Article

An Adaptive Refinement Scheme for Depth Estimation Networks

Amin Alizadeh Naeini, Mohammad Moein Sheikholeslami and Gunho Sohn*

Department of Earth and Space Science and Engineering, Lassonde School of Engineering, York University, 4700 Keele Street, Toronto, Ontario M3J1P3, Canada
* Correspondence: gsohn@yorku.ca

Abstract: Deep learning, specifically the supervised approach, has proved to be a breakthrough in depth prediction. However, the generalization ability of deep networks is still limited, and they cannot maintain a satisfactory performance on some inputs. Addressing a similar problem in the segmentation field, a scheme (f-BRS) has been proposed to refine predictions in the inference time. f-BRS adapts an intermediate activation function to each input by using user clicks as sparse labels. Given the similarity between user clicks and sparse depth maps, this paper aims at extending the application of f-BRS to depth prediction. Our experiments show that f-BRS, fused with a depth estimation baseline, is trapped in local optima, and fails to improve the network predictions. To resolve that, we propose a double-stage adaptive refinement scheme (DARS). In the first stage, a Delaunay-based correction module significantly improves the depth generated by a baseline network. In the second stage, a particle swarm optimizer (PSO) delineates the estimation through fine-tuning f-BRS parameters—that is, scales and biases. DARS is evaluated on an outdoor benchmark, KITTI, and an indoor benchmark, NYUv2 while for both the network is pre-trained on KITTI. The proposed scheme outperformed rival methods on both datasets.

Keywords: depth estimation; optimization; deep learning

1. Introduction

Dense depth maps play a crucial role in a variety of applications, such as simultaneous localization and mapping (SLAM) [1], visual odometry [2], and object detection [3]. With the advent of deep learning (DL) and its ever-growing success in most fields, DL methods have also been utilized for generating dense depth (DD) maps and have demonstrated a prominent improvement in this field.

Depending on the primary input, DL-based depth generation methods can be categorized into depth completion and estimation methods. Depth completion methods try to fill the gaps present in input sparse depth (SD) maps [4,5], whereas depth estimation ones attempt to estimate depth for each pixel of an input image [6–10]. Although the results provided by depth completion methods [4,11] are usually more accurate than those from depth estimation ones, they need to be supplied by a remarkably large number of DD maps as targets in the training stage. This is while collecting such data in a real-world application is an expensive and time-consuming task [8,12].

In parallel, depth estimation methods take either SD [13] or DD maps [14,15] as the target during training. Between these two depth estimation approaches, using SD maps usually leads to less accurate results but is more viable than using DD ones. Because SD-based methods only need SD maps which can be provided using a LiDAR sensor and without any need for post-processing or labeling effort. Considering the above issues, depth estimation methods which use sparse depth maps are preferred, especially for most real-world cases in which access to large-enough accurate DD maps is difficult or even impossible.

Similar to all DL methods, DL-based depth estimation methods can be categorized into supervised and unsupervised approaches. Supervised ones are more popular because of their superior performance in depth estimation [14,16]. However, supervised approaches, also named guided depth estimation, suffer from the generalization problem. In other terms, DL models trained on an arbitrary dataset are not able to preserve their satisfying performance neither on unseen data samples nor even on hard seen ones. This problem is more severe in applications such as SLAM and autonomous vehicles, where the test environment is under a constant change. Hence the high variety in input samples leads to accuracy degradation [17].

In the segmentation field, a similar problem, i.e., limited generalization ability, has been alleviated by backpropagating refinement scheme (BRS) [18]. This method has been proposed to optimize an input to a segmentation network based on user clicks. The method performs this process via backpropagating through the whole network; thus, it suffers from the computational burden. To speed up the process, feature BRS (f-BRS) [19] has been proposed to only backpropagate through several last layers and optimizes activation responses of an intermediate layer using some introduced parameters. Optimizing those parameters during inference can be viewed as an adaptive approach because an intermediate activation function and indeed the output is adapted to input data. In other words, f-BRS converts the baseline network to a functionally adaptive method, in which the shape of activation function changes in the inference time [20,21].

