IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 30, 2021, accepted May 18, 2021, date of publication May 21, 2021, date of current version May 28, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3082697

Exploiting Motion Perception in Depth Estimation Through a Lightweight Convolutional Neural Network

PEDRO NUNO LEITE AND ANDRY MAYKOL PINTO
Institute for Systems and Computer Engineering, Technology and Science (INESC TEC), 4200-465 Porto, Portugal
Faculty of Engineering, University of Porto, 4200-465 Porto, Portugal

Corresponding author: Pedro Nuno Leite (pedro.nuno@inesctec.pt)

This work was supported in part by the National Funds through the Portuguese Funding Agency Fundação para a Ciência e a Tecnologia (FCT) under Project UIDB/50014/2020. The work of Pedro Nuno Leite was supported in part by the Portuguese Government through the Fundação para a Ciência e a Tecnologia (FCT) by the Ph.D. Grant under Grant 2020.06949.BD.

ABSTRACT Understanding the surrounding 3D scene is of the utmost importance for many robotic applications. The rapid evolution of machine learning techniques has enabled impressive results when depth is extracted from a single image. High-latency networks are required to achieve these performances, rendering them unusable for time-constrained applications. This article introduces a lightweight Convolutional Neural Network (CNN) for depth estimation, NEON, designed for balancing both accuracy and inference times. Instead of solely focusing on visual features, the proposed methodology exploits the Motion-Parallax effect to combine the apparent motion of pixels with texture. This research demonstrates that motion perception provides crucial insight about the magnitude of movement for each pixel, which also encodes cues about depth since large displacements usually occur when objects are closer to the imaging sensor. NEON's performance is compared to relevant networks in terms of Root Mean Squared Error (RMSE), the percentage of correctly predicted pixels (δ_1) and inference times, using the KITTI dataset. Experiments prove that NEON is significantly more efficient than the current top ranked network, estimating predictions 12 times faster; while achieving an average RMSE of 3.118 m and a δ_1 of 94.5%. Ablation studies demonstrate the relevance of tailoring the network to use motion perception principles in estimating depth from image sequences, considering that the effectiveness and quality of the estimated depth map is similar to more computational demanding state-of-the-art networks. Therefore, this research proposes a network that can be integrated in robotic applications, where computational resources and processing-times are important constraints, enabling tasks such as obstacle avoidance, object recognition and robotic grasping.

INDEX TERMS CNN, depth estimation, mobile robotics, motion perception, parallax.

I. INTRODUCTION

Autonomous systems require a complete and accurate 3D perception of its surroundings to operate. Nevertheless, enabling this awareness remains an extremely challenging problem in computer vision. Advances in this field directly impact a myriad of applications, e.g. autonomous driving, augmented reality and robotics.

The acquisition of three-dimensional information is limited in terms of the range and resolution of the sensor. Recent developments in machine learning techniques have expanded the possibilities for 3D scene representation, enabling depth information to be estimated from a single image. State-of-the-art results that emerge from these techniques are quite accurate; however, most depend on powerful deep convolutional networks [1], [2], rendering them unusable in time-constrained applications such as robotic grasping, obstacle avoidance, 3D reconstruction and object recognition [3]–[6].

The ability to comprehend visual changes in a sequence of images can be referred to as motion perception. It plays a crucial part in autonomous systems, providing relevant information about the speed and direction of any moving object in a scene [7]. Additionally, information that directly translates motion can also be valuable for 3D perception. This article proposes a lightweight encoder-decoder architecture,

The associate editor coordinating the review of this manuscript and approving it for publication was Byoung Wook Choi.

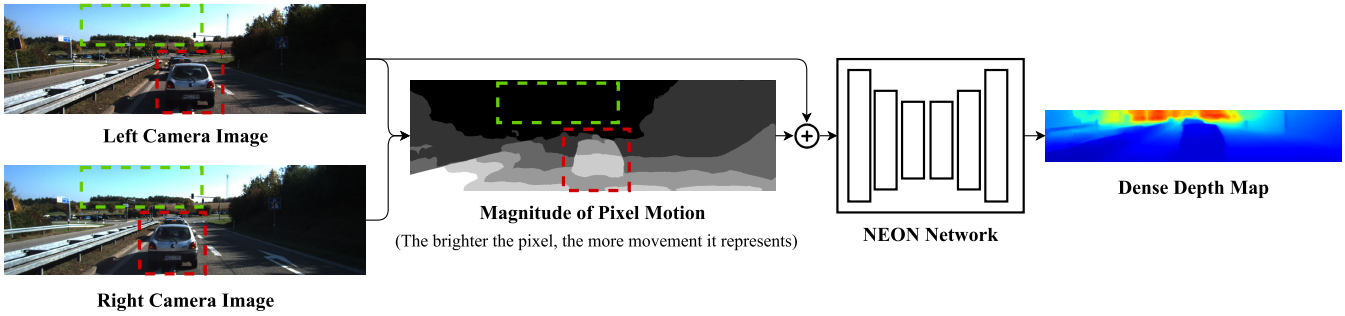


FIGURE 1. A visual representation of the proposed approach. The magnitude of motion allows for a relative sense of depth, obtained by exploiting the notion of Motion-Parallax: objects closer to an observer (highlighted in red) move faster than objects further away (highlighted in green). NEON combines this input with the texture-based features captured from an RGB image to estimate highly reliable depth maps, in a fraction of the time when compared to current State-of-the-Art solutions.

NEON (deNse dEpth from mOtioN-parallax), that exploits the notion of Motion-Parallax - *objects closer to an observer are perceived to move faster than objects further away* [8] - by making use of the magnitude of pixel motion in a flow field. This information is combined with texture-based features from an RGB image, to obtain a dense and accurate depth map of the scene. A visual representation of the proposed pipeline is presented in Fig.1.

By introducing the magnitude of motion as part of the network's input, the learning task shifts focus from solely learning how to predict depth from texture, shadows and occlusions; to also weighing in which magnitude of motion corresponds to what depth value. This allows the network's complexity to be reduced, being capable of estimating depth maps with low-latency, while maintaining high levels of accuracy. In summary, the contributions of this work include:

- A lightweight Convolutional Neural Network (CNN), designed to balance performance and inference times. Near State-of-the-Art results are achieved - the average absolute difference in terms of Root Mean Squared Error is 36 cm. NEON takes a fraction of the time to estimate predictions, being 12 times faster than the currently top ranked network.
- A thorough study on the impact of the introduction of the motion perception features as an additional input to the network. Ablation studies indicate a significant performance gain when the Motion-Parallax effect is exploited; and reveal that the network gives more weight to the newly introduced information than to texture-based features.
- An extensive benchmark analysis of the results obtained from evaluating NEON's performance on Eigen's [9] split of the KITTI dataset. Being composed of real world data, this dataset allows NEON to be tested in a multitude of scenarios, under various lighting conditions and phenomena. Finally, NEON is compared to the most relevant networks to date.

The remainder of this article is structured as follows: Section II shortly presents a literature review; Section III introduces the architecture of the proposed network; All the conducted experiments are presented in section IV; Finally, section V summarizes the conducted work and provides a critical view of the obtained results.

II. RELATED WORK

Traditional methods rely on hand-crafted features and probabilistic models. The advances in deep learning bypass this step, extracting important characteristics directly from the input. This section presents relevant works in the scope of monocular (section II-A) and motion-based (section II-B) depth estimation, as well as efficient networks capable of carrying out this task (section II-C).

A. MONOCULAR DEPTH ESTIMATION

1) SUPERVISED LEARNING APPROACHES

A supervised approach at estimating monocular depth is first presented in the work of Saxena *et al.* [10]. The image is divided into small patches and hand-crafted filters are used to extract relevant local features. Finally, a Markov Random Field (MRF) models the relation between patches, achieving a global representation of the scene. This work is later extended to a patch-based model that learns both the 3D location and orientation of each patch [11]. Eigen *et al.* [9] propose a complementary two-stack network, where first a coarse prediction is inferred at a global level, and it is later refined locally by a fine-scale network. Comparatively to previous approaches, this is the first work to learn directly from pixel information without the need of hand-crafted features. A combined application of a CNN and a Conditional Random Field is introduced by Liu *et al.* [12] achieving a more detailed and visually sharp prediction. Kuznietsov *et al.* [13] propose a semi-supervised method. The sparse LiDAR ground truths are used during training, however the learning is conditioned to infer photoconsistent depth maps by means of a direct image pair alignment loss. The work of Fu *et al.* [1] shapes the learning of a depth map as a quantized ordinal regression while avoiding unnecessary spatial pooling operations, in order to obtain faster convergence times. Yin *et al.* [14] introduce 3D geometric constraints (virtual normal directions) as a loss term. This approach leads the network to

encode strong global constraints, that in turn allow for a less noisy and more faithful projection of the estimated depth map into the 3D space. In 2020, Johnston and Carneiro [15] proposed two mechanisms to improve self-supervised monocular depth estimation: self-attention, to enable the exploitation of more general features at non-contiguous regions; and, discrete disparity prediction which provides robust and sharper depth maps. This approach leads to the best results, in the KITTI benchmark, so far as the self-supervised learning task is concerned.

