Interaction (LGFI), both proposed in Lite-Mono [9], are applied. CDC is a module that utilizes dilated convolutions to extract multi-scale local features. On the other hand, LGFI is a module that enhances the interaction between features and increases nonlinearity using an attention mechanism. However, the CDC and LGFI pipelines suffer from reduced spatial resolution caused by downsampling, which leads to the loss of fine structures and details at object boundaries. To address this issue and refine high-resolution skip connection features, we introduce the MCACoder as in Figure 3b. Within the MCACoder, after passing through the CDC and LGFI pipelines, MCA is applied. MCA captures complex interactions across the W , H , and C dimensions of the input feature map using three parallel branches. This approach significantly boosts high-resolution image processing without the heavy computational cost of full-resolution attention. Specifically, see Figure 3c; the left branch captures interactions within the spatial W dimension, the middle branch captures interactions within the spatial H dimension, and the right branch focuses on inter-channel interactions. In the left and middle branches, the feature map is reoriented using a permutation operation to effectively capture long-range dependencies between the channel and spatial dimensions. After applying Squeeze and Excitation transformations to each branch, the resulting attention weights are multiplied with the rotated feature map element-wise to enhance the feature representation. Finally, the outputs from all branches are aggregated through simple averaging to create the final attention-enhanced feature map, which retains the same shape as the original input. This structure allows the MCA to effectively handle multidimensional feature interactions with almost no additional parameter overhead, thereby improving edge sharpness. Subsequently, a 1×1 projection halves the channel count, reducing both the number of model parameters and the computational load while preserving fine details that would otherwise be lost at lower resolutions. By placing the MCA at the output of the second encoder stage, we refine the features from the shallow skip connection. Shallow skip features primarily retain high-frequency components such as object boundaries, thin lines, and texture edges, but are also prone to containing undesirable artifacts like texture copy and lighting-dependent pseudo-disparities. By applying W , H , and C -wise axial gating based on global mean and variance statistics, the MCA emphasizes locally useful cues while suppressing disruptive components. Applying this directly before the skip-decoder fusion ensures that high-resolution information is filtered from the very beginning of the restoration process, thereby preventing the downstream propagation of boundary blur, halos, and texture-based misestimations. In contrast, applying a similar axial gating to deep-level skips offers a smaller gain in local refinement. Deep features have low spatial resolution and are already highly contextualized by the broad receptive fields and attention mechanisms within the encoder, creating functional redundancy with the global representation.

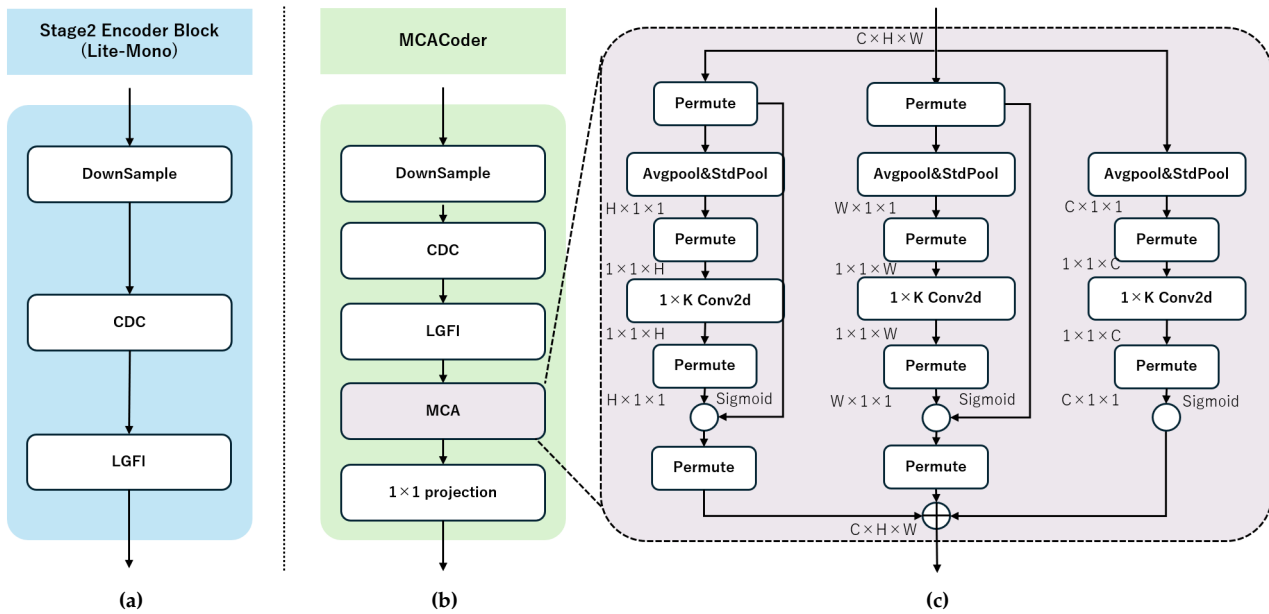


Figure 3. (a) Lite-Mono Stage-2 encoder block, (b) the proposed MCACoder, and (c) the MCA module details within MCACoder.

In the third and fourth stages, the output from the preceding stage is again down-sampled with a 3×3 stride-2 convolution. This output then passes through the CDC-LGFI pipeline to extract increasingly coarse but highly expressive feature maps of size $\frac{H}{8} \times \frac{W}{8} \times C_3$ and size $\frac{H}{16} \times \frac{W}{16} \times C_4$. We evaluated our design choices by quantitatively measuring model accuracy using depth error and efficiency using FLOPs and model parameters. For the encoder's fourth stage, we noticed a bottleneck where it added little to the model's overall accuracy but still used many model parameters. To fix this and substantially improve computational efficiency, we reduced the number of blocks in this stage from the 10 used in Lite-Mono [9] to just 4. In each of these stages, newly generated multi-scale pooled maps are concatenated to reinforce global context, and cross-stage connections propagate intermediate features forward. The resulting set of multi-scale features is then sent to the depth decoder via skip connections, allowing the model to perform lightweight, high-speed inference that retains both fine-grained detail and broad contextual information.

