

MC-Depth: Modular and Compute-Efficient Monocular Depth Estimation for Outdoor On-Board Vehicle Perception Systems

Marie-Claire Iatrides[1,2(✉)], Petra Gomez-Krämer[1], Olfa Ben Ahmed[3], and Sylvain Marchand[1]

[1] L3i Laboratory, La Rochelle University, La Rochelle, France
{marie-claire.iatrides,petra.gomez,sylvain.marchand}@univ-lr.fr
[2] Association Ferrocampus, Saintes, France
[3] Xlim Institute of Research, Poitiers University, Poitiers, France
olfa.ben.ahmed@univ-poitiers.fr

Abstract. The transportation sector is still expanding into autonomous vehicles and requires them to perceive their environment with growing precision through sensors. In this paper, we propose a modular framework for monocular depth estimation, built by pairing off-the-shelf encoders with lightweight decoders to support resource-efficient deployment in resource-constrained environments. Our focus is outdoor, vehicle-centric, on-board perception, targeting platforms such as cars, trains, and drones. Specifically, we introduce MCDec-dw, a new lightweight decoder that combines depthwise separable convolutions with efficient channel attention. In addition, we consider five encoder backbones that can be paired with three depthwise-compatible decoders, enabling systematic exploration of accuracyâĂŞefficiency trade-offs. For each of these parts, we also explored scaling techniques independently in order to reduce the impact of our final models on an on-board system to achieve deployment in a resource-constrained environment. Our work was tested on KITTI with above state-of-the-art results for real-world implementation. With our developed segments, our method achieves performances up to 92.2% in δ_1 threshold accuracy with 0.312 absolute relative error for as little as 3.96 M parameters with an inference speed of 94 FPS. We also propose model combinations with a model size of 1.62 M parameters with a very low decrease in performance and up to 76 FPS.

Keywords: Lightweight CNN · Resource-Efficient ML · Monocular Depth Estimation · Autonomous Vehicles · Efficient Channel Attention

1 Introduction

In the transport industry, many advertise the dawn of new autonomous systems to revolutionize our experience. But to achieve the autonomy required for such

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027
M. De Marsico et al. (Eds.): ICPR 2026, LNCS 16822, pp. 80–95, 2027.
https://doi.org/10.1007/978-3-032-31452-9_6

a system, industry stakeholders must develop not only safe or accurate systems, but also insert technological innovations developed by researchers into the existing constraint bound systems we have today. With the example of vehicles such as trains or cars, the artificial intelligence aided modules must not only answer to the tasks a conductor or driver must perform, but also, follow the integration constraints on-board the vehicle without restricting the existing systems. One of the major concerns of autonomous vehicles is the perception of their environment. In order to appreciate what is around them, such systems can benefit from depth estimation. In this task, the system estimates relative or metric distances in the vehicle's environment using acquired camera data in real-time.

Deep Learning is not new to image analysis tasks, whether it is for classification, segmentation or detection. Depth estimation is no exception and numerous techniques have been introduced to solve this task. However, with recent technologies, state-of-the-art solutions rely on high computational costs with ever growing architectures and complex models. A high number of trainable parameters makes these solutions poorly suited to real-time on-board applications [19].

In this paper, we propose a resource-efficient approach to monocular depth estimation (MC-Depth) for outdoor vehicular perception, designed for deployment on resource-constrained on-board systems. We develop an on-board system that uses low resolution images as the only input to assess and analyze a vehicle's surrounding. Our approach focuses on lightweight architectures to limit energy requirements and optimize on-board memory consumption. In order to satisfy real-world on-board deployment, the models proposed are built with the integration of different scaling techniques and performance oriented mechanisms to satisfy both performance and on-board associated constraints. We apply a novel systematic modular design to our models with an encoder-decoder interchangeable architecture for adaptability to existing systems. With this decoder we surpass state-of-the-art models on the KITTI dataset for MDE whichever encoder used with real-time applicable inference. We designed our model as a simple pipeline architecture not reliant on parallel guiding networks and with a self-reliant feature decoder that proves the ability of lightweight models for MDE tasks. In total, we provide combinations between five encoders and three decoders for a comparative study and compatibility analysis. Our models can be used on both GPU and CPU platforms and only require up to 9GB of RAM to function. Our contributions are threefold:

- We propose a novel lightweight decoder, MCDec-dw, and model, MC-Depth.
- We design a modular model architecture with plugin encoders and decoders.
- And we demonstrate real-world and real-time applicability of MC-Depth on both low power GPU and CPU with above state-of-the-art performance on KITTI.

Our paper is organized as follows. Firstly, we review the architectures and models developed in previous works. Secondly, we describe the limits of our intended project and its possible applications. Subsequently, we introduce our method and our results. And lastly, we offer remarks on our work as well as an analysis of its real-world applicability.

2 Related Works

In this section, we briefly peruse similar research in the field. Furthermore, we review different scaling techniques applied to this task. To continue our review, we explored different techniques introduced to either implement or guide depth estimation networks.