The current State-of-the-Art results are achieved by the Big-to-Small (BTS) network, introduced by Lee *et al.* [2]. Local planar guidance layers are incorporated into several stages along the decoding phase. This novel layer allows for an explicit relation from internal feature maps to the desired prediction. This methodology shows significant improvements when compared to previous works, however, frequent artifacts can be observed in predicted depth maps for the KITTI benchmark dataset. The authors refer to the sparsity of the ground truth data as the catalyst for this issue.

2) UNSUPERVISED LEARNING APPROACHES

An unsupervised learning method is presented by Godard *et al.* [16], where depth estimation is formulated as an image reconstruction problem. This approach makes use of stereo image pairs, and tries to predict one perspective from the other by using the estimated disparities. The estimation error is used as the loss function, enforcing left-right consistency. Garg *et al.* [17] also treat this problem as an image reconstruction, proposing an unsupervised encoder-decoder network that is optimized through the photometric error in the reconstruction. Mathew and Mathew [18] introduce an unsupervised technique based on a shared encoder-decoder design. It makes use of stereo pairs during the training process, estimating a depth map for each of the images. The target view is obtained by inverse warping the source view, using the geometric camera projection, and the learning is enforced through a stereo positive-negative loss.

B. ESTIMATING DEPTH FROM MOTION

Estimating depth from relative pixel motion is a complex computer vision problem. Classic mathematical approaches exploit camera motion, spline approximations and epipolar geometry to infer relative depth maps [19]–[21]. The work of Pathak *et al.* [22] takes advantage of the motion parallax phenomena in a virtual reality context. Two spherical images are used to estimate a dense optical flow, which is then decomposed into a relative depth map.

When it comes to machine learning approaches, the work proposed by Mancini *et al.* [23] uses both RGB images and optical flows as inputs for a CNN, to estimate depth maps for an obstacle detection task. This methodology prioritizes fast inference times, sacrificing the accuracy of the estimated predictions. Training is conducted with synthetic data and validation is later conducted on the KITTI dataset. The network is capable of running at 30 Hz on an NVIDIA Tesla K40 GPU, achieving an average RMSE of 7.508 m. The authors address the methodology's weakness in close-range estimations, which limits the practical application of this work. Additionally, training with synthetic data does not account for the implications of real world scenarios, which would further hinder the network's performance. A different approach is presented by Ummenhofer *et al.* [24], where the learning task mimics a typical structure-from-motion problem. The DeMoN network is composed of two consecutive encoder-decoder structures. The first predicts the optical flow, whereas the second takes this prediction into consideration while inferring depth maps and surface normals. A comparable approach is presented by Wang and Shen [25] where the network first jointly estimates optical flow and camera motion. A triangulation layer is then proposed to encode this information and, finally, a depth map is estimated. Jiang *et al.* [26] propose a self-supervised network, that is pretrained with relative depth maps obtained from the information encoded in the optical flow's motion field. The work of Chen *et al.* [27] also couples together the learning of both optical flows and depth. Their network simultaneously learns to predict optical flows and depth maps, benefiting from exchanges of information between both segments of the network. The research presented above gains from jointly learning how to estimate depth and interpret motion, however, shaping the learning task in such a way requires a very high degree of network complexity. The use of cascaded or parallel networks that share information and weights, not only suffers from a much more arduous training process, but also limits the applicability of these methodologies in real-time tasks, due to the sheer amount of necessary calculations.

C. EFFICIENT ARCHITECTURES FOR DEPTH ESTIMATION

Most of the works presented above rely on powerful networks, which lead to inference times that are unsuitable for real-time applications. Few networks have been specially designed for low latency (e.g. MobileNet [28], [29], EfficientNet [30]) however, these are tailored to perform feature extraction, whereas very little emphasis is put into efficient upsampling methods. The work of Wofk *et al.* [31] focus precisely on this problem, proposing an encoder-decoder network with low latency upsampling layers. Pruning techniques are used to further reduce computational costs, achieving 178 Hz on an NVIDIA Jetson TX2 embedded GPU platform, for a single RGB input of size 224×224. Fast inference times are also achieved by the architecture introduced by Nekrasov *et al.* [32], composed by a MobileNet V2 and a lightweight RefineNet [33]. It takes approximately 13 ms, on an NVIDIA Geforce GTX 1080 Ti GPU, to estimate both a monocular-based depth map and the correspondent semantic segmentation of the scene.

A few efficient unsupervised techniques have also been explored in the literature. Poggi *et al.* [34] make use of a pyramidal structure-based network to estimate monocular depth, achieving 2 Hz on the Raspberry Pi 3 and 40 Hz on a standard CPU. Liu *et al.* [35] make use of a recurrent module within

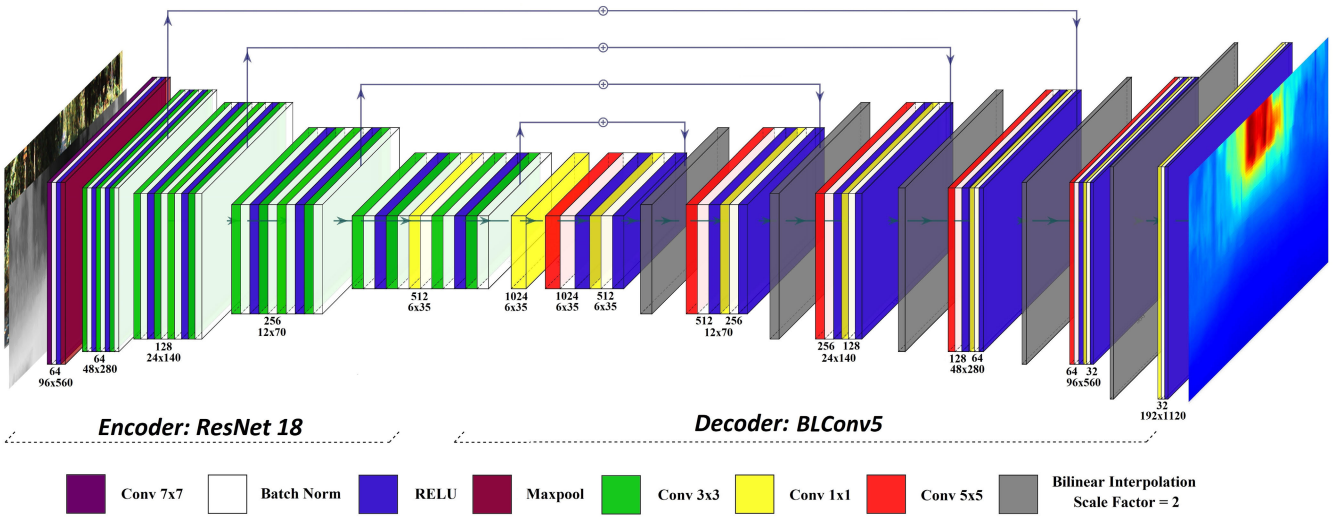


FIGURE 2. The general encoder-decoder architecture of the proposed network (NEON): the encoder, is composed of a ResNet18 followed by a 1 × 1 convolutional layer that asserts the number of channels to the decoder, which is comprised of five upsampling layers, each followed by a bilinear interpolation with a scale factor of two. Skip additive connections are implemented between both. Feature maps are presented as Channels x Height x Width.

their encoder, as well as a novel upsampling method based on depthwise separable convolutions, to achieve satisfactory performance while keeping the network as lightweight as possible. This approach is capable of running at 110 Hz on an NVIDIA Geforce GTX 1080 Ti GPU and about 2 Hz on a Raspberry Pi 3.

Most of the works presented above sacrifice accuracy in order to have low latency. In comparison, this research presents an approach capable of reaching near state-of-the-art performances while spending 6.4 ms per frame. The short inference times are achieved through the design of a lightweight architecture, and, the high depth map accuracy is assured through the additional input (the magnitude of motion), which proved to have the desired impact on the network's performance.

III. NEON-A LIGHTWEIGHT NETWORK FOR DEPTH ESTIMATION

The proposed lightweight CNN is introduced in section III-A. A thorough discussion of how motion perception can be exploited in depth estimation tasks is presented in section III-B. Finally, the most relevant implementation details are discussed along section III-C.