3.4. Decoder

The depth decoder restores the spatial resolution of the features obtained from the encoder. Restoration proceeds stage by stage, doubling the height and width of the feature maps at each step. First, Figure 4a shows the Lite-Mono Upsampling Block, which places a Conv Block both before and after the upsampling operation. Each Conv Block consists of a 3×3 convolution followed by an ELU activation. By contrast, Figure 4b illustrates the proposed DS-Upsampling Block, which serves as the core of each restoration step. In this block, the incoming features are first compressed by a Conv Block, then upsampled by bilinear interpolation, and finally refined by a depthwise separable convolution placed after upsampling. This design allows the model to first increase the feature map's resolution and integrate detailed information from the skip connections. Subsequently, the depthwise separable convolution refines this high-resolution feature map directly. This is crucial for effectively optimizing important boundaries and fine details added during the upsampling process, which enhances the model's accuracy. Placing the depthwise separable convolution before upsampling would have caused the refinement to happen at a lower resolution, leading to a loss of critical details before they could be learned. Therefore, our model adopts

the sequence of a Conv Block, followed by upsampling, and then a depthwise separable convolution.

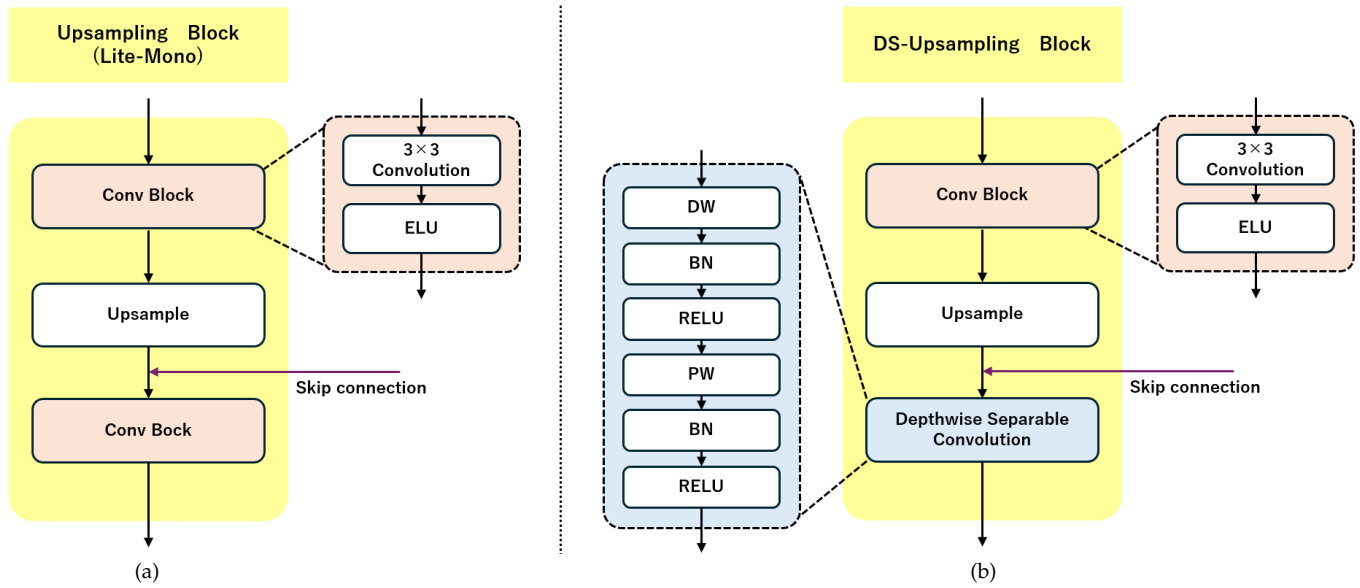


Figure 4. (a) Lite-Mono Upsampling Block and (b) the proposed DS-Upsampling Block used in the Depth Decoder.

The depthwise separable convolution operates in two parts. First, a depthwise convolution applies a separate 3×3 filter to each channel. Then, a pointwise (1×1) convolution combines information across channels. In our system, each DS-Upsampling Block performs the sequence of depthwise convolution, followed by BatchNorm, then ReLU, then pointwise convolution, another BatchNorm, and finally another ReLU as a single unit. To further quantitatively demonstrate the computational superiority of our method, we compare the parameters and computational cost of a standard 3×3 convolution with a depthwise separable convolution. We assume the number of input channels is C_{in} , the number of output channels is C_{out} , the kernel size is K (where $K = 3$ in this paper), and the spatial size after upsampling is $H \times W$. Bias terms are omitted, as they are assumed to be handled by BatchNorm. The parameter counts are given by Equation (1) for a standard 3×3 and Equation (2) for a depthwise separable layer, while the computational costs (MACs) are given by Equations (3) and (4), respectively.

$$Params_{conv} = K^2 C_{in} C_{out} \quad (1)$$

$$Params_{dwsep} = K^2 C_{in} + C_{in} C_{out} \quad (2)$$

$$MACs_{conv} \propto HW K^2 C_{in} C_{out} \quad (3)$$

$$MACs_{dwsep} \propto HW (K^2 C_{in} + C_{in} C_{out}) \quad (4)$$

A standard 3×3 convolution performs spatial feature extraction and inter-channel mixing simultaneously, so both the parameter count and the computational cost grow multiplicatively with $K^2 C_{in} C_{out}$ (Equations (1) and (3)). In contrast, a depthwise separable convolution factorizes the operation into depthwise (spatial) and pointwise (channel mixing) steps, replacing the cost with the additive sum $K^2 C_{in} + C_{in} C_{out}$ (Equations (2) and (4)). This structural difference systematically reduces the required parameters and operations even for the same C_{in} and C_{out} . Moreover, because the parameter count is independent of the spatial size (H, W), substituting a depthwise separable layer for a standard 3×3 directly lowers the number of parameters. For computation, applying this substitution at the high-resolution stages after upsampling effectively lightens the work that scales with

(H, W) , yielding large FLOPs reductions while preserving local accuracy. Moreover, each DS-Upsampling Block is followed by a prediction head that outputs inverse depth maps at full, $\frac{1}{2}$, and $\frac{1}{4}$ resolution, respectively.

3.5. PoseNet

Following [7,37], we employ a standard PoseNet for camera-pose estimation. Specifically, a pretrained ResNet-18 [38] serves as the pose encoder. This encoder takes two consecutive RGB frames as input, which are concatenated along the channel axis. The encoder's output is subsequently fed into a four-layer convolutional pose decoder. This decoder predicts the 6DoF relative pose between adjacent frames.

3.6. Self-Supervised Learning

Following [9], we formulate monocular depth estimation as an image reconstruction task. During training, the network minimizes a combination of photometric reprojection loss and edge-aware smoothness loss in a self-supervised fashion.