2.1 Depth Estimation

In the early precepts of depth estimation, in 2005, a Laplacian probabilistic model [23] was used, only providing rough and highly pixelized predictions. Later, two main approaches were explored, monocular or stereo prediction. In the case of stereo images, the models exploit the geometrical information provided through the viewpoint shift of both cameras. While efficient, it requires a higher computational overhead which is not considered "frugal". In contrast, monocular tasks require a single viewpoint but must extract more context and features in order to infer geometrical information [18,20]. Recent rise in computational ability has allowed for bigger models that can obtain high performances at the cost of integrability in smaller systems like with Depth Anything [31,32](24.8M parameters). Another more resource-conscious approach strives to achieve equivalent performances with significantly smaller models such as FastDepth [29] (3.93M parameters) with comparable performances on indoor settings only which we want to extend to outdoor scenes. In order to balance performance and resource management, we have chosen to limit our approach to monocular depth estimation but also implement scaling mechanisms to limit model size.

2.2 Scaling of Resources

Resource conscious networks allow models to run on smaller devices or simply as a way to bypass complex computations hindering the training process [7]. Skip connections were first introduced as a way to fix the vanishing gradient problem in large architectures but soon proved to provide contextual information useful for prediction reconstruction. It was demonstrated with UNets [21], later with ResNets [8] and then further explored with densely convolutional blocks in DenseNets [11]. Similarly, depthwise separable convolutions [2] were introduced to reduce model computational needs but causes the gradient calculation to take into account lesser context thus impacting performances. Another popular method relies on quantization which converts model weights into smaller formats but that also causes loss in precision as well [33]. In our approach, we integrate the benefits of skip connections by proposing efficient classifiers as encoder backbones and implementing depthwise convolutions in our architecture. To counter the drop in performances we turn towards attention mechanisms.

2.3 Attention-Based Methods

The main types of attention are channel and spatial attention. SE-Block [10] uses fully connected layers to model the inter-dependency between channels but big

group convolutions increase memory consumption and reduce efficiency [16]. As for spatial attention, Gather-Excite Network (GENet [9]) and Point-wise Spatial Attention Network (PSANet [36]) networks both propose high performing attention modules with a high computational burden. As a solution, the ECA-Block [28] uses a lightweight efficient channel attention mechanism reduced to a single dimension vector. It significantly improves performance for classification tasks, in our work we implement this idea for lightweight architectures in our MDE. Lastly, hybrid methods can not only include channel and spatial attention but also adaptive feature refinement with for example the convolutional Block Attention module (CBAM) [30]. These methods, using spatial attention, do not allow for diminished resolution computation as it requires feature map sized outputs and therefore carry high computational overhead.

Previous works introduce spatial attention modules as the center approach to solving deep learning tasks with the Transformer [26]. The introduction of the transformer architecture [26] for language tasks quickly proved to be transferable to vision tasks with impressive performances [5]. As such, many approaches to depth estimation quickly followed with the introduction of multi-scale fusion attention modules [1,14], deformable cross-attention modules and adaptive bins method (SimMDE model) [15] but also the isolation of transformer modules implemented into a CNN-based architecture with, for example, PDNet [14]. All these models, while having high results carry the burden of a quadratic complexity inherent to Transformer-based attention not acceptable for lightweight approaches. Surprisingly, TinyDepth [1] and another UNet-inspired lightweight model [17] do achieve acceptable performance with 6.2M and 8M parameters respectively using transformer-inspired modules. In contrast, PDNet counts 76.99M parameters and SimMDE with 30.9M parameters [15] for comparable results. Our work aims for a lightweight implementation of our models and we have therefore chosen to limit ourselves to CNN-based architectures in order to limit resource requirements and ensure lower complexity of the model even with high dimension input.

2.4 Guided Approaches

Many recent works implement a parallel Pose-Net to infer geometrical information from a monocular point of view [13,24,25,34,35]. These methods introduce modules ranging from multi-range feature extraction (Lite-Mono [34] with 3.1M parameters or RTIA-Mono [35] with 1.38M parameters), atrous convolution residual modules [24] (3.0M parameters), feature adaptive fusion module with boundary refinement (BoRe-Depth [13] with 8.7M parameters) or an additional pretrained semantic segmentation parallel model (MG-Mono [27] with 1.2M parameters). The works present a deceptive amount of trainable parameters as they do not encompass the PoseNet and prove to require higher computational costs during training than anticipated. Another method relies on edge guided networks such as EGD-Net [4] (2.21M parameters) Sobel operator feature extraction and RGB filtering with cross-model fusion of features with MsGf [25].

These lightweight models have an intricate fixed architecture not easily made modular and thus not easy to integrate to more complex architectures.

In our work we aim to create a single pipeline model that estimates depth with monocular images without introducing the complexity of transformer-based architectures or requiring parallel trained models. In contrast to previous works, our model uses a supervised training method. We also use a channel-wise attention mechanism as a way to counter performance drop due to smaller resolutions usage during training and work to compete with other light-weight methods while proposing a highly modular architecture easily integrated into other systems.