A. NETWORK ARCHITECTURE

The proposed network is based on an encoder-decoder structure. Its full architecture is presented in Fig.2. The encoder is responsible for extracting high-level characteristics from the input. These become a set of low-resolution feature maps that need to be merged and upsampled, so that the network is able to output a dense and full resolution depth map. This task is handled by the decoder. In the context of robotic applications, processing power is a constrained resource. For this reason, this work aims to develop a lightweight network capable of inferring predictions in useful time, i.e. a few milliseconds, while having minimal impact on accuracy.

The chosen encoder is based on the ResNet18. Its residual connections allow for an easier optimization due to the improved flow of gradients [36], while providing a good trade-off between accuracy and inference times (refer to section IV-A). The ResNet is adapted from image classification tasks, therefore some modifications are necessary. The first convolutional layer has been altered to support a 4 channel input (an RGB image where the magnitude of motion is added as the fourth channel), and the standard classification layers removed and replaced with a 1 × 1 convolution that asserts the correct number of channels for the decoder.

On the other hand, the decoder is the most time consuming part of these types of network structures [31]. The use of depthwise separable convolutions results in significantly lower computational times. This concept is based on factorizing a standard convolution into two operations: a depthwise convolution followed by a 1 × 1 pointwise convolution [37]. The first applies a filter, whereas the second merges the outputs. Contrary to a standard convolution, these depthwise layers convolve with a single channel at a time, hence the lower complexity [31]. The chosen decoder is composed by five layers, where the upsampling is carried out by depthwise convolutions followed by bilinear interpolation with a scale factor of 2.

Similarly to the residual connections between layers within the ResNet architecture, there are also benefits from mirroring the same logic between encoder and decoder. The detailed features encoded from the original input can be lost as these become iteratively more compact feature maps. Skip connections allow high-level features from the encoder to be

merged with low-level ones, during the upsampling process, leading to a more detailed depth map. Fig.2 depicts the four implemented connections between each ResNet block and the corresponding upsampling layer.

B. EXPLOITING MOTION PERCEPTION IN A DEPTH ESTIMATION TASK

The three-dimensional movement of objects in a scene - translations and rotations - can be represented in a two-dimensional space. The 2D vectors that translate this motion, not only offer information about speed and direction, but also encode information about depth.

An observer moving in a static scene can be defined with a motion profile composed by: a translational velocity, $\dot{\Gamma} = (t_X, t_Y, t_Z)^T$; and a rotational velocity, $\Omega = (w_X, w_Y, w_Z)^T$. Since both components are described in the imaging sensor's coordinate system, said system moves along with the camera. A point $P = (X, Y, Z)^T$ represented in this coordinate space, will thus change its position accordingly.

Considering that this point P is perceived to move away from the camera, its velocity $\dot{P}$ is given by [8]:

$$\dot{P} = -\dot{\Gamma} - \Omega \times P \tag{1}$$

For a pin-hole camera model, x and y can be expressed as:

$$x = f\frac{X}{Z}; \quad y = f\frac{X}{Z} \tag{2}$$

where f represents the camera's focal length. The two-dimensional velocity vector v, of the projected point p, can thus be represented by:

$$v \equiv (u, v)^T = \begin{bmatrix} \frac{\partial x}{\partial t} \\ \frac{\partial y}{\partial t} \end{bmatrix} \tag{3}$$

Substituting equation 2 into 3:

$$v = \frac{f}{Z}\begin{bmatrix} \dot{X} - \frac{X\dot{Z}}{Z} \\ \dot{Y} - \frac{Y\dot{Z}}{Z} \end{bmatrix} = \frac{1}{Z}\begin{bmatrix} f\dot{X} - \dot{Z}x \\ f\dot{Y} - \dot{Z}y \end{bmatrix}$$
$$= \frac{1}{Z}\begin{bmatrix} f & 0 & -x \\ 0 & f & -y \end{bmatrix}\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} \tag{4}$$

Now using equation 1:

$$v = \frac{1}{Z}\begin{bmatrix} -f & 0 & x \\ 0 & -f & y \end{bmatrix}\dot{\Gamma} + \frac{1}{Z}\begin{bmatrix} -f & 0 & x \\ 0 & -f & y \end{bmatrix}\begin{bmatrix} 0 & Z & -Y \\ -Z & 0 & X \\ Y & -X & 0 \end{bmatrix}\Omega \tag{5}$$

Finally, the two-dimensional velocity of p can be formulated as:

$$v = \frac{1}{Z}\begin{bmatrix} -f & 0 & x \\ 0 & -f & y \end{bmatrix}\dot{\Gamma} + \begin{bmatrix} \frac{xy}{f} & -f - \frac{x^2}{f} & y \\ f + \frac{y^2}{f} & -\frac{xy}{f} & -x \end{bmatrix}\Omega \tag{6}$$

Two major conclusions can be inferred from equation 6. The first relates to the fact the vector sum of the velocity does not directly depend on the three-dimensional position of point P within the camera's coordinate system. The apparent motion, therefore, solely refers to the observer. Additionally, the inverse of the depth appears coupled to the translational component of the velocity, acting as a scaling factor: the closer point p is to the observer, the faster it is perceived to move. This phenomena is denominated Motion-Parallax. Fig. 3 depicts a practical example of how the parallax effect can be exploited to infer a relative sense of depth. The information extracted from motion perception can thus be a valuable addition to NEON's input data.

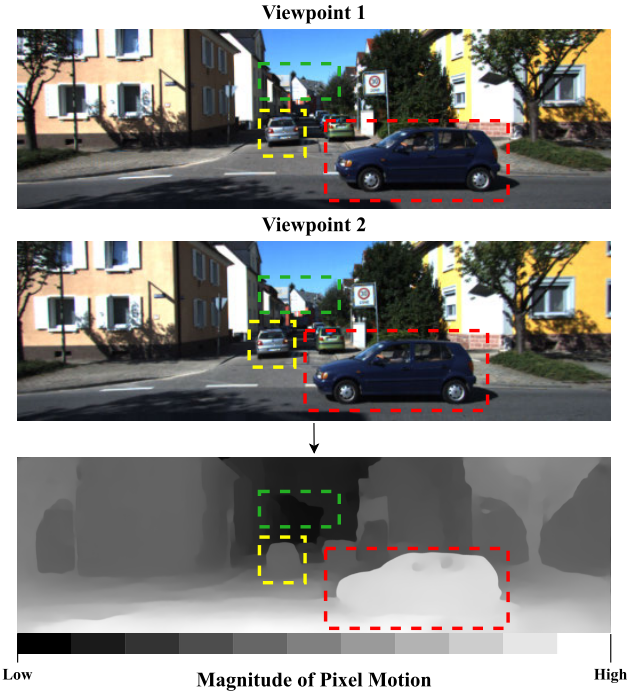


FIGURE 3. Motion perception allows for a relative sense of depth to be extracted from the 2D projection of motion vectors. Considering two distinct viewpoints of the same static scene, the pixel displacement in between both encodes information about its depth. The higher the magnitude of motion associated to a pixel, the closer it is to the observer and vice-versa (refer to the highlighted regions of the image).

C. IMPLEMENTATION DETAILS

The KITTI dataset [38] is one of the most well known benchmark datasets for depth prediction tasks. It is composed of 61 outdoor scenes, captured in different contexts, providing visual information from a stereo camera pair with a 375 × 1241 resolution, as well as point cloud data gathered from a 64 channel Velodyne LiDAR. The split of the KITTI data proposed by Eigen *et al.* [9] is used for training

(23158 instances) and evaluation (697 instances) purposes. The ground-truth data is a 2D projection of the LiDAR point-cloud, therefore, the loss of a degree implies the loss of some information. To mitigate the issues of training with sparse data, the parts of the image where there is no corresponding depth information are cropped, and, the ground-truth is infilled by interpolating the missing depth values according to the colorization algorithm proposed by Levin *et al.* [39].

Motion perception can introduce valuable information in a depth estimation task. To exploit this information, the intrinsic movement of each pixel must be estimated. Optical flow techniques are a classic approach to obtain the motion vectors for each pixel [40]. Recent deep learning techniques improve upon this concept being capable of estimating fast and robust flow fields. The FlowNet 2.0 proposed by Ilg *et al.* [41], the current state-of-the-art network, is used offline to obtain the spatial evolution of pixels, assigning them the respective motion vector.