The synthesized target image $\hat{I}_t$ is generated from a source frame I_s (either the previous or next frame) using a warping function F , based on the predicted depth map D_t , estimated camera pose P , and camera intrinsics K :

$$\hat{I}_t = F(I_s, P, D_t, K) \quad (5)$$

The photometric reprojection loss L_p is computed as a weighted sum of Structural Similarity Index Measure (SSIM) and L1 loss between the synthesized image $\hat{I}_t$ and the target image I_t :

$$L_p(\hat{I}_t, I_t) = \alpha \frac{1 - \text{SSIM}(\hat{I}_t, I_t)}{2} + (1 - \alpha) \|\hat{I}_t - I_t\|_1 \quad (6)$$

The weight α is experimentally set to 0.85.

To account for occlusions and out-of-view areas, we adopt the minimum reprojection loss across multiple source frames:

$$L_p(I_s, I_t) = \min_{I_s \in \{\text{prev}, \text{next}\}} L_p(\hat{I}_t, I_t) \quad (7)$$

$$\mu = \min_{I_s \in \{\text{prev}, \text{next}\}} L_p(I_s, I_t) > \min_{I_s \in \{\text{prev}, \text{next}\}} L_p(\hat{I}_t, I_t) \quad (8)$$

The final image reconstruction loss is then given by

$$L_r(\hat{I}_t, I_t) = \mu \cdot L_p(I_s, I_t) \quad (9)$$

To encourage smoothness in the predicted inverse depth map d_t , while preserving image edges, we apply the following edge-aware smoothness loss:

$$L_{\text{smooth}} = |\partial_x d_t^*| e^{-|\partial_x I_t|} + |\partial_y d_t^*| e^{-|\partial_y I_t|} \quad (10)$$

Here, $d_t^* = d_t / \bar{d}_t$ denotes the inverse depth map normalized by its mean.

The final total loss is the average of the combined losses over three output scales $s \in \{1, \frac{1}{2}, \frac{1}{4}\}$:

$$L = \frac{1}{3} \sum_{s \in \{1, \frac{1}{2}, \frac{1}{4}\}} (L_r + \lambda L_{\text{smooth}}), \quad \lambda = 10^{-3} \quad (11)$$

4. Experiments

This section details our experimental methodology and presents the results. We begin by describing the dataset used for training and evaluation. Subsequently, we outline the implementation details, including hyperparameters, data augmentation techniques, and evaluation metrics. We then present the quantitative and qualitative results on the KITTI dataset and the Cityscapes dataset, followed by an analysis of model complexity and inference speed. Finally, we conduct comprehensive ablation studies to assess the individual contributions of the DS-Upsampling Block, the MCACoder, and the simplified encoder design.

4.1. Dataset

4.1.1. KITTI Dataset

We use the KITTI dataset [13], which comprises 61 stereo driving sequences captured with synchronized, rectified cameras, 3D LiDAR, GPS, and IMU by the Karlsruhe Institute of Technology and the Toyota Technological Institute at Chicago. Following the Eigen split [48], we train on 39,180 monocular image triplets, validate on 4424, and test on 697. All frames share a single set of camera intrinsics, with their focal length being the average over the entire dataset [7]. During evaluation, we clamp predicted depths to the range 0 m–80 m.

4.1.2. Cityscapes Dataset

We use the Cityscapes dataset [14]. Following [29,49,50], we train on 69,731 images extracted from monocular sequences, which we preprocess into triplets using the script from [29]. We do not use stereo pairs or semantic labels. We evaluate on the 1525 test images using the ground truth provided by [51]. As with KITTI, predicted depths are clipped to 0 m–80 m.

4.2. Implementation Details

Our method is implemented in PyTorch 2.1.0 and trained on a PC with an AMD Ryzen 9 5950X 16-Core Processor and an NVIDIA GeForce RTX 4090 graphical processing unit (GPU) with 24 GB RAM (NVIDIA Corporation, Santa Clara, CA, USA). The software environment consists of an Ubuntu 22.04 system and Python 3.10.13. Both the depth network and the pose estimation network are pretrained on ImageNet [52]. We use the AdamW optimizer [53]. For KITTI, we train for 30 epochs with a batch size of 12 and an initial learning rate of 1×10^{-4} . For Cityscapes, we train for 10 epochs with a batch size of 8, using the same initial learning rate.

For KITTI, we apply data augmentation during preprocessing to improve robustness. Specifically, horizontal flips and color augmentations are each applied with a 50% probability. Color augmentations include brightness, saturation, and contrast adjustments within [80–120%], and hue jitter within [−10%–+10%]. These are applied in random order, following the protocol in [7,31,37].

We use the seven standard metrics proposed by Eigen et al. [4] to report accuracy. These metrics consist of four error measures (Abs Rel, Sq Rel, RMSE, and RMSE log) and three accuracy thresholds ($\delta < 1.25$, $\delta < 1.25^2$, and $\delta < 1.25^3$). Their definitions follow.

$$\text{Abs Rel} = \frac{1}{|N|} \sum_{i \in N} \frac{|d_i - d_i^*|}{d_i^*} \quad (12)$$

$$\text{Sq Rel} = \frac{1}{|N|} \sum_{i \in N} \frac{\|d_i - d_i^*\|^2}{d_i^*} \quad (13)$$

$$\text{RMSE} = \sqrt{\frac{1}{|N|} \sum_{i \in N} \|d_i - d_i^*\|^2} \quad (14)$$

$$\text{RMSE log} = \sqrt{\frac{1}{|N|} \sum_{i \in N} \|\log(d_i) - \log(d_i^*)\|^2} \quad (15)$$

$$\text{Accuracies} = \max\left(\frac{d_i}{d_i^*}, \frac{d_i^*}{d_i}\right) = \delta < \text{threshold} \quad (16)$$

4.3. KITTI Results

The proposed framework is compared with other representative methods with model parameters (Params) less than 35 M, and the results are shown in Table 1. For Lite-Mono [9], Lite-Mono-small [9], and MonoLENS, we report results obtained from our own training and evaluation environment. For all other models, we use values reported in their respective papers. MonoLENS uses only 1.8 M parameters, which is approximately one-eighteenth the size of the largest model, Monodepth2-Res50 [7], making it extremely compact. Despite this compact model size, MonoLENS outperforms Monodepth2-Res18 [7] on every error and accuracy metric and matches or exceeds the performance of the much larger Monodepth2-Res50 [7]. It also achieves equal or better results than mid-sized models such as R-MSFM6 [37] and Lite-Mono [9]. In particular, MonoLENS improves the Sq Rel by around 4.5% over Lite-Mono [9]. These results demonstrate that MonoLENS achieves an optimal balance between model size and depth-estimation accuracy, making it highly suitable for real-time inference on resource-constrained edge devices.

Table 1. Comparison of evaluation metrics for each method on the KITTI Eigen split [48]. The best value for each metric is shown in **bold**, and the second best is underlined. “M”: trained on KITTI monocular videos; “M+Se”: trained on monocular videos with semantic segmentation.