3 Perimeter of Study

For scene perception tasks, we are confronted with a lack of open-source data, driven mainly by the difficulty of providing clean and annotated data. On the other hand, within automobile trends, many innovations have been introduced in order to develop the first autonomous driving vehicles. These systems have then also been introduced to other sectors like the railway industry or the development of smaller on-board systems like drones. In the last few years, there have been many outdoor datasets introduced for tasks ranging from obstacle detection to semantic segmentation for automobile applications. We describe in the following the task we address in this article as well as the dataset we used.

Our work is aimed to be implemented as part of a regional project for cleaner transportation on rails but our lack of data in the field has led us to study car oriented datasets that would still fulfill our environment analysis objectives. For a system to perceive its surroundings, the first step is the introduction of sensors that allow the system to appreciate its environment. In an frugal optic, it is interesting to limit a system to less sensors. For instance, to perceive its environment, the installation could be reduced to a single camera. Using this setup, the system could perform analysis through monocular depth estimation. In this task, the system is required to establish relative or exact distances between the sensor and the objects of the scene. In the case of vehicles, the system needs to be able to perform in an outdoor setting. As such, these requirements are used as guides to choose amongst the available open-source datasets.

In accordance with our objectives, we have focused on a monocular outdoor scenes dataset: KITTI [3]. This dataset contains both images of urban and sub-urban areas from the viewpoint of a car with their corresponding depth maps. It is composed of 50k images taken in and around the city of Karlsruhe with their corresponding sparse depth maps acquired through Lidar point cloud projection. KITTI contains wide resolution images (1382×512 pixels) coupled with point clouds comprised of 100k points per cycle with a 64 vertical resolution. We have considered using other datasets such as Cityscapes but while it contains higher resolution depth maps, it is also hindered by numerous objective visual biases that hinders training for smaller models on reduced resolution images. In order to mimic real-world conditions, we have applied a simple resolution reduction preprocessing on the available data. We reduce our data to 320×256

pixel resolution images without cropping the images as many have been in previous work. While we agree that only 60% of the image is effectively annotated because of the orientation of the Lidar sensors we do believe that is, in fact, closer to real-world conditions and must therefore be taken into account for training. Using these datasets, we wish to train our models for environment appreciation in order to make this knowledge transferable to other circulation settings while answering a need for frugal deep learning approaches on smaller resolution data. As there are not many relevant available datasets, we limit our evaluation to an urban and suburban vehicular environment.

4 Proposed Approach

In this section, we detail the different architectures that we designed to satisfy energy, storage and computation power constraints necessary for on-board systems. This work also aims to develop resource-efficient deep learning architectures using scaling techniques. The task of monocular depth estimation provides a depth map that essentially resumes to a pixel-wise classification using geometries featured in an image while restricting the model to a single sensor view of the environment. As a result, it provides a detailed description of the entities that constitute the environment of the sensor equipped vehicle. While this task is relatively costly in terms of computation, results can provide fine-grained information on the input content thus providing the system with the necessary information to react to what is around it.

Our method is built as a modular framework for encoder-decoder interchangeability. The obtained models share a similar architecture composed of a classifier type model as an encoding backbone, an adapter layer specific to each encoder and a plugin decoder. In our work, we explored different model reduction techniques in order to optimize our architectures to achieve minimal resource requirements ideal for use on-board systems such as cars, trains or other vehicles.

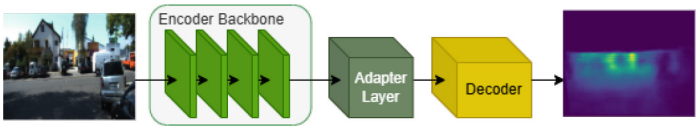


Fig. 1. Model general architecture.

4.1 Encoders

We propose a series of five different encoders based on the architectures of UNet [21], DenseNet-56 and 67 [11], ResNet-18 [8] and MobileNet [22]. MobileNet and ResNet-18 were selected in order to efficiently compare our method with

previous work on resource-efficient monocular depth estimation [29]. Comparatively, DenseNets and UNet were introduced for a better integration of other tasks with our future work in mind. Each of these models was not initially developed for depth estimation but segmentation for UNet and classification for the others. As a result, we modified their architecture to satisfy our needs. In order to make these encoders compatible with our modular plug-and-play decoders as illustrated in Fig. 1, we also added adapters as bottlenecks that ensured compatibility with our pipeline.

For this paper, we have limited our scope to encoders built with a 4 block sequence of down-sampling operations for UNet and DenseNet-56 while DenseNet-67 counts 5 blocks, and also the well known ResNet-18 and MobileNet classifiers with their last layer removed. Because all these models have a different bottleneck channel and feature size as described in Table 1, an adapter block was added for each architecture. The output of this layer is set to be a tensor of 1024 channels with a feature size of 8×10 in order to fit into our decoder pipeline.

Table 1. Model bottlenecks' output **channel** and (feature size) with an input of 320 × 256 and adapter block.