Motion vectors are often estimated through a sequential approach, i.e. pixel $I(x, y)$ at time t, moves by δ_x and δ_y during the time interval δ_t:

$$I(x, y, t' = t + \delta_t) = I(x + \delta_x, y + \delta_y, t) \tag{7}$$

Instead, the flow field is calculated from a stereo pair of images captured at the same time instant t:

$$I_{Right}(x, y, t) = I_{Left}(x + \delta_x, y + \delta_y, t) \tag{8}$$

This way, one is able to obtain two different viewpoints from the same static scene, which is not possible with the sequential approach. Additionally, the entropy caused by the constant speed changes in a moving vehicle, which can break the optical flow constraints if the movement of the pixels becomes too steep, can be bypassed.

The motion vectors are then transformed into polar coordinates, and, since the aim is to obtain how much a pixel moves and not its direction, only the magnitude is used as input for the network. This information is finally concatenated with the RGB image, forming an 4x192x1120 input. Additionally, to increase the generalization capabilities of the trained models, data augmentation is performed:

- Random rotations (between $[-5, 5]$ degrees) are applied with a 60% probability;
- Horizontal flips with a 60% probability;
- Further augmentation is done to the RGB image by randomly shifting the brightness, contrast and gamma values;
- Motion blur is added to the RGB image to simulate a vehicle's movement.

Vertical flips are not considered because of the application context, e.g. the road should always be on the bottom.

IV. EXPERIMENTS

This section thoroughly discusses the results obtained by evaluating NEON's performance in Eigen *et al.*'s [9] test split of the KITTI dataset. Along this section, ablation studies are presented for both the network's design and choice of input modalities. Finally, a state-of-the-art benchmark is provided and conclusions are inferred by using the work of Lee *et al.* [2] as baseline for comparison. All the discussed experiments were run with an NVIDIA GeForce RTX 2080 SUPER with 8GB of VRAM, and the Intel i5-8600K CPU @ 3.60GHz with 6 cores. The implementation of the NEON network was conducted with the deep learning framework Pytorch. Optimization is carried out through the Stochastic Gradient Descent algorithm, with a learning rate of 0.1, decreased by 10 every 3 epochs. Training is conducted with a batch size of 8, during 12 epochs and the Huber loss is used to guide the learning task. A weight-decay of $1e-3$ is applied to ensure regularization.

The following metrics are used to evaluate the inferred predictions during this section:

- **RMSE**: Root Mean Squared Error;
- **REL**: Mean Absolute Relative Error;
- δ_n: Percentage of predicted pixels whose relative error is within a threshold (refer to (9)).

$$\delta_n = \frac{card(\{\hat{y} : max\{\frac{\hat{y}_i}{y_i}, \frac{y_i}{\hat{y}_i}\}) < 1.25^n}{card(\{y_i\})} \tag{9}$$

where y_i and $\hat{y}_i$ are the ground truth and prediction, respectively. *card* represents the cardinality of a set.

Since the global scale of a depth map can be ambiguous [9], the Scale-Invariant Logarithmic Error (SIlog) is also considered as a metric of evaluation:

$$SIlog = \frac{1}{n}\sum_i (d_i)^2 - \frac{1}{n^2}(\sum_i d_i)^2 \tag{10}$$

where n is the number of valid pixels in the ground truth, and d_i is given by:

$$d_i = \log(\hat{y_i}) - \log(y_i) \tag{11}$$

where $\hat{y_i}$ and y_i represent the prediction and ground truth, respectively.

Finally, since fast processing times are indeed a concern for robotic applications, the model's efficiency - i.e. the elapsed GPU time during inference - is also considered as an evaluation metric.

A. ABLATION STUDIES: NETWORK DESIGN

During the design of the architecture presented in section III-A, multiple encoders - MobileNet, ResNet18, VGG11, DenseNet121 - and decoders - BLConv5, DeConv5, UpProj, UpConv - were experimented with. This section aims at discussing the most relevant conclusions inferred from these experiments, justifying the choice of the ResNet18 and BLConv5 for the encoder-decoder structure, as well as the benefits of having additive connections between both. Every model was trained with the same hyperparameters, as described in section IV, and the Huber loss function.

1) ENCODER

The extraction and encoding of high-level features is a crucial part of any network. To choose an encoder with the best possible compromise between accuracy and complexity, multiple networks that have been adapted and used in state-of-the-art depth estimation applications were studied. For sake of a fair comparison, the BLConv5 (with depthwise convolutions) was used as the decoder across every experiment.

Similarly to the process described in section III-A, each of the presented encoders was adapted to work with a four channel input and, the classification layers were removed and substituted with a convolution layer that asserts the number of necessary channels for the decoder.

The VGG network [42] has been used in several depth estimation works (e.g. [1], [26]), however its large amount of Multiply-Accumulate[1] (MAC) operations (see table 1) make it rather time consuming to train. When compared with a ResNet, trained within the same conditions, the latter does perform better as seen in the work presented by Fu *et al.* [1]. The simplest version of this network, VGG11, was used during these experiments. The results presented in table 1 align with the aforementioned statement, since VGG has the worst performance out of the remaining encoders, with a 3.406 m RMSE and 93.7% of correctly predicted pixels.

TABLE 1. Comparison between multiple encoders. For a fair comparison the BLConv5 decoder was fixed across every experiment. The MAC operation count was obtained for a 4x192x1120 input (CxHxW) and doesn't take the decoder into account.

Encoder-BLConv5	RMSE (m)	SIlog ($\log(m)$ %)	δ_1 (%)	#Params (M)	MACS (G)	Time (ms)
MbN	3.302	5.69	93.8	3.21	2.51	**4.8**
RN18	3.184	5.41	94.1	11.7	8.09	5.8
VGG11	3.406	**5.33**	93.7	9.23	44.27	12.2
DN121	**3.120**	5.51	**94.3**	6.96	12.52	18.2

Lower is Better: RMSE, SIlog, Inference Time
Higher is Better: δ_1
MbN = MobileNet; RN = ResNet; DN = DenseNet

A network that has commonly been used for this regression task is the ResNet [36]. Its residual connections allow for the training of deeper networks while avoiding vanishing gradients. This residual network and its variants are at the core of state-of-the-art architectures such as the BTS network presented by Lee *et al.* [2]. On the other hand, the DenseNet [43] also promotes a strong gradient flow, since each consecutive block is an iterative concatenation of previous feature maps. From testing both in this application, one is able to conclude that these achieve very similar results (in terms of RMSE and δ_1, refer to table 1). However, when comparing both SIlog percentages (5.41% and 5.51%, respectively) the ResNet does infer a more faithful representation of the scene, i.e. it better captures its scale. Furthermore, the DenseNet takes the longest time to estimate a prediction out of the four encoders, approximately 3.2 times higher than the ResNet.

[1]Estimated with: https://github.com/sovrasov/flops-counter.pytorch

Finally, the MobileNet [28] is one of the most lightweight networks that have been used for estimating depth maps in real-time [31]. Its low amount of parameters and MAC operations (2.51G) make it the fastest network to train as well as the quickest to infer a prediction (4.8 ms), while still performing rather well. When comparing it to the ResNet, the latter is able to achieve better results, being marginally slower. Therefore, from this set of experiments, the ResNet can be considered the most suitable network for a performance-latency compromise.

2) DECODER

The second part of the architecture, the decoder, is responsible for merging the features extracted by the encoder and upsampling them to a given resolution. The ResNet18 is the encoder of choice, while several upsampling layers were studied on a fixed five layer decoding structure with a final convolution at the end.

The work presented by Laina *et al.* [44] introduces two distinct upsampling layers:

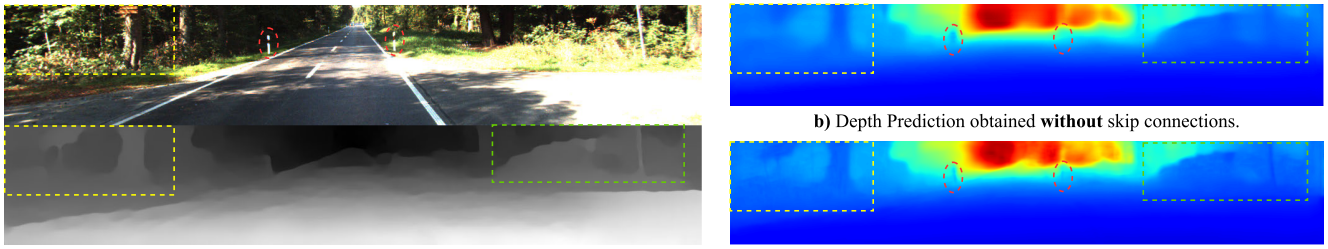
- UpConv: composed by a 2×2 unpooling stage followed by a 5×5 convolution;
- UpProj: composed by a 2×2 unpooling operation followed by two branches: one performing both a 5×5 and a 3×3 convolution; and the other with a single 5×5 convolution. Both branches are added up and fed to a ReLU layer.