Method	Data	Depth Error (Lower Is Better)				Depth Accuracy (Higher Is Better)			Params (M)
		Abs Rel	Sq Rel	RMSE	RMSE Log	δ_1	δ_2	δ_3	
GeoNet [50]	M	0.149	1.060	5.567	0.226	0.796	0.935	0.975	31.6
DDVO [54]	M	0.151	1.257	5.583	0.228	0.810	0.936	0.974	28.1
Monodepth [21]	M	0.148	1.344	5.927	0.247	0.803	0.922	0.964	20.2
EPC++ [55]	M	0.141	1.029	5.350	0.216	0.816	0.941	0.976	33.2
Struct2depth [56]	M	0.141	1.026	5.291	0.215	0.816	0.945	0.979	31.6
Monodepth2-Res18 [7]	M	0.115	0.903	4.863	0.193	0.877	0.959	<u>0.981</u>	14.3
Monodepth2-Res50 [7]	M	<u>0.110</u>	0.831	4.642	<u>0.187</u>	<u>0.883</u>	0.962	0.982	32.5
SGDepth [57]	M + Se	0.113	0.835	4.693	0.191	0.879	<u>0.961</u>	<u>0.981</u>	16.3
Johnston et al. [15]	M	0.111	0.941	4.817	0.189	0.885	<u>0.961</u>	<u>0.981</u>	14.3+
Lite-HR-Depth [31]	M	0.116	0.845	4.841	0.190	0.866	0.957	0.982	3.1
R-MSFM3 [37]	M	0.114	<u>0.815</u>	4.712	0.193	0.876	0.959	<u>0.981</u>	3.5
R-MSFM6 [37]	M	0.112	0.806	4.704	0.191	0.878	0.960	<u>0.981</u>	3.8
Lite-Mono [9]	M	0.109	0.872	4.712	<u>0.187</u>	0.885	<u>0.961</u>	0.982	3.1
Lite-Mono-small [9]	M	0.112	0.896	4.797	0.189	0.879	0.960	<u>0.981</u>	<u>2.5</u>
MonoLENS (ours)	M	<u>0.110</u>	0.833	<u>4.644</u>	0.185	<u>0.883</u>	0.962	0.982	1.8

The results in Figure 5 clearly show how well our new method works for monocular depth estimation. We put MonoLENS’s depth predictions next to those from Monodepth2 [7] and Lite-Mono [9] for comparison. Look at the first three rows, which show thin, rod-like structures. Here, Monodepth2 [7] and Lite-Mono [9] often create lines that are faint or broken. However, MonoLENS keeps the object shapes continuous and correctly shows fine details. In the fourth and fifth rows, which display traffic signs, the other methods make the edge between the sign and background blurry. But MonoLENS clearly separates the sign from what’s around it and correctly captures its outlines. These comparisons together tell us that MonoLENS is stronger and more accurate than current methods when estimating the depth of very thin structures and small objects. A main

reason for these good results is our proposed MCACoder. The MCA module inside the MCACoder picks out important features. It does this by gathering channel-wise statistics and making 1D convolutional gates along each axis. This helps keep fine structures without needing a lot of computing power at full resolution. Because of this, the MCACoder effectively holds onto and highlights details of thin lines and small objects, which often get lost when processed at lower resolutions. This makes depth estimation more exact and stable, even in areas where older models might be unclear.

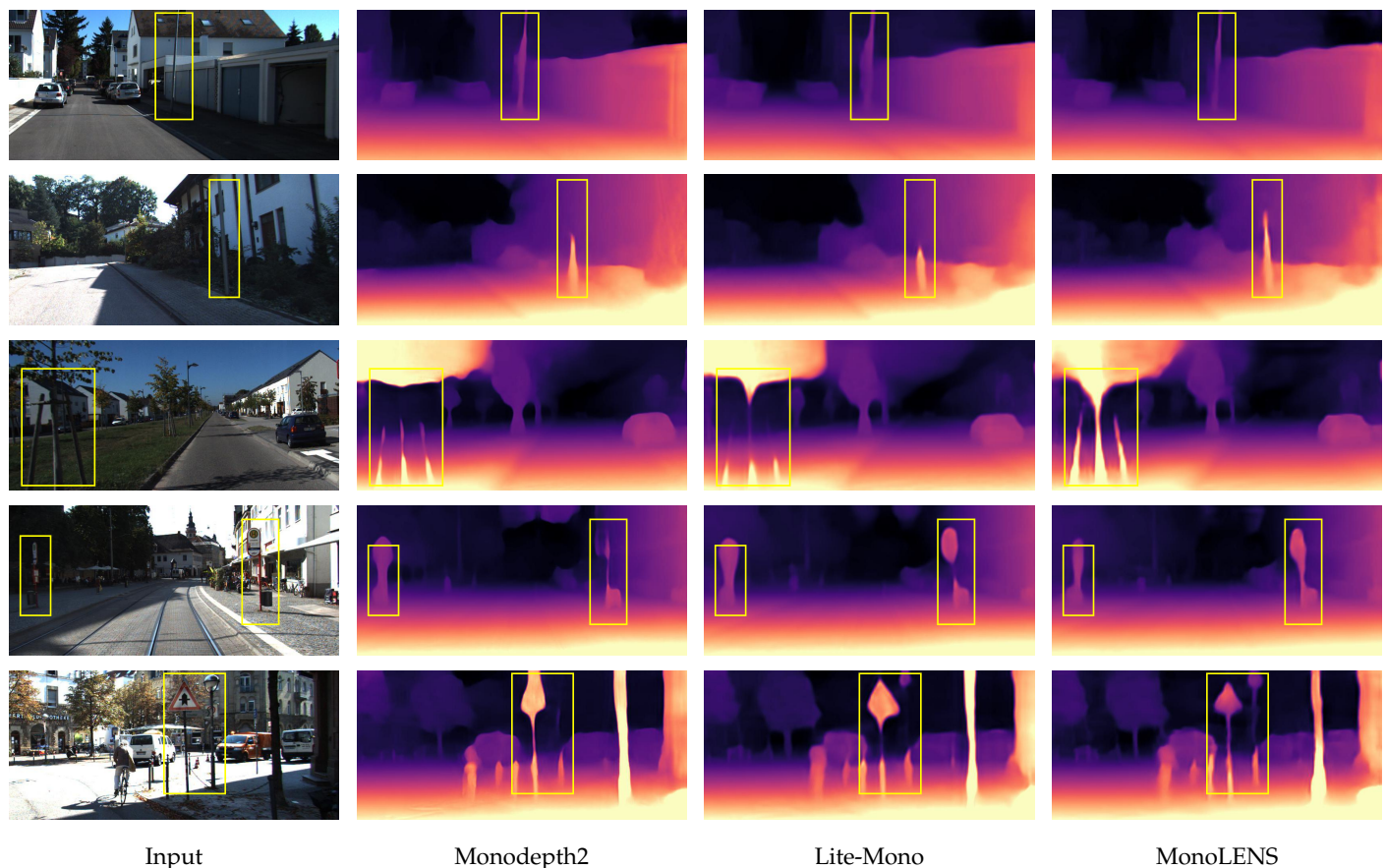


Figure 5. Qualitative results on KITTI dataset. Qualitative results refer to a visual evaluation that compares the clarity of boundaries, the consistency of surfaces, and the presence or absence of artifact. Here are some depth maps generated by Monodepth2 [7], Lite-Mono [9], and MonoLENS (ours).