Blocks	UNet	MobileNet	ResNet-18	DenseNet-56/-67	
BottleNeck	**1024** (16×20)	**1024** (8×10)	**512** (8×10)	**288** (8×10)	**448** (8×10)
Adapter block	Conv2x2	Identity	Conv1x1	Conv1x1	Conv1x1

4.2 Decoders

In an encoder-decoder architecture, the decoding branch is meant to merge and upsample the features extracted through the encoder path to produce a prediction map. The main element of the decoding branch is the upsample operation used whether it is a transposed convolution, interpolation or unpooling operation. In our work we explore 3 main decoder blocks: UpConv [12], NNConvdw [29] and our original MCDec-dw, each built with a 5 block repetition followed by a final point-wise convolution. The upsample block comprises of a depthwise separable convolution we call Conv Block detailed in Fig. 2 and interpolation with a scale factor of 2. Our method differs from other works as it does not rely on multi-task guidance or sectional training but instead uses more classical convolution methods and skip connections coupled with an attention mechanism to augment the model's ability to reconstruct information with additional context without sacrificing complexity. We also added a depthwise separable convolution mechanism with MCDec-dw, similarly to NNConvdw, in order to greatly decrease the size of our decoder with only minor impact on performances.

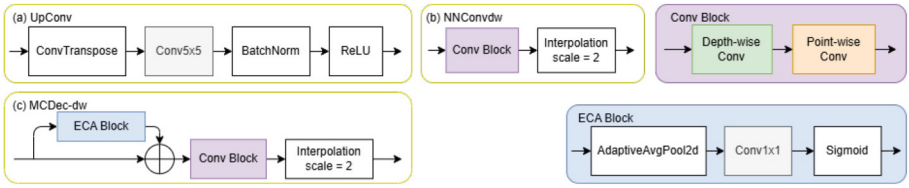


Fig. 2. Decoder block operations: (a) UpConv, (b) NNConvdw and (c) MCDec-dw.

UpConv. The first decoder, UpConv, is a classic decoder block built as a succession of 2×2 unpooling operations followed by a single 5×5 convolution as introduced for fully convolutional residual networks [12] (Fig. 2(a)). This decoder architecture was the first to include deep CNN architectures for depth estimation tasks. This method uses high complexity operations but relies on the fully-connected layers' ability to tackle high-dimensional regression problems.

NNConvdw. The second decoder, NNConvdw, operates a 5×5 convolution followed by a nearest-neighbor interpolation method with a 2 scale factor as implemented in FastDepth [29] (Fig. 2(b)). As mentioned in their paper, the order of the operations also has an impact on computation complexity. In this architecture, the interpolation is performed after the convolution which lowers the resolution, thus reducing the overall cost of the decoder. The suffix 'dw' stands for depthwise separable convolutions that allow for lower complexity of the operation to ensure better inference.

MCDec-dw. Our novel decoder, MCDec-dw, is designed based on a residual connection mechanism fused with a channel attention block and upsample block as described in Fig. 2(c). The attention mechanism is based on the ECA module from ECA-Net [28]. The output is then concatenated with the layer input in order to fuse local cross-channel interaction with the global feature detailed in finer grained features.

There are multiple approaches to guiding models through attention maps. The main mechanism relies on channel-wise or spatial-wise attention. This mechanism was first proposed to mimic the brain's ability to prioritize information through visual cues. Spatial-wise mechanisms require the use of spatial information in previous feature maps or even the input and therefore require the computation of complex operations that inevitably incur higher computational costs which do not align with our objectives. As such, we explored the impact of adding a channel-wise mechanism that have a much lower impact on model size and complexity. As explained in previous work [16] capturing local cross-channel interaction while beneficial to learn channel attention, it requires an important amount of parameters which, in turn, brings higher architectural complexity. Excessive group convolutions directly increase memory costs, therefore decreasing efficiency. As a solution, ECA proposes to capture local cross-channel interaction using a band matrix $\mathbf{W}_k$ with

$$\mathbf{W}_k = \begin{bmatrix} w^{1,1} & \cdots & w^{1,k} & 0 & 0 & \cdots & \cdots & 0 \\ 0 & w^{2,2} & \cdots & w^{2,k+1} & 0 & \cdots & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & w^{C,C-k+1} & \cdots & w^{C,C} \end{bmatrix} \tag{1}$$

This matrix counts $k \times C$ parameters which is considerably less than the $C \times C$ present in the regular Squeeze and Excitation (SE) channel attention mechanism [10]. The ECA method uses group dependency to outperform the SE method as the weight of y_i is calculated by considering its interaction with its k neighbors as described in Eq. 2 with Ω_i^k is the set of k adjacent neighbors.

$$\omega_i = \sigma\left(\sum_{j=1}^{k} w_i^j y_i^j\right), y_i^j \in \Omega_i^k \tag{2}$$

It was then proven that the ECA method allows for equal performances while using shared learning parameters for all channels as described in Eq. 3.

$$\omega_i = \sigma\left(\sum_{j=1}^{k} w^j y_i^j\right), y_i^j \in \Omega_i^k \tag{3}$$

This method can therefore be reduced to a single dimension convolution with a k kernel size with $\omega = \sigma\left(\text{Conv1D}_k(y)\right)$. This operation, constituting the backbone of the ECA module, while ensuring low complexity and, in our case, a low impact on model size, also strengthens the model's ability to grasp and utilize local features during training. This module ensures both efficiency and effectiveness with very low compromise on computational overhead and memory usage.