The UpProj module leads to the better quantitative results in terms of RMSE and δ_1, with 3.181 m and 94.4%, respectively. It also performs the best when it comes to capturing the global scale of the scene, reaching a percentage of 5.04 SIlog (see table 2). On the downside, this decoder employs a rather complex structure, which is reflected in the higher number of learnable parameters and MAC operations (120G), which consequently lead to a high inference time of 41.5 ms. The UpConv, on the other hand, offers a better complexity-accuracy trade-off, halving the time of inference at the cost of a slightly less accurate prediction (average RMSE of 3.225 m).

Further comparison can be drawn with the decoder variants proposed by Wofk *et al.* [31]:

- BLConv5: composed by a 5×5 convolution followed by bilinear interpolation with scale factor of two;
- DeConv5: composed by a 5×5 transpose convolution.

The DeConv5 performs metrically worse than the remaining decoders, achieving an RMSE of 3.438 m, as well as a percentage of correctly predicted pixels of 93.8%. When it comes to the BLConv5, even though the RMSE is higher (3.215 m), and the overall quality of the prediction is slightly worse when compared to the UpProj module (both in terms of δ_1 - 94.1%; and global scale - 5.24% SIlog); it presents itself as the better option for this architecture. The BLConv5 massively outperforms the UpProj in inference times, being capable of outputting a prediction 4.2 times faster.



a) Network Inputs (from top to bottom): RGB image; Magnitude of Motion.

b) Depth Prediction obtained **without** skip connections.

c) Depth Prediction obtained **with** skip connections.

FIGURE 4. Impact of skip additive connections. Dense depth maps inferred from the same architecture, with and without skip connections between encoder-decoder. The more detailed prediction, in c), is the result of the addition of high-level features from the encoder into the decoding layers; on the other hand, the prediction presented in b) is blurry and less detailed since it is not capable of maintaining most of the input's features.

TABLE 2. Comparison between multiple decoders. For a fair comparison the ResNet18 was fixed as the encoder across every experiment. The MAC operation count was obtained for a 4x192x1120 input (CxHxW) and doesn't take the encoder into account.

ResNet18-Decoder	RMSE (m)	SIlog ($\log(m)$ %)	δ_1 (%)	#Params (M)	MACS (G)	Time (ms)
BLConv5 [31]	3.215	5.24	94.1	17.46	13.78	**9.9**
DeConv5 [31]	3.438	5.19	93.8	17.46	13.81	11.6
UpProj [44]	**3.181**	**5.11**	**94.4**	38.07	120.11	41.5
UpConv [44]	3.225	5.04	94.2	17.46	55.10	22.5
Lower is Better: RMSE, SIlog, Inference Time						
Higher is Better: δ_1						

TABLE 3. The benefits of skip additive connections between encoder and decoder.

ResNet18 - BLConv5	RMSE (m)	REL (m)	SIlog ($\log(m)$ %)	δ_1 (%)	Time (ms)
Standard	3.184	0.069	5.41	94.1	**5.8**
W/ Skip Add.	**3.118**	**0.064**	**4.89**	**94.5**	6.4
Lower is Better: RMSE, REL, SIlog, Inference Time					
Higher is Better: δ_1					

TABLE 4. Comparison between different loss functions.

ResNet18-BLConv5	RMSE (m)	REL (m)	SIlog ($\log(m)$%)	δ_1 (%)
L1	3.125	**0.062**	4.90	94.4
L2	3.149	0.073	5.87	93.9
Huber	**3.118**	0.064	**4.89**	**94.5**
Lower is Better: RMSE, REL, SIlog				
Higher is Better: δ_1				

As stated in section III-A, depthwise separable convolutions lead to lower computational complexity. The use of these layers on the decoder of choice, the BLConv5, lowers the total MAC operations to 0.755G, slightly improves the RMSE (see table 1, where the ResNet18 is used along the BLConv5 with depthwise convolutions) and leads to a 4.1 ms gain on inference time.

3) IMPACT OF THE SKIP CONNECTIONS

When the encoded features reach the decoder, these have lost a lot of the original input's detail. The use of skip additive connections between encoder and decoder leads to a noticeable increase in the quality and detail of the prediction.

To experiment the impact of these connections, the ResNet18 is used as the encoder, the BLConv5 (with depthwise convolutions) as decoder and four additive connections between both were implemented. By looking at table 3, one can conclude that the network quantitatively benefits from these connections. The RMSE slightly improves to 3.118 m, while the δ_1 increases to 94.5%. Metrically, the highest impact is reflected on the ability of the network in capturing the global scale of the scene, achieving a SIlog of 4.89%. However, the additional operations have the downside of increasing the inference time to 6.4 ms.

When it comes to the higher level of detail in the prediction, visible differences can be seen in the highlighted areas of Fig.4. Comparing both Fig.4b) and Fig.4c), the latter is capable of retaining features extracted from both the RGB and magnitude of motion inputs, by making use of the skip connections. On the contrary, the depth map inferred by a network that does not connect both the encoder and decoder structures (Fig.4b) produces a blurry output which lacks most of the original features and resolution.

4) LOSS FUNCTION

Once the architecture had been decided, the next logical step was to optimize its learning process. To study the most suitable loss function for the problem at hand, several models were trained within the same conditions (refer to section IV for implementation details), varying the loss function.

The l_2 function is more sensitive to outliers, due to having a squared term, therefore larger distances are further penalized. This fact leads to a worse performance when compared to the remaining loss functions presented in table 4. Even if the RMSE is very similar across the table, the models optimized through an l_2 function still perform the worst in terms of SIlog (5.87%) and δ_1 (93.9%).

On the other hand, the l_1 and Huber loss functions perform rather similarly, however, the former tends to over-smooth boundaries [45]. The Huber loss behaves as a hybrid between the l_1 and l_2 functions, leading to the best performing model, both metrically (with an RMSE of 3.118 m) and quantitatively (achieving a SIlog of 4.89% and a δ_1 of 94.5%).

TABLE 5. State-of-the-Art comparison on the KITTI dataset. All methods are evaluated on the test split by Eigen *et al.* [9].

Method		Configuration (Learning, Cap)		Modality	RMSE (m)	REL (m)	δ_1 (%)	δ_2 (%)	δ_3 (%)
Saxena *et al.* (2009)	[11]	S	0-80m	RGB	8.734	0.280	60.1	82.0	92.6
Eigen *et al.* (2014)	[9]	S	0-80m	RGB	7.156	0.190	69.2	89.9	96.7
Liu *et al.* (2016)	[12]	S	0-80m	RGB	7.046	0.217	65.6	88.1	95.8
*Godard *et al.* (2017)	[16]	U	0-80m	RGB	4.935	0.114	86.1	94.9	97.6
Kuznietsov *et al.* (2017)	[13]	SeS	0-80m	RGB	4.621	0.113	86.2	96.0	98.6
Gan *et al.* (2018)	[46]	S	0-80m	RGB	3.933	0.098	89.0	96.4	98.5
Fu *et al.* (2018)	[1]	S	0-80m	RGB	**2.727**	0.072	93.2	98.4	99.4
Yin *et al.* (2019)	[14]	S	0-80m	RGB	3.258	0.072	93.8	99.0	99.8
Lee *et al.* (2019)	[2]	S	0-80m	RGB	2.756	**0.059**	**95.6**	**99.3**	**99.8**
Johnston *et al.* (2020)	[15]	SS	0-80m	RGB	4.699	0.106	88.9	96.2	98.2
Mathew *et al.* (2020)	[18]	U	0-80m	RGB	4.790	0.089	90.4	96.9	**98.8**
Liu *et al.* (2020)	[35]	U	0-80m	RGB	5.264	0.141	82.5	94.1	97.6
**Jiang *et al.* (2018)	[26]	SS	0-80m	RGB + Flow	4.903	0.125	84.0	95.1	98.3
Ours (NEON)		S	0-80m	RGB + Mag	**3.118**	**0.064**	**94.5**	**98.4**	**99.4**
Garg *et al.* (2016)	[17]	U	0-50m	RGB	5.104	0.169	74.0	90.4	96.2
*Godard *et al.* (2017)	[16]	U	0-50m	RGB	3.729	0.108	87.3	95.4	97.9
Kuznietsov *et al.* (2017)	[13]	SeS	0-50m	RGB	3.518	0.108	87.5	96.4	98.8
Fu *et al.* (2018)	[1]	S	0-50m	RGB	2.271	0.071	93.6	98.5	99.5
Lee *et al.* (2019)	[2]	S	0-50m	RGB	**1.925**	**0.056**	**96.4**	**99.4**	**99.9**
Liu *et al.* (2020)	[35]	U	0-50m	RGB	4.067	0.135	83.8	94.7	97.8
Ours (NEON)		S	0-50m	RGB + Mag	**2.243**	**0.059**	**95.1**	**98.6**	**99.5**

Lower is Better: RMSE, REL, SIlog
Higher is Better: δ_1, δ_2, δ_3

* Pretrained on the CityScapes Dataset [47].
** The flow is used to infer a relative depth map, which is used for pretraining the network, it doesn't serve directly as an input.