As user clicks and SD maps are both sparse labels, it seems that the application of the above-mentioned scheme, f-BRS [19], can be generalized to depth estimation. Accordingly, we present an inference-time functionally adaptive refinement scheme for depth estimation networks (see Figure 1). For this purpose, f-BRS [19] is used which can be injected into any DL baseline and adapts the model to hard and unseen environments or different datasets. Nevertheless, f-BRS suffers from two fundamental issues which prevent it from being applicable in depth estimation. The first problem is associated with its nature which cannot manage highly inaccurate products (here, the depth predicted by the network). Since f-BRS has been originally proposed for interactive segmentation [19], where there is often no need for a considerable modification. To resolve this, a sliced Delaunay correction (SDC) module is designed to carry out a correction using SD maps and provide an appropriate initial value for the optimizer. The second problem is that f-BRS is trapped in local optima due to its local optimizer. To address this, f-BRS is equipped with particle swarm optimization (PSO) as a global optimizer [22]. For simplicity, this novel generalization of f-BRS which is applicable to depth estimation is named as double-stage adaptive refinement scheme (DARS), where the first stage is done through SDC and the second stage is conducted by optimizing the f-BRS parameters, i.e., adapting the activation maps.

Overall, our contributions can be summarized as:

- A novel double-stage adaptive refinement scheme for monocular depth estimation networks. The proposed scheme needs neither offline data gathering nor offline training, because it uses available pre-trained weights.
- Introduction of functional adaptation schemes in the field of depth generation, for the first time.
- A model-agnostic scheme which can be plugged into any baseline. In this paper, we selected Monodepth2 [23], as one of the most widely used baselines for depth estimation.

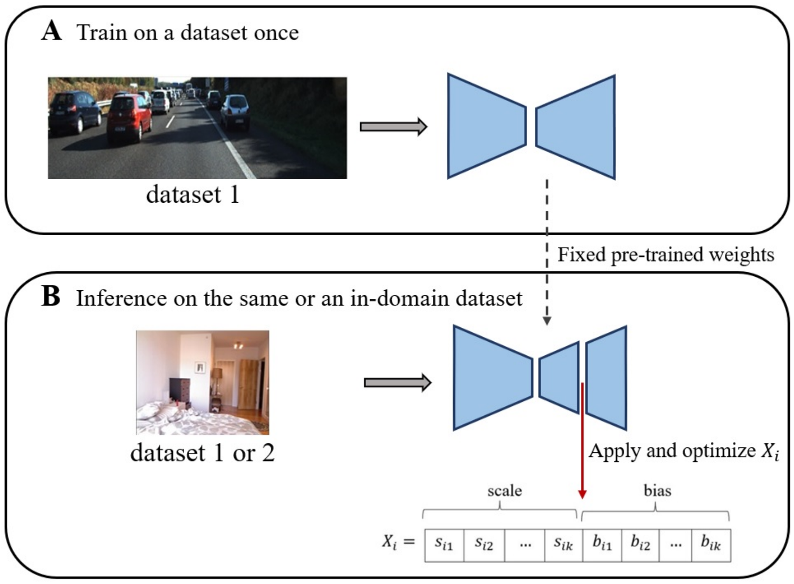

A Train on a dataset once
dataset 1
Fixed pre-trained weights
B Inference on the same or an in-domain dataset
dataset 1 or 2
Apply and optimize X_i
scale
bias
$X_i = s_{i1}\ s_{i2} \ldots s_{ik}\ b_{i1}\ b_{i2} \ldots b_{ik}$

Figure 1. Overall performance of the proposed scheme. Keeping the pre-trained weights fixed, channel-wise scales and biases are applied to an intermediate activation map and are optimized to improve the predictions for the input dataset. The input dataset can be identical to the training dataset or a new one.

2. Related Work

Here, we initially provide an overview for unsupervised and supervised depth estimation methods. In the last part, a brief review of functionally adaptive networks is brought.

2.1. Unsupervised Depth Estimation Methods

These methods use color consistency loss between stereo images [24], temporal ones [25], or a combination of both [23] to train a monocular depth estimation model. Many attempts have been made to rectify the self-supervision by new loss terms like left-right consistency [26], temporal depth consistency [27], or cross-task consistency [28–30]. Of these improvements, Monodepth2 has attracted substantial attention because of the different sets of techniques it has used for modification [23]. To the best of our knowledge, methods in this category have been presented for either outdoor environments, such as the above ones, or indoor environments as in [31]. Not being applicable for both indoor and outdoor datasets can be regarded as a drawback of these methods. Another problem of these methods is that they suffer from low accuracy.

2.2. Supervised Depth Estimation Methods

The inputs to these methods are only images and they use either DD maps or SD maps as targets. This group can be categorized into DD-based and SD-based methods. Of these two, DD-based ones need DD maps during their training. DD-based methods, like AdaBins [14] and BTS [15], learn based on the error between predicted depth maps and DD maps. The main disadvantage of these methods is their need to DD maps for training.

Unlike DD-based methods, SD-based ones use SD maps only. Training data are not an issue for these methods because current robots and mapping systems can capture both images and SD maps simultaneously. The distance