4.4. Cityscapes Results

Table 2 compares the proposed framework with representative methods on the Cityscapes dataset [14]. “M” denotes training on monocular sequences only, and “M + Se” denotes training with additional semantic information. For Lite-Mono [9] and MonoLENS, we report results obtained from our own training and evaluation environment; for all other methods, we use values reported in their original papers. Compared with Lite-Mono, MonoLENS is slightly worse on Abs Rel and Sq Rel but matches or surpasses it on the remaining metrics. In particular, RMSE improves by about 1.3%. Against the broader set of methods, MonoLENS ranks second on Abs Rel and first on RMSE, RMSE log, and $\delta_1, \delta_2, \delta_3$ (including ties). These results indicate that even without semantics (M), MonoLENS achieves strong accuracy and remains competitive with methods that use semantic supervision (M+Se).

Table 2. Comparison of evaluation metrics for each method on the Cityscapes dataset [14]. The best value for each metric is shown in **bold**, and the second best is underlined. “M”: trained on Cityscapes monocular sequences; “M+Se”: trained on monocular sequences with semantic segmentation.

Method	Data	Depth Error (Lower Is Better)				Depth Accuracy (Higher Is Better)		
		Abs Rel	Sq Rel	RMSE	RMSE Log	δ_1	δ_2	δ_3
Struct2Depth 2 [58]	M + Se	0.145	1.737	7.280	0.205	0.813	0.942	0.976
Pilzer et al. [59]	M	0.240	4.264	8.049	0.334	0.710	0.871	0.937
Monodepth2 [7]	M	0.129	1.569	6.876	<u>0.187</u>	0.849	0.957	<u>0.983</u>
Videos in the Wild [60]	M + Se	0.127	1.330	6.960	0.195	0.830	0.947	0.981
Lite-Mono [9]	M	0.121	<u>1.475</u>	<u>6.732</u>	0.181	<u>0.866</u>	<u>0.961</u>	0.985
MonoLENS	M	<u>0.123</u>	1.506	6.642	0.181	0.870	0.962	0.985

4.5. Complexity and Speed Evaluation

We evaluated the FLOPs, FPS, and inference time of our proposed MonoLENS, comparing its performance against R-MSFM [37], Lite-Mono-small [9], and Lite-Mono [9]. FLOPs were measured using an NVIDIA GeForce RTX 4090, while FPS and inference time were assessed on the NVIDIA Jetson Orin Nano. As presented in Table 3, our method demonstrates an excellent balance between model size and inference speed. MonoLENS reduces total FLOPs by approximately 20.3%. In particular, the decoder FLOPs drop by 62.9%. It achieves the highest throughput and delivers about 4.7% lower latency at batch size 2 than Lite-Mono [9].

This fast inference performance and efficiency mainly come from the introduction of our proposed DS-Upsampling Block. This block is designed to dramatically reduce computational cost while maintaining the expressive power of traditional back-to-back 3×3 convolutions. Specifically, adopting depthwise separable convolution enables equivalent information processing with fewer operations, thereby significantly reducing the model’s overall FLOPs and runtime. As a result, the decoder achieves a notable reduction in runtime while retaining high-frequency details, making it ideally suited for real-time inference. This efficient design directly contributes to MonoLENS’s excellent balance and performance.

Table 3. A comparison of FLOPs, FPS, and inference time on the KITTI dataset. The best value for each metric is shown in **bold**, and the second best is underlined.

Method	FLOPs [G]			FPS	Inference Time [ms]	
	Total	Encoder	Decoder		Batch 2	Batch 6
R-MSFM3 [37]	16.468	2.449	14.020	36.3	43.1	127.7
Lite-Mono-small [9]	<u>4.746</u>	4.028	0.718	<u>37.6</u>	<u>42.7</u>	<u>127.2</u>
Lite-Mono [9]	5.032	4.314	0.718	36.5	44.7	134.2
MonoLENS (ours)	4.008	3.743	0.266	38.8	42.6	124.1

4.6. Ablation Study

Table 4 presents the results of our ablation study, systematically evaluating the contribution of each proposed module to the overall performance of the MonoLENS framework. As discussed in Section 3.4, replacing the decoder’s standard convolutions with depthwise separable convolutions in the DS-Upsampling Block yields a 9.4% reduction in FLOPs and a 3.4% reduction in model parameters while maintaining comparable accuracy. Conversely, omitting depthwise separable convolutions and using only standard convolutions increases parameters and computational cost relative to the depthwise separable variant. Next, incorporating only the MCACoder significantly improves performance, with the Sq Rel

decreasing by approximately 1.5% and the RMSE lowering by approximately 0.4%. This demonstrates the effectiveness of our attention mechanism in enhancing depth prediction quality. Finally, reducing the encoder’s fourth stage from ten to four blocks results in substantial model compression, cutting model parameters by 38.9% and FLOPs by 11.3%. This validates our strategy for efficient encoder design without significant performance degradation. These results emphasize the effectiveness of our design choices in achieving high-performance monocular depth estimation in a lightweight framework.

Table 4. Ablation study on model components. Evaluated on KITTI Eigen split [48] with input size 640×192 .