4.3 Reducing Model Complexity

In order to design a resource-efficient architecture, we explored different scaling techniques. Skip connections [8] were first introduced as a learning mechanism for deep networks in order to mitigate the effect of degrading gradient. This mechanism was later proven to be an efficient method to reduce model size while conserving high level features to inject into the decoding process for better prediction. Taking advantage of this, we develop a new lightweight Convolutional Neural Network (CNN) as a solution towards the use of artificial intelligence (AI) with on-board deployment in mind. In this context, we utilized these skip connections to both reduce the footprint of our models but also strengthen the performance of our smaller architectures by adding context and conserving higher level features throughout the encoder. On the decoding side, residual connections are also introduced in our novel decoder, MCDec, to reinforce global information extraction and strengthen prediction through an efficient cross-channel attention block that only slightly increases the complexity of the decoder while bringing a clear performance gain.

Another mechanism we explored is depthwise separable convolutions. On the one hand, compared to regular convolutions, the succession of operations

detailed in Fig. 3 results in less parameters per convolutional layer. But, on the other hand, it decreases the amount of context passed down throughout the calculations, thus furthering the usefulness of the previously mentioned skip connections. These convolutions can be implemented in our work in both the encoder and decoder independently.

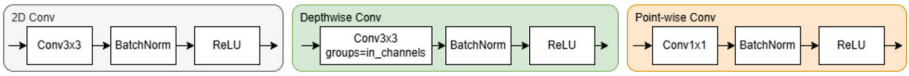


Fig. 3. Depthwise separable convolution versus regular convolution.

5 Experiments and Results

We present in this section the setup used to implement our approach and the evaluation method established.

Five metrics were considered to comprehensively evaluate performance for our depth estimation task, including threshold accuracy (δ_x), absolute relative error ($AbsRel$), mean absolute error (MAE), root mean squared error ($RMSE$), and scale-invariant logarithmic metric ($SiLog$). Furthermore, we selected some metrics to measure the efficiency or frugality level of the models, including floating-point operations ($FLOPs$) and the multiply-accumulate score ($MACs$).

The evaluation metrics selected in this work are those most widely applied and proven to be representative in the task of depth estimation but also the comparison of models with a frugal optic.

5.1 Implementation

For this work, the models are trained from scratch for 100 epochs on our NVIDIA GeForce RTX 2080 Ti GPU with a 3.9GHz processor and Threadripper Pro 3955WX CPUs installed with 256Go of RAM. Our implementation also supports CPU and testing samples were performed on 20 Intel i9-10940X CPUs installed with 60Go of RAM. Parameters include a learning rate initialized at 1e-5 but implemented with a scheduler, a batchsize of 8 for training and 4 for validation. The training was conducted with a mean squared error loss minimized through an Adam optimizer. The aforementioned parameters were selected through both theoretical compatibility to our task and then adjusted via empirical analysis to obtain a satisfying combination of input parameters and evaluation processes.

Each dataset is divided with 60-20-20 proportions for training, validation and testing sets with a random split and a manual seed set at 0 for repetition purposes. As for data preprocessing, we have worked to enforce input efficiency by reducing the resolution of the image from its original 1382×512 to 320×256 pixels. By doing so, we greatly decrease the model's input of information and

limit the amount of context it is able to extract from the input during training. It was implemented to evaluate the model's ability to work with low quality sensors in the context of real-world on-board deployment, more specifically, vehicles.

Our architecture allows for an easily permutable combination of encoder and decoder architectures between the ones that were implemented in our work. As mentioned in our approach, we have implemented 6 different encoders, compatible with 3 decoders, 2 of which were introduced in previous work but never in our specific context or for this task. As mentioned in Table 2, our work also re-implements the model FastDepth [29] by combining a MobileNet backbone with an NNConvdw decoder.