S = Supervised; U = Unsupervised; SeS = Semi-Supervised; SS = Self-Supervised

B. ABLATION STUDY: THE IMPACT OF EXPLOITING MOTION PERCEPTION

This section aims at studying the impact of different input modalities on the performance of the network. The first experiment consists in removing the magnitude of motion as input, training and evaluating the network within a monocular depth estimation context. The obtained results, see table 6, show that the RMSE drastically increases to 4.930 m. Similarly, the SIlog percentage also increases (8.21%), indicating an overall worse performance.

When the network receives the magnitude of motion as the sole input, i.e. with no support from the texture provided by an RGB image, it still achieves an RMSE of 3.608 m and an average of 90.5% of correctly predicted pixels.

From these experiments one is able to deduce that the network gives more weight to the features computed from the magnitude of motion than to the ones extracted from the texture. However, the most accurate predictions (both quantitatively and qualitatively) are obtained when both modalities are combined. This ablation study not only validates the usefulness of introducing the magnitude of pixel motion as an input to the network, but also proves that both inputs provide valuable information, that leads to an accurate and detailed depth map.

TABLE 6. The impact of different modalities on the performance of the network.

Modality	RMSE (m)	REL (m)	SIlog (log(m)%)	δ_1 (%)
RGB	4.930	0.132	8.21	82.0
Mag. Motion	3.608	0.155	6.55	90.5
RGB + Mag.	**3.118**	**0.064**	**4.89**	**94.5**

Lower is Better: RMSE, REL, SIlog
Higher is Better: δ_1

C. COMPARING NEON TO THE STATE-OF-THE-ART

The Eigen split of the KITTI dataset [9] has been widely used as a benchmark for depth prediction tasks. To compare the results obtained from the proposed approach to current state-of-the-art, the most relevant works to date have been compiled in table 5. The presented results are gathered from the corresponding papers, with the exception of Saxena *et al.* [11] which is obtained from Eigen's comparison in [9].

There have been considerable improvements in the state-of-the-art since the first pioneer works have been published. Both supervised and unsupervised techniques now present themselves as viable options to infer accurate and detailed depth maps, however, few of these are actually usable in real scenarios where resources are constrained.

Jiang *et al.* [26] make use of optical flow information to estimate relative depth, which is later used to pretrain the network. This is the most comparable approach to the one presented in this article however, the NEON network is trained to directly exploit the magnitude of movement in each pixel and combine it with features extracted from an RGB image. The

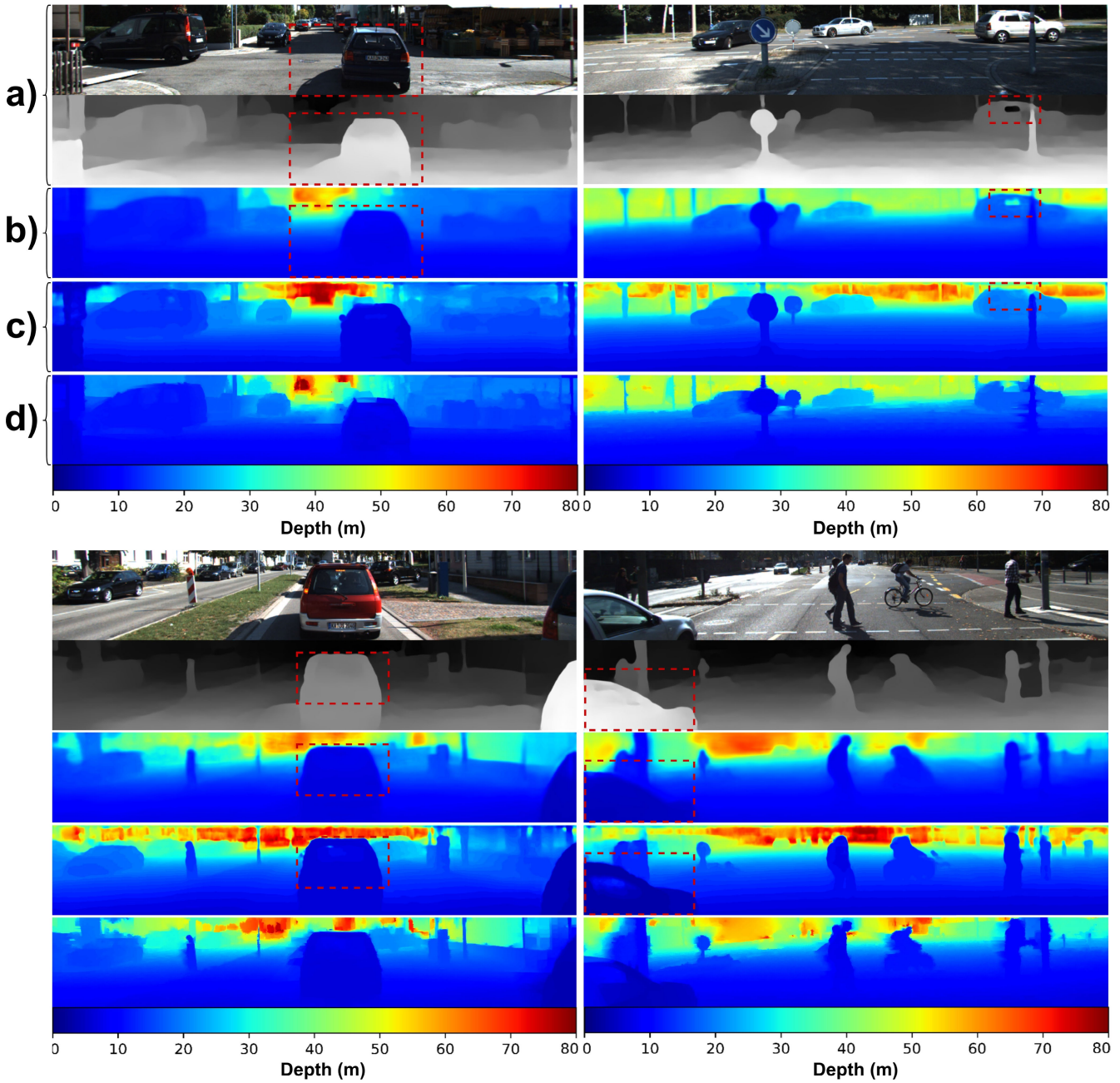


FIGURE 5. Visual comparison between the NEON and BTS [2] networks. A) NEON's input data, an RGB image and the magnitude of motion, respectively; B) NEON's output prediction; C) BTS' output prediction; D) Infilled ground truth. The proposed NEON network achieves an average RMSE of 3.118 m, being marginally worse than the BTS network. In terms of efficiency, NEON is 12 times faster at inferring a prediction than the BTS network.

results of both networks can only be compared within a global picture, with NEON achieving a better performance across all metrics, since Jiang's work is self-supervised and it cannot be directly compared to NEON's supervised methodology.

The current state-of-the-art performance is achieved by the BTS network proposed by Lee *et al.* [2]. The core of this network is based on a ResNext101 [48], which has 112.8M learnable parameters, **10 times** more than NEON's encoder, the ResNet18. This huge complexity difference reflects itself, not only in the quality of the predictions, but also in inference times. Fig.5 presents a visual comparison for the output predictions of both networks, NEON (row b) and BTS (row c). The latter does infer a more visually pleasing depth map, less blurry and with sharper edges, achieving an average RMSE of 2.756 m and 95.6% of correctly predicted pixels. On the other hand, NEON, achieves a lower δ_1 value of 94.5%; while also being marginally worse metrically (with a difference of 0.362 m), averaging an RMSE of 3.118 m.