Depthwise Seprable	MCA Coder	Reduced Encoder Blocks	Depth Error (Lower Is Better)				Depth Accuracy (Higher Is Better)			FLOPs (G)	Params (M)
			Abs Rel	Sq Rel	RMSE	RMSE Log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$		
			0.109	0.872	4.712	0.187	0.885	0.961	0.982	5.032	3.069
✓			0.110	0.868	4.722	0.186	0.885	0.961	0.982	4.559	2.965
	✓		0.110	0.859	4.694	0.187	0.883	0.961	0.982	5.033	3.069
		✓	0.110	0.821	4.664	0.186	0.881	0.961	0.982	4.461	1.875
✓	✓		0.109	0.860	4.700	0.185	0.887	0.962	0.982	4.579	2.970
✓		✓	0.111	0.854	4.699	0.188	0.880	0.960	0.982	3.988	1.772
	✓	✓	0.110	0.835	4.699	0.187	0.883	0.961	0.982	4.462	1.875
✓	✓	✓	0.110	0.833	4.644	0.185	0.883	0.962	0.982	4.008	1.777

4.6.1. Ablation Study on the DS-Upsampling Block

We evaluated how the position of depthwise separable convolution in the DS-Upsampling Block affects depth error, FLOPs, and model parameters. We tested three settings. Before the upsampling layer. After the upsampling layer. On both sides of the upsampling layer.

As shown in Table 5, placing the operator after the upsampling layer gives the best balance of error, FLOPs, and model parameters. A Conv Block first processes features. The network then upsamples by bilinear interpolation. The skip connection is concatenated. A depthwise convolution refines features at full resolution. A pointwise convolution mixes channels. This improves boundaries and thin structures and keeps gradients strong near the prediction head. Compared with the baseline, FLOPs drop by about 9.4% and model parameters drop by about 3.4% while accuracy is maintained or improved. In contrast, placing the operator before upsampling applies it at low resolution and then uses bilinear interpolation, which is not learnable. After upsampling, new high-frequency details from the skip connection are not optimized together with a learnable spatial operator at full resolution, so boundaries become weaker and the error increases. Using the operator on both sides inserts two depthwise separable units with BatchNorm and ReLU around a non-learnable interpolation step. This encourages over-smoothing and makes optimization harder because the gradient path becomes longer and statistics become less stable. These results show that applying depthwise separable convolution after upsampling is effective for efficient feature extraction and information retention.

Table 5. Ablation study on the DS-Upsampling Block. We evaluate the effect of inserting depthwise separable convolution before and/or after the upsampling layer on the KITTI Eigen split [48] with input size 640×192 .

Depthwise Separable (Before Upsampling)	Depthwise Separable (After Upsampling)	Depth Error (Lower Is Better)				FLOPs (G)	Params (M)
		Abs Rel	Sq Rel	RMSE	RMSE Log		
		0.109	0.872	4.712	0.187	5.032	3.069
✓		0.112	0.893	4.748	0.187	4.910	2.978
	✓	0.110	0.868	4.722	0.186	4.559	2.965
✓	✓	0.111	0.885	4.770	0.187	4.437	2.875

4.6.2. Ablation Study on the Number of Blocks in the Encoder

We conducted an ablation study to determine the optimal number of blocks for our proposed method. We varied the number of blocks in the fourth stage of the encoder and evaluated the model's performance. As presented in Table 6, the results reveal that progressively reducing the number of blocks in the fourth stage of the encoder from ten to four allows for substantial reductions in FLOPs and model parameters while largely maintaining depth estimation accuracy. For instance, by decreasing the number of blocks from ten to four, we successfully reduced FLOPs by approximately 11.4% and model parameters by approximately 38.9%. Considering the optimal trade-off between accuracy and computational efficiency, we adopted four blocks for the fourth stage of the encoder in our proposed method. Specifically, reducing the number of blocks from ten to four keeps the changes in Abs Rel and RMSE within about 1%, with RMSE log effectively unchanged. Meanwhile, Sq Rel improves by about 5.9%. When further reducing to three blocks, Abs Rel and RMSE log begin to degrade by around 1%, so we adopt the four blocks in this work.

Table 6. Ablation study on the number of blocks in the fourth stage of the encoder. We evaluated on the KITTI Eigen split [48] with input size 640×192 .

Blocks in the Fourth Stage	Depth Error (Lower Is Better)				FLOPs (G)	Params (M)
	Abs Rel	Sq Rel	RMSE	RMSE Log		
10	0.109	0.872	4.712	0.187	5.032	3.069
7	0.112	0.896	4.797	0.189	4.746	2.472
6	0.110	0.846	4.693	0.186	4.651	2.273
5	0.109	0.838	4.658	0.185	4.556	2.074
4	0.110	0.821	4.664	0.186	4.461	1.875
3	0.112	0.841	4.697	0.188	4.366	1.676

5. Conclusions

This paper proposes MonoLENS, a novel architecture for lightweight and accurate self-supervised monocular depth estimation. MonoLENS employs three key approaches, including the strategic integration of depthwise separable convolutions for efficient feature extraction and information retention, the MCACoder for effectively capturing multi-scale contextual information, and an optimized number of blocks in the encoder that balances computational efficiency with accuracy.

Extensive experiments on the KITTI dataset and the Cityscapes dataset demonstrated that our proposed method achieves an excellent balance between model size and depth estimation accuracy compared with leading existing lightweight self-supervised monocular depth estimation models. Notably, MonoLENS achieved comparable or superior accuracy to much larger models despite its extremely compact size, showcasing high robustness, particularly in estimating the depth of thin structures and small objects. Furthermore, evaluations on the NVIDIA Jetson Orin Nano confirmed its fast inference speed despite significantly reduced FLOPs and model parameters, demonstrating its high suitability for real-time inference on resource-constrained edge devices.

However, several technical challenges remain. In particular, the depth estimation accuracy of objects or backgrounds that are overexposed due to strong light sources, such as direct sunlight, tends to significantly degrade. In regions where strong light sources like sunlight directly hit and cause the image to be overexposed, the ambiguity of depth information increases, making it difficult for the model to accurately estimate depth in some cases. Improving robustness under such specific conditions is an important challenge for future research. However, the MonoLENS proposed in this study presents a new direction for monocular depth estimation that combines efficiency with high accuracy, promising

significant contributions to various real-world applications, including autonomous driving and robotics.

Author Contributions: Conceptualization, G.H.; Funding acquisition, X.K. and H.T.; Investigation, G.H.; Methodology, G.H.; Software, G.H.; Supervision, T.S., X.K., H.Y. and H.T.; Validation, G.H.; Writing—original draft, G.H.; Writing—review and editing, T.S., X.K., H.Y. and H.T. All authors have read and agreed to the published version of the manuscript.