Table 2. Model characteristics with FastDepth [29] as {MobileNet+NNConvdw}

Encoder	Decoder	Inference (FPS)			File (MB)	GFLOPs	GMACs	Parameters
		CPU	GPU					
UNet	UpConv	6.46	33.59	±0.36	162.1	85.29	42.59	40.51 M
	NNConvdw	8.51	82.37	±0.53	95.3	43.89	21.89	23.80 M
	MCDec-dw	8.21	80.52	±0.54	95.3	43.89	21.89	23.80 M
ResNet-18	UpConv	8.95	45.21	±0.91	116.8	47.98	23.98	29.16 M
	NNConvdw	8.50	**142.46**	±8.30	50.0	6.58	3.28	12.46 M
	MCDec-dw	7.07	113.92	±6.39	50.0	6.59	3.28	12.46 M
DenseNet-56	UpConv	6.22	32.50	±0.40	73.4	53.05	26.43	18.33 M
	NNConvdw	7.25	77.14	±0.90	**6.6**	11.65	5.73	**1.62** M
	MCDec-dw	8.27	75.81	±2.24	**6.6**	11.65	5.73	**1.62** M
DenseNet-67	UpConv	4.65	26.29	±0.50	77.3	65.00	32.36	19.28 M
	NNConvdw	4.54	50.63	±0.42	10.5	23.60	11.66	2.57 M
	MCDec-dw	4.50	50.10	±0.22	10.5	23.60	11.66	2.57 M
MobileNet	UpConv	7.14	45.51	±1.11	82.8	43.83	21.90	20.67 M
	NNConvdw [29]	6.10	111.15	±6.88	16.0	**2.43**	1.20	3.96 M
	MCDec-dw	**10.05**	93.57	±4.64	16.0	2.44	1.20	3.96 M

Model characteristics present in this same table shows that our new decoder, MCDec-dw, has the same number of parameters, FLOPs and MACs as NNConvdw with a delta lower than 0.005G difference in FLOPs which leads to the conclusion that our method has a very low negative impact on model complexity compared to NNConvdw. Model checkpoint files are also the smallest for each encoder with our MCDec module. Tracking during testing also shows very low requirements in terms of computational overhead as our method requires as little as 0.54GB of RAM to run for the MobileNet-MCDec-dw combination. On the other hand, the inference shows more ambiguous results. For evaluation purposes, we averaged the number of FPS across 50 iterations of the model on a single image and calculated their mean and standard deviation, provided in Table 2, across 100 iterations of this protocol. Results show a slight decrease in

FPS on the GPU between our decoder, MCDec-dw, and NNConvdw due to the added complexity of our attention module. After further analysis, we have concluded that the standard deviation variance can be attributed to server usage. We have chosen for our experiments to run our tests on a SLURM cluster which, while it does not allow us to fully isolate the hardware for our specific running job, enables us to perform our experiments on lower performing GPUs and CPUs for better evaluation on frugal technology. As a result, internal node memory is subject to interference from other users or cluster management which can impact performances. On the one hand, this behavior, helps to simulate the influence of other running algorithms on a real-world system. On the other hand, inference measures become unstable when operating on CPU only which explains the close results between combinations. If we consider a vehicle that travels at 50km/h (13.89 m/s), we can conclude that MCDec-dw, when combined with any of our encoders, completes an analysis at worst every 0.28 m when running on GPU and 3.09 m on CPU.

Table 3. Performance of the models for different encoder-decoder configurations trained for 100 epochs on KITTI (320 × 256 resolution) with FastDepth [29] as {MobileNet+NNConvdw} and MC-Depth as {MobileNet+MCDec-dw}.

Encoder	Decoder	$\delta_1 \uparrow$	$\delta_2 \uparrow$	$\delta_3 \uparrow$	SiLog ↓	AbsRel ↓	MAE ↓	RMSE ↓
UNet	UpConv	0.881	0.920	0.952	3.308	0.386	0.030	0.100
	NNConvdw	0.893	0.944	0.979	0.914	0.379	0.027	0.087
	MCDec-dw	0.912	0.963	**0.988**	2.152	**0.303**	**0.022**	**0.073**
ResNet-18	UpConv	0.890	0.940	0.973	0.707	0.351	0.027	0.090
	NNConvdw	0.902	0.951	0.980	5.593	0.322	0.025	0.085
	MCDec-dw	0.903	0.958	0.984	0.552	0.322	0.024	0.080
DenseNet-56	UpConv	0.881	0.925	0.952	1.831	0.379	0.024	0.094
	NNConvdw	0.901	0.952	0.975	2.491	0.314	0.032	0.103
	MCDec-dw	0.902	0.962	**0.988**	0.415	0.333	0.024	0.078
DenseNet-67	UpConv	0.871	0.924	0.955	3.033	0.347	0.041	0.125
	NNConvdw	0.905	0.958	0.983	0.249	0.322	0.023	0.076
	MCDec-dw	0.911	0.964	**0.988**	0.626	0.318	0.023	0.076
MobileNet	UpConv	0.902	0.945	0.977	0.966	0.299	0.030	0.094
	NNConvdw [29]	0.920	0.960	0.980	0.276	0.376	0.024	0.083
	MCDec-dw	**0.922**	**0.967**	**0.988**	**0.137**	0.312	**0.022**	0.075

In terms of performance, as shown in Table 3 the combinations with our novel MCDec-dw decoder systematically increases performances compared to other decoder types. These results confirm that our additional cross-channel attention mechanism enables the model to better evaluate local features and capture interaction between adjacent information in the feature map. Our best

performing model combines a MobileNet backbone with a MCDec-dw decoder as shown in the last line of Table 3. In contrast, Table 2 shows it is only the third best architecture in terms of size as it counts up to 3.96M parameters. On the other hand, the smallest architectures, which have a DenseNet backbone, demonstrate in Table 3 lower performance in δ_1 accuracy but catch up with δ_3 accuracy results. Also, error rates have very close results, all of which highlights good global feature extraction and good geometrical understanding but lower precision for these combinations. We notice a significant gap in SiLog errors which range from 5.593 to 0.137. As explained in the original paper [6] this error is designed to credit uniformity in the direction and scale of errors related to their respective ground truth values. As a result, low error shows the models' ability to predict the relative geometry of the scene while remaining up to scale. It also shows that our decoder has a high capability to grasp local level features when used with our best performing combination.