In terms of efficiency, NEON is capable of making predictions approximately **12 times** faster than BTS. For the sake

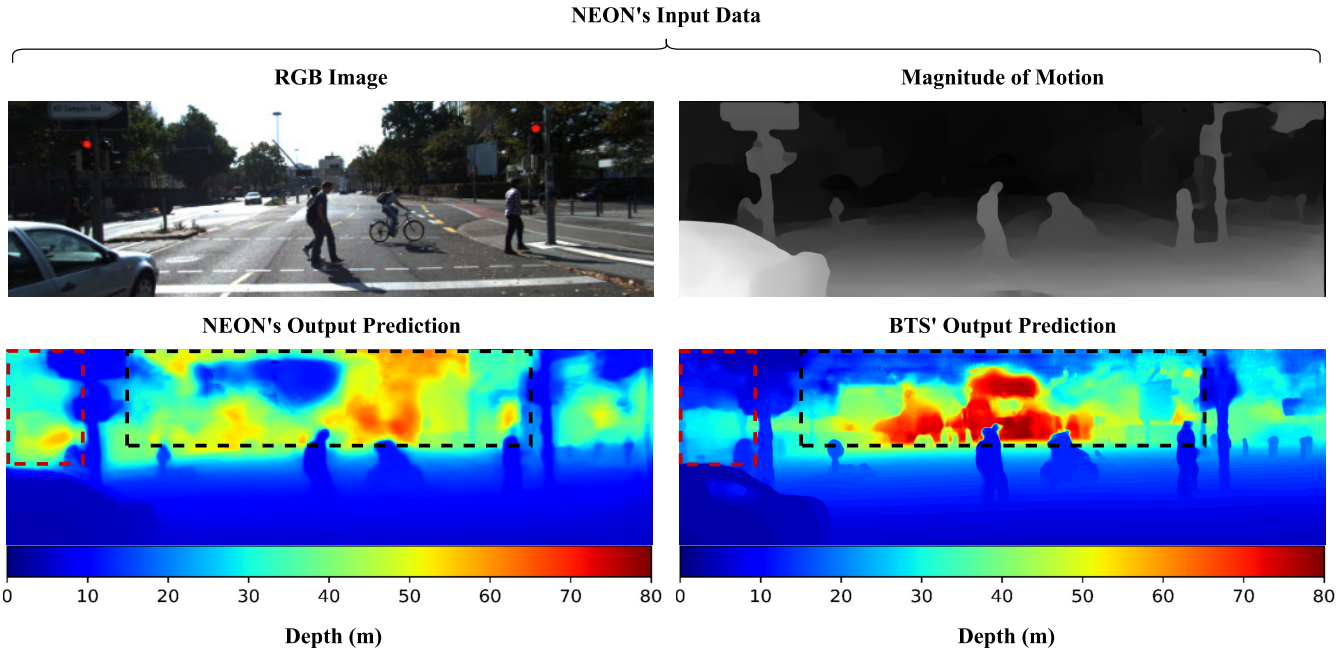


FIGURE 6. Exploring NEON's behavior in unsupervised regions. By referring to the highlighted regions, the prediction estimated by the BTS network indicates that the sky region is rather close to the observer. On the other hand, NEON depicts a much more logical depth map on the unsupervised regions. The reason behind this can be attributed to the fundamental learning task, where NEON not only learns from texture-based data, but also from magnitude of pixel motion.

of a fair comparison, both networks were benchmarked on the same hardware (refer to section IV for specifications), and the input size of the BTS network has been adjusted to 192 × 1120, matching NEON's input size. While NEON takes just 6.4 ms to infer a depth map, the BTS network takes 76.69 ms. Unlike previous state-of-the-art approaches, the methodology presented in this article is capable of maintaining high levels of accuracy and detail within its depth maps, while significantly reducing inference times.

Transparent surfaces, e.g. a car's glass window, are a classic problem in depth prediction tasks - in addition to being see through, these are also not detected by IR-based technologies. Since the BTS network is fully dependent on visual features such as texture and shadows, it is prone to making errors when it comes to these materials. The proposed NEON network has a better response to these scenarios, since it is not limited to a single type of input data. A clear example of the described behavior can be seen on the highlighted regions, in the bottom row, of Fig.5. On the other hand, since the magnitude of motion, that serves as additional input, is obtained from an optical flow field, it is also susceptible to erroneous situations: lack of texture, shadows and light flares (refer to the top row of Fig.5) are classical conditions that can misguide the optical flow algorithm [8].

A final comparison can be drawn between NEON and BTS. The lack of a complete ground-truth is fruit of the LiDAR's projection onto the 2D space, as well as the natural sparseness of the sensor. To avoid this issue, NEON's input is cropped so that the portions of the input with no correspondent ground-truth are not used. On the other hand, the BTS

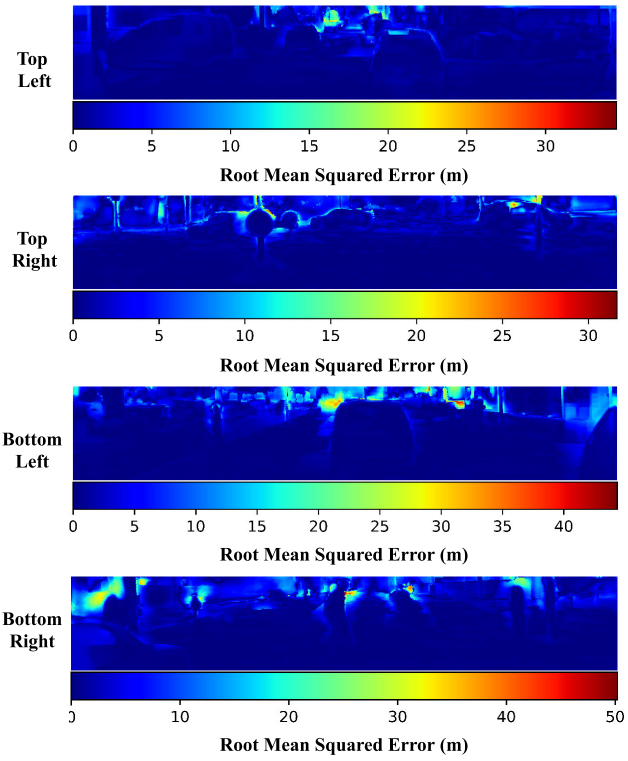


FIGURE 7. The RMSE maps relative to NEON's depth map predictions for the scenarios presented in Fig.5.

network is trained with the original sized input. This fact leads to the training of several unsupervised regions, that translate into non logical portions of the output depth map. To explore

NEON's behavior on the same unsupervised regions, the original data (i.e., with no crops) was fed to the network. Fig.6 depicts the results of this experiment. The regions highlighted in this figure correspond mostly to the sky. The depth map inferred by the BTS network predicts this area as being very close to the observer, which is obviously not correct. On the other hand, the output estimated by NEON depicts a much more logical depth map. Thus, shaping the learning task to not only learn how to predict depth from texture-based features, but also from the magnitude of pixel movement leads to models with better generalization capabilities, that produce logical predictions even in unsupervised regions.

Larger distances tend to be increasingly harder to estimate, regardless of the applied technique. By observing the RMSE maps presented in Fig.7, the majority of NEON's prediction error is present over long distances. This fact can be further explored by looking at the results in table 5, when the network is capped at 50 m, the average RMSE decreases by approximately 28%.

V. CONCLUSION

This article proposes a lightweight convolutional neural network - NEON - that exploits motion perception, taking advantage of the parallax effect to infer about the apparent motion of a scene. This relative sense of depth is combined with texture-based information provided by an RGB image. Ablation studies conducted to study the impact of the different input modalities prove that the features captured from this additional input have great weight on the final prediction, vastly improving the network's performance.

Research is vast in relevant works that resort to complex encoder structures such as the ResNext101 ($\sim$ 113M parameters) or VGG19 ($\sim$ 144M parameters). The proposed encoder-decoder architecture makes use of 29M learnable parameters in total, and approximately 22G MAC operations, being capable of outputting predictions in 6.4 ms ($\sim$ 156 Hz). Contrary to approaches that sacrifice accuracy for low-latency, NEON achieves near state-of-the-art accuracy on the KITTI dataset, with an average RMSE of 3.118 m and 94.5% of correctly predicted pixels. NEON provides the best accuracy-complexity trade-off, performing approximately 12 times faster than the current State-of-the-Art network, while having minimal cost on performance.

The methodology presented in this article can thus be integrated in time-constrained tasks that require fast and accurate 3D information to function, such as robotic grasping or even autonomous driving. Additionally, NEON's generalization capabilities allow the network to provide relevant information within completely distinct contexts. A repository containing the code basis for this work, as well as instructions on how to reproduce its results, can be found in https://github.com/pedronunoleite/NEON.

REFERENCES

[1] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2002–2011.

[2] J. Han Lee, M.-K. Han, D. W. Ko, and I. H. Suh, "From big to small: Multi-scale local planar guidance for monocular depth estimation," 2019, *arXiv:1907.10326*. [Online]. Available: http://arxiv.org/abs/1907.10326

[3] D. Morrison, P. Corke, and J. Leitner, "Learning robust, real-time, reactive robotic grasping," *Int. J. Robot. Res.*, vol. 39, nos. 2–3, pp. 183–201, Mar. 2020.