Funding: This work is partly supported by a research grant provided by the Suzuki Foundation.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to its integration into a larger proprietary software framework that is currently under development and subject to confidentiality agreements with our collaborators.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Vu, H.H.; Labatut, P.; Pons, J.P.; Keriven, R. High Accuracy and Visibility-Consistent Dense Multiview Stereo. *IEEE Trans. Pattern Anal. Mach. Intell.* **2011**, *34*, 889–901. [\[CrossRef\]](#) [\[PubMed\]](#)
2. Li, Y.; Ma, L.; Zhong, Z.; Liu, F.; Chapman, M.A.; Cao, D.; Li, J. Deep Learning for Lidar Point Clouds in Autonomous Driving: A Review. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 3412–3432. [\[CrossRef\]](#) [\[PubMed\]](#)
3. Liu, Y.; Liu, H.; Li, Y.; Zhao, S.; Yang, Y. Building BIM Modeling Based on Multi-Source Laser Point Cloud Fusion. *J. Geogr. Inf. Sci.* **2021**, *23*, 763–772.
4. Eigen, D.; Puhersch, C.; Fergus, R. Depth Map Prediction from a Single Image Using a Multi-Scale Deep Network. *Adv. Neural Inf. Process. Syst.* **2014**, *27*, 2366–2374.
5. Fanello, S.R.; Keskin, C.; Izadi, S.; Kohli, P.; Kim, D.; Sweeney, D.; Criminisi, A.; Shotton, J.; Kang, S.B.; Paek, T. Learning to Be a Depth Camera for Close-Range Human Capture and Interaction. *ACM Trans. Graph.* **2014**, *33*, 1–11. [\[CrossRef\]](#)
6. Wei, Y.; Liu, S.; Rao, Y.; Zhao, W.; Lu, J.; Zhou, J. NerfingMVS: Guided Optimization of Neural Radiance Fields for Indoor Multi-View Stereo. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 5610–5619.
7. Godard, C.; Mac Aodha, O.; Firman, M.; Brostow, G.J. Digging into Self-Supervised Monocular Depth Estimation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 3828–3838.
8. Liu, S.; Yang, L.T.; Tu, X.; Li, R.; Xu, C. Lightweight Monocular Depth Estimation on Edge Devices. *IEEE Internet Things J.* **2022**, *9*, 16168–16180. [\[CrossRef\]](#)
9. Zhang, N.; Nex, F.; Vosselman, G.; Kerle, N. 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, Vancouver, BC, Canada, 17–24 June 2023; pp. 18537–18546.
10. Higashiuchi, G.; Shimada, T.; Kong, X.; Tomiyama, H. Efficient Monocular Depth Estimation Using Depthwise Separable Convolutions and Multidimensional Cooperative Attention (in Japanese). In Proceedings of the Forum on Information Technology, Information Processing Society of Japan, Hokkaido, Japan, 3–5 September 2025.
11. Chollet, F. Xception: Deep Learning with Depthwise Separable Convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1251–1258.
12. Yu, Y.; Zhang, Y.; Cheng, Z.; Song, Z.; Tang, C. MCA: Multidimensional Collaborative Attention in Deep Convolutional Neural Networks for Image Recognition. *Eng. Appl. Artif. Intell.* **2023**, *126*, 107079. [\[CrossRef\]](#)
13. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision Meets Robotics: The KITTI Dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [\[CrossRef\]](#)
14. Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; Schiele, B. The Cityscapes Dataset for Semantic Urban Scene Understanding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 3213–3223.
15. Johnston, A.; Carneiro, G. Self-Supervised Monocular Trained Depth Estimation Using Self-Attention and Discrete Disparity Volume. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 4756–4765.

16. Laina, I.; Rupprecht, C.; Belagiannis, V.; Tombari, F.; Navab, N. Deeper Depth Prediction with Fully Convolutional Residual Networks. In Proceedings of the International Conference on 3D Vision, Stanford, CA, USA, 25–28 October 2016; pp. 239–248.
17. Fu, H.; Gong, M.; Wang, C.; Batmanghelich, K.; Tao, D. Deep Ordinal Regression Network for Monocular Depth Estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2002–2011.
18. Lee, J.H.; Han, M.K.; Ko, D.W.; Suh, I.H. From Big to Small: Multi-Scale Local Planar Guidance for Monocular Depth Estimation. *arXiv* **2019**, arXiv:1907.10326.
19. Bauer, Z.; Li, Z.; Orts-Escolano, S.; Cazorla, M.; Pollefeys, M.; Oswald, M.R. NVS-MonoDepth: Improving Monocular Depth Prediction with Novel View Synthesis. In Proceedings of the International Conference on 3D Vision, London, UK, 1–3 December 2021; pp. 848–858.
20. Bhat, S.F.; Alhashim, I.; Wonka, P. AdaBins: Depth Estimation Using Adaptive Bins. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 4009–4018.
21. Godard, C.; Mac Aodha, O.; Brostow, G.J. Unsupervised Monocular Depth Estimation with Left-Right Consistency. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 270–279.
22. Garg, R.; Bg, V.K.; Carneiro, G.; Reid, I. Unsupervised CNN for Single View Depth Estimation: Geometry to the Rescue. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; pp. 740–756.
23. Sun, J.; Zheng, N.N.; Shum, H.Y. Stereo Matching Using Belief Propagation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2003**, *25*, 787–800. [[CrossRef](#)]
24. Poggi, M.; Tosi, F.; Mattoccia, S. Learning Monocular Depth Estimation with Unsupervised Trinocular Assumptions. In Proceedings of the International Conference on 3D Vision, Verona, Italy, 5–8 September 2018; pp. 324–333.
25. Watson, J.; Firman, M.; Brostow, G.J.; Turmukhambetov, D. Self-Supervised Monocular Depth Hints. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 2162–2171.
26. GonzalezBello, J.L.; Kim, M. Forget about the Lidar: Self-Supervised Depth Estimators with Med Probability Volumes. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 12626–12637.
27. Zhu, S.; Brazil, G.; Liu, X. The Edge of Depth: Explicit Constraints between Segmentation and Depth. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 13116–13125.
28. Peng, R.; Wang, R.; Lai, Y.; Tang, L.; Cai, Y. Excavating the Potential Capacity of Self-Supervised Monocular Depth Estimation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 15560–15569.
29. Zhou, T.; Brown, M.; Snavely, N.; Lowe, D.G. Unsupervised Learning of Depth and Ego-Motion from Video. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1851–1858.
30. Shu, C.; Yu, K.; Duan, Z.; Yang, K. Feature-Metric Loss for Self-Supervised Learning of Depth and Egomotion. In Proceedings of the European Conference on Computer Vision, Virtual, 23–28 August 2020; pp. 572–588.
31. Lyu, X.; Liu, L.; Wang, M.; Kong, X.; Liu, L.; Liu, Y.; Chen, X.; Yuan, Y. HR-Depth: High Resolution Self-Supervised Monocular Depth Estimation. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 19–21 May 2021; pp. 2294–2301.
32. Yin, W.; Liu, Y.; Shen, C.; Yan, Y. Enforcing Geometric Constraints of Virtual Normal for Depth Prediction. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 5684–5693.
33. Wofk, D.; Ma, F.; Yang, T.J.; Karaman, S.; Sze, V. FastDepth: Fast Monocular Depth Estimation on Embedded Systems. In Proceedings of the International Conference on Robotics and Automation, Montreal, QC, Canada, 20–24 May 2019; pp. 6101–6108.
34. Nekrasov, V.; Dharmasiri, T.; Spek, A.; Drummond, T.; Shen, C.; Reid, I. Real-Time Joint Semantic Segmentation and Depth Estimation Using Symmetric Annotations. In Proceedings of the International Conference on Robotics and Automation, Montreal, QC, Canada, 20–24 May 2019; pp. 7101–7107.
35. Hu, J.; Fan, C.; Jiang, H.; Guo, X.; Gao, Y.; Lu, X.; Lam, T.L. Boosting Lightweight Depth Estimation via Knowledge Distillation. In Proceedings of the International Conference on Knowledge Science, Engineering and Management, Guangzhou, China, 16–18 August 2023; pp. 27–39.
36. Sheng, F.; Xue, F.; Chang, Y.; Liang, W.; Ming, A. Monocular Depth Distribution Alignment with Low Computation. In Proceedings of the International Conference on Robotics and Automation, Philadelphia, PA, USA, 23–27 May 2022; pp. 6548–6555.
37. Zhou, Z.; Fan, X.; Shi, P.; Xin, Y. R-MSFM: Recurrent Multi-Scale Feature Modulation for Monocular Depth Estimating. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 12777–12786.
38. Zhang, Z.; Xu, C.; Yang, J.; Gao, J.; Cui, Z. Progressive Hard-Mining Network for Monocular Depth Estimation. *IEEE Trans. Image Process.* **2018**, *27*, 3691–3702. [[CrossRef](#)] [[PubMed](#)]
39. Hoang, V.T.; Jo, K.H. PyDNet: An Efficient CNN Architecture with Pyramid Depthwise Convolution Kernels. In Proceedings of the International Conference on System Science and Engineering, Dong Hoi, Vietnam, 20–21 July 2019; pp. 154–158.