Table 4. Comparison to state-of-the-art models.

Model	Training data	Parameters	GFLOPs	GMACs	$\delta_1 \uparrow$	RMSE $\downarrow$	GPU (FPS)
DepthAnythingV2 [31]	Custom	24.8 M	NA	12.28	0.851	0.220	53.86 ±1.58
Lite-Mono [34]	KITTI video	**3.07 M**	6.70	3.34	0.850	0.213	56.49 ±2.32
BoRe-Depth [13]	KITTI	8.66 M	14.37	7.13	0.871	0.237	18.49 ±0.31
FastDepth [29]	Small KITTI	3.96 M	**2.43**	**1.20**	0.920	0.083	**111.16** ±6.88
MC-Depth (Ours)	Small KITTI	3.96 M	2.44	**1.20**	**0.922**	**0.075**	93.57 ±4.64

While comparing our approach with other state of the art monocular depth estimation methods, Table 4 shows very high performance on the KITTI dataset with a smaller resolution. Following our "frugal" systemic approach, our models are trained on smaller images which provide less context during training. Comparatively, the other architectures we compare with our work have been trained on both larger datasets in terms of diversity but also with more context and high resolution images. As mentioned in Sect. 3, we also chose to retain non annotated regions of the data for training whereas previous work crop the useless data which causes their model to systematically fill in the blanks and attempt to predict values on the whole image instead of focusing on the areas of interest. For this study, we still tested all these models on our smaller resolution KITTI dataset. Results show that while being the second smallest network, our model outperforms significantly other models. In terms of inference, our method is only second to the FastDepth model which is explained by the addition of our efficient channel attention module. We can conclude from these results that for our expected field of application with a restricted sensor setup, our model is more efficient and more fitted to current needs than previous ready available models (Fig. 4).

Overall, Tables 2 and 3 confirm relatively high performance for very lightweight models with the added handicap of low resolution learning data. These

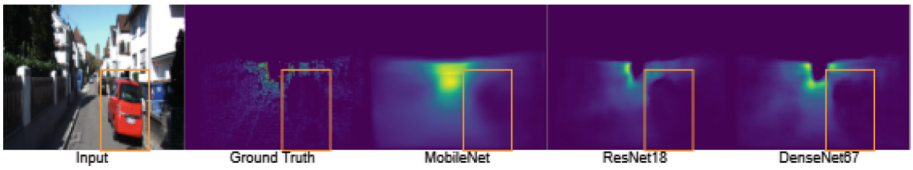


Fig. 4. Sample of subjective results of encoders combined with MCDec-dw.

results prove to show the potential of our models in a targeted context such as resource-constrained environment integration with a fleet of vehicles equipped with this system. It could be used in combination with other technologies to monitor surroundings for maintenance purposes.

6 Conclusion

We proposed a novel resource-efficient architecture and attention-based lightweight decoder for MDE on images of outdoors vehicular scenes. We contrasted our method with previous lightweight approaches and proved higher performance on the KITTI dataset with small resolution inputs conclusive with a real-world implementation. Using an ECA module our decoder surmounts the loss of context caused by depthwise separable convolutions without losing the advantage of a substantially reduced model size. In this paper, we focused on reducing model size without sacrificing performance while maintaining a highly modular architecture compliant with an easier system integration. In future perspectives, we aim to transfer the model's capacity to other scene types such as railway environment as well as integrating this approach into a more complete vehicular scene perception system.

Acknowledgments. We thank our partners at Ferrocampus with whom we collaborated on the TELLi project, financed by ADEME in the France 2030 campaign. This work benefited from access to the computing resources of the L3i laboratory, operated and hosted by the University of La Rochelle.Both are financed by the French government and the Region Nouvelle-Aquitaine.

References

1. Cheng, Z., Zhang, Y., Yu, Y., Song, Z., Tang, C.: TinyDepth: lightweight self-supervised monocular depth estimation based on transformer. Eng. Appl. Artif. Intell. **138**, 109313 (2024)
2. Chollet, F.: Xception: deep learning with depthwise separable convolutions arXiv:1610.02357 [cs] (2017)
3. Cordts, M., et al.: The cityscapes dataset for semantic urban scene understanding. arXiv:1604.01685 (2016)