[4] D. F. Campos, A. Matos, and A. M. Pinto, "An adaptive velocity obstacle avoidance algorithm for autonomous surface vehicles," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 8089–8096.

[5] A. M. Pinto and A. C. Matos, "MARESye: A hybrid imaging system for underwater robotic applications," *Inf. Fusion*, vol. 55, pp. 16–29, Mar. 2020.

[6] Y. Wang, W.-L. Chao, D. Garg, B. Hariharan, M. Campbell, and K. Q. Weinberger, "Pseudo-LiDAR from visual depth estimation: Bridging the gap in 3D object detection for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 8437–8445.

[7] A. M. Pinto, A. P. Moreira, M. V. Correia, and P. G. Costa, "A flow-based motion perception technique for an autonomous robot system," *J. Intell. Robotic Syst.*, vol. 75, nos. 3–4, pp. 475–492, Sep. 2014.

[8] A. M. Pinto, "Visual motion analysis based on a robotic moving system," Ph.D. dissertation, Dept. Eng., Univ. Porto, Porto, Portugal, 2014. [Online]. Available: https://hdl.handle.net/10216/73552

[9] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 3, Jan. 2014, pp. 2366–2374.

[10] A. Saxena, S. H. Chung, and A. Y. Ng, "Learning depth from single monocular images," in *Proc. Adv. Neural Inf. Process. Syst.*, 2005, pp. 1161–1168.

[11] A. Saxena, M. Sun, and A. Y. Ng, "Make3D: Learning 3D scene structure from a single still image," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 5, pp. 824–840, May 2009.

[12] F. Liu, C. Shen, G. Lin, and I. Reid, "Learning depth from single monocular images using deep convolutional neural fields," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 10, pp. 2024–2039, Oct. 2016.

[13] Y. Kuznietsov, J. Stückler, and B. Leibe, "Semi-supervised deep learning for monocular depth map prediction," in *Proc. 30th IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2215–2223.

[14] W. Yin, Y. Liu, C. Shen, and Y. Yan, "Enforcing geometric constraints of virtual normal for depth prediction," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 5683–5692.

[15] A. Johnston and G. Carneiro, "Self-supervised monocular trained depth estimation using self-attention and discrete disparity volume," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 4755–4764.

[16] C. Godard, O. M. Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *Proc. 30th IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6602–6611.

[17] R. Garg, B. V. Kumar, G. Carneiro, and I. Reid, "Unsupervised CNN for single view depth estimation: Geometry to the rescue," in *Proc. 14th Eur. Conf. Comput. Vis. (ECCV)*, in Lecture Notes in Computer Science, 2016, pp. 740–756.

[18] A. Mathew and J. Mathew, "Monocular depth estimation with SPN loss," *Image Vis. Comput.*, vol. 100, Aug. 2020, Art. no. 103934.

[19] F. Y. Shih and C. Prathuri, "Depth cues from optical flow," in *Proc. SPIE, 8th Intell. Robots Comput. Vis., Algorithms Techn.*, D. P. Casasent, Ed., vol. 1192, Mar. 1990, pp. 770–777.

[20] B. Shahraray and M. K. Brown, "Robust depth estimation from optical flow," in *Proc. 2nd Int. Conf. Comput. Vis.*, 1988, pp. 641–650.

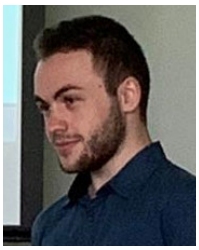
[21] C. Chang and S. Chatterjee, "Depth from stereo flow," in *Proc. IEEE Conf. Int. Conf. Syst., Man Cybern.*, vol. 2, Nov. 1989, pp. 586–591.

[22] S. Pathak, A. Moro, H. Fujii, A. Yamashita, and H. Asama, "Virtual reality with motion parallax by dense optical flow-based depth generation from two spherical images," in *Proc. IEEE/SICE Int. Symp. Syst. Integr. (SII)*, Dec. 2017, pp. 887–892.

[23] M. Mancini, G. Costante, P. Valigi, and T. A. Ciarfuglia, "Fast robust monocular depth estimation for obstacle detection with fully convolutional networks," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 4296–4303.

[24] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox, "DeMoN: Depth and motion network for learning monocular stereo," in *Proc. 30th IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5622–5631.

[25] K. Wang and S. Shen, "Flow-motion and depth network for monocular stereo and beyond," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 3307–3314, Apr. 2020.
[26] H. Jiang, G. Larsson, M. Maire, G. Shakhnarovich, and E. Learned-Miller, *Self-Supervised Relative Depth Learning for Urban Scene Understanding* (Lecture Notes in Computer Science: Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 11215. Cham, Switzerland: Springer, 2018, pp. 20–37.
[27] J. Chen, X. Yang, Q. Jia, and C. Liao, "DENAO: Monocular depth estimation network with auxiliary optical flow," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Feb. 28, 2020, doi: 10.1109/TPAMI.2020.2977021.
[28] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*. [Online]. Available: http://arxiv.org/abs/1704.04861
[29] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.
[30] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. 36th Int. Conf. Mach. Learn. (ICML)*, Jun. 2019, pp. 10691–10700.
[31] D. Wofk, F. Ma, T.-J. Yang, S. Karaman, and V. Sze, "FastDepth: Fast monocular depth estimation on embedded systems," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 6101–6108.
[32] V. Nekrasov, T. Dharmasiri, A. Spek, T. Drummond, C. Shen, and I. Reid, "Real-time joint semantic segmentation and depth estimation using asymmetric annotations," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 7101–7107.
[33] G. Lin, A. Milan, C. Shen, and I. Reid, "RefineNet: Multi-path refinement networks for high-resolution semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5168–5177.
[34] M. Poggi, F. Aleotti, F. Tosi, and S. Mattoccia, "Towards real-time unsupervised monocular depth estimation on CPU," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 5848–5854.
[35] J. Liu, Q. Li, R. Cao, W. Tang, and G. Qiu, "MiniNet: An extremely lightweight convolutional neural network for real-time unsupervised monocular depth estimation," *ISPRS J. Photogramm. Remote Sens.*, vol. 166, pp. 255–267, Aug. 2020.
[36] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[37] L. Sifre, "Rigid-motion scattering for image classification," Ph.D. dissertation, Dept. Appl. Math., Ecole Polytechnique, Palaiseau, France, 2014. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.672.7091
[38] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, Sep. 2013.
[39] A. Levin, D. Lischinski, and Y. Weiss, "Colorization using optimization," in *Proc. ACM SIGGRAPH Papers (SIGGRAPH)*, 2004, pp. 689–694.
[40] A. M. Pinto, M. V. Correia, A. P. Moreira, and P. G. Costa, "Unsupervised flow-based motion analysis for an autonomous moving system," *Image Vis. Comput.*, vol. 32, nos. 6–7, pp. 391–404, Jun. 2014.
[41] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "FlowNet 2.0: Evolution of optical flow estimation with deep networks," in *Proc. 30th IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1647–1655.
[42] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–14.
[43] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2261–2269.
[44] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *Proc. 4th Int. Conf. 3D Vis. (3DV)*, Oct. 2016, pp. 239–248.
[45] P. N. Leite, R. J. Silva, D. F. Campos, and A. M. Pinto, *Dense Disparity Maps From RGD and Sparse Depth Information Using Deep Regression Models* (Lecture Notes in Computer Science), vol. 12131. Cham, Switzerland: Springer, Jul. 2020.
[46] Y. Gan, X. Xu, W. Sun, and L. Lin, *Monocular Depth Estimation with Affinity, Vertical Pooling, and Label Enhancement* (Lecture Notes in Computer Science: Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 11207. Cham, Switzerland: Springer, 2018, pp. 232–247.
[47] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.
[48] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. 30th IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5987–5995.



PEDRO NUNO LEITE was born in Porto, Portugal, in 1996. He received the M.Sc. degree in electrical and computer engineering from the Faculty of Engineering, University of Porto (FEUP), in 2019, where he is currently pursuing the Ph.D. degree. He is also affiliated with INESC TEC—CRAS, working as an Assistant Researcher. His current research interests include underwater robotics and perception techniques, computer vision, and deep learning.

ANDRY MAYKOL PINTO received the Ph.D. degree in electrical and computer engineering from the Faculty of Engineering, University of Porto, Portugal, in 2014. He is currently an Assistant Professor with the Faculty of Engineering, University of Porto, and a Senior Researcher with the Centre for Robotics and Autonomous Systems, INESC TEC. He is also a principal investigator of national and international research and development projects related to robotic-based operation and maintenance activities for offshore infrastructures. His main research interests include multi-domain perception, underwater imaging, artificial intelligence, and mobile robotics.