40. Varma, A.; Chawla, H.; Zonooz, B.; Arani, E. Transformers in Self-Supervised Monocular Depth Estimation with Unknown Camera Intrinsics. *arXiv* **2022**, arXiv:2202.03131.
41. Zhao, C.; Zhang, Y.; Poggi, M.; Tosi, F.; Guo, X.; Zhu, Z.; Huang, G.; Tang, Y.; Mattoccia, S. MonoViT: Self-Supervised Monocular Depth Estimation with a Vision Transformer. In Proceedings of the International Conference on 3D Vision, Prague, Czech Republic, 12–15 September 2022; pp. 668–678.
42. Lee, Y.; Kim, J.; Willette, J.; Hwang, S.J. MPViT: Multi-Path Vision Transformer for Dense Prediction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–20 June 2022; pp. 7287–7296.
43. Zhang, G.; Tang, X.; Wang, L.; Cui, H.; Fei, T.; Tang, H.; Jiang, S. RepMono: A Lightweight Self-Supervised Monocular Depth Estimation Architecture for High-Speed Inference. *Complex Intell. Syst.* **2024**, *10*, 7927–7941. [[CrossRef](#)]
44. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861.
45. Zhang, Z.; Wang, Y.; Huang, Z.; Luo, G.; Yu, G.; Fu, B. A Simple Baseline for Fast and Accurate Depth Estimation on Mobile Devices. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 2466–2471.
46. Liu, Z.; Wang, Q. Edge-Enhanced Dual-Stream Perception Network for Monocular Depth Estimation. *Electronics* **2024**, *13*, 1652. [[CrossRef](#)]
47. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. CBAM: Convolutional Block Attention Module. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 3–19.
48. Eigen, D.; Fergus, R. Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 2650–2658.
49. Yang, Z.; Wang, P.; Wang, Y.; Xu, W.; Nevatia, R. LEGO: Learning Edge with Geometry All at Once by Watching Videos. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 225–234.
50. Yin, Z.; Shi, J. GeoNet: Unsupervised Learning of Dense Depth, Optical Flow and Camera Pose. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 1983–1992.
51. Watson, J.; Mac Aodha, O.; Prisacariu, V.; Brostow, G.; Firman, M. The Temporal Opportunist: Self-Supervised Multi-Frame Monocular Depth. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 1164–1174.
52. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. ImageNet: A Large-Scale Hierarchical Image Database. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
53. Loshchilov, I.; Hutter, F. Decoupled Weight Decay Regularization. *arXiv* **2017**, arXiv:1711.05101.
54. Wang, C.; Buenaposada, J.M.; Zhu, R.; Lucey, S. Learning Depth from Monocular Videos Using Direct Methods. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2022–2030.
55. Luo, C.; Yang, Z.; Wang, P.; Wang, Y.; Xu, W.; Nevatia, R.; Yuille, A. Every Pixel Counts++: Joint Learning of Geometry and Motion with 3D Holistic Understanding. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *42*, 2624–2641. [[CrossRef](#)] [[PubMed](#)]
56. Casser, V.; Pirk, S.; Mahjourian, R.; Angelova, A. Depth Prediction without the Sensors: Leveraging Structure for Unsupervised Learning from Monocular Videos. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, Hawaii, USA, 27–28 January 2019; Volume 33, pp. 8001–8008.
57. Klingner, M.; Termöhlen, J.A.; Mikolajczyk, J.; Fingscheidt, T. Self-Supervised Monocular Depth Estimation: Solving the Dynamic Object Problem by Semantic Guidance. In Proceedings of the European Conference on Computer Vision, Virtual, 23–28 August 2020; pp. 582–600.
58. Casser, V.; Pirk, S.; Mahjourian, R.; Angelova, A. Unsupervised Monocular Depth and Ego-Motion Learning with Structure and Semantics. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Long Beach, CA, USA, 15–20 June 2019.
59. Pilzer, A.; Xu, D.; Puscas, M.; Ricci, E.; Sebe, N. Unsupervised Adversarial Depth Estimation Using Cycled Generative Networks. In Proceedings of the International Conference on 3D Vision, Verona, Italy, 5–8 September 2018; pp. 587–595.
60. Gordon, A.; Li, H.; Jonschkowski, R.; Angelova, A. Depth from Videos in the Wild: Unsupervised Monocular Depth Learning from Unknown Cameras. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 8977–8986.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.