4. Dong, X., Garratt, M.A., Anavatti, S.G., Abbass, H.A., Dong, J.: Lightweight monocular depth estimation with an edge guided network. arXiv:2209.14829 [cs] (2022)
5. Dosovitskiy, A., et al.: An image is worth 16x16 words: transformers for image recognition at scale. arXiv:2010.11929 cs (2021)
6. Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. arXiv:1406.2283 cs (2014)
7. Evchenko, M., Vanschoren, J., Hoos, H.H., Schoenauer, M., Sebag, M.: Frugal Machine Learning (2021)
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778 (2016)
9. Hu, J., Shen, L., Albanie, S., Sun, G., Vedaldi, A.: Gather-excite: exploiting feature context in convolutional neural networks. arXiv:1810.12348 cs (2019)
10. Hu, J., Shen, L., Albanie, S., Sun, G., Wu, E.: Squeeze-and-Excitation Networks. arXiv:1709.01507 cs (2019)
11. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2261–2269 (2017)
12. Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., Navab, N.: Deeper depth prediction with fully convolutional residual networks. In: Fourth International Conference on 3D Vision (3DV), pp. 239–248 (2016)
13. Liu, C., Li, J., Zhang, S., Liu, C., Li, J., Zhang, X.: BoRe-Depth: self-supervised monocular depth estimation with boundary refinement for embedded systems. arXiv:2511.04388 [cs] (2025)
14. Liu, J., Zhang, Y.: High quality monocular depth estimation with parallel decoder. Sci. Rep. **12**, 16616 (2022)
15. Liu, X., Tang, S., Feng, M., Guo, X., Zhang, Y., Wang, Y.: A simple monocular depth estimation network for balancing complexity and accuracy. Sci. Rep. **15**(1), 12860 (2025)
16. Ma, N., Zhang, X., Zheng, H.T., Sun, J.: ShuffleNet V2: practical guidelines for efficient CNN architecture design. In: Proceedings of the European Conference on Computer Vision (ECCV), vol. 11205, pp. 116–131 (2018)
17. Ma, R., Chen, S., Zhang, Q.: Lightweight monocular depth estimation. arXiv:2212.11363 [cs] (2022)
18. Masoumian, A., Rashwan, H.A., Cristiano, J., Asif, M.S., Puig, D.: Monocular depth estimation using deep learning: a review. Sensors **22**(14), 5353 (2022)
19. Miller, K., Lohn, A.: Onboard AI: Constraints and Limitations. Tech. rep, Center for Security and Emerging Technology (2023)
20. Rajapaksha, U., Sohel, F., Laga, H., Diepeveen, D., Bennamoun, M.: Deep learning-based depth estimation methods from monocular image and videos: a comprehensive survey. ACM Comput. Surv. **56**(12), 315:1–315:51 (2024)
21. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. arXiv:1505.04597 cs (2015)
22. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: MobileNetV2: inverted residuals and linear bottlenecks. arXiv:1801.04381 [cs] (2019)
23. Saxena, A., Chung, S., Ng, A.: Learning depth from single monocular images. In: Advances in Neural Information Processing Systems, vol. 18, MIT Press (2005)
24. Sui, X., Gao, S., Xu, A., Zhang, C., Wang, C., Shi, Z.: Lightweight monocular depth estimation using a fusion-improved transformer. Sci. Rep. **14**(1), 22472 (2024)

25. Tian, X., He, Z., Zhang, Y., Liu, F., Gu, T.: MsGf: a lightweight self-supervised monocular depth estimation framework with multi-scale feature extraction. Sensors **25**(20), 6380 (2025)
26. Vaswani, A., et al.: Attention Is All You Need. arXiv:1706.03762 cs (2023)
27. Wang, P., Liu, S., Li, Q., Yi, Y., Wang, J.: MG-Mono: a lightweight multi-granularity method for self-supervised monocular depth estimation. Patt. Recogn. **171**, 112223 (2026)
28. Wang, Q., Wu, B., Zhu, P., Li, P., Zuo, W., Hu, Q.: ECA-Net: efficient channel attention for deep convolutional neural networks. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 11531–11539 (2020)
29. Wofk, D., Ma, F., Yang, T.J., Karaman, S., Sze, V.: FastDepth: fast monocular depth estimation on embedded systems. arXiv:1903.03273 (2019)
30. Woo, S., Park, J., Lee, J.Y., Kweon, I.S.: CBAM: convolutional block attention module. arXiv:1807.06521 cs (2018)
31. Yang, L., Kang, B., Huang, Z., Xu, X., Feng, J., Zhao, H.: Depth anything: unleashing the power of large-scale unlabeled data. arXiv:2401.10891 (2024)
32. Yang, L., et al.: Depth Anything V2. arXiv:2406.09414 (2024)
33. Yvinec, E.: Efficient Neural NetworksâĂŤ: Post Training Pruning and Quantization. phdthesis, Sorbonne Université (2023)
34. 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, pp. 18537–18546 (2023)
35. Zhao, B., He, H., Xu, H., Shi, P., Hao, X., Huang, G.: RTIA-Mono: real-time lightweight self-supervised monocular depth estimation with global-local information aggregation. Digital Sign. Process. **156**, 104769 (2025)
36. Zhao, H., et al.: PSANet: point-wise spatial attention network for scene parsing. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11213, pp. 270–286. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01240-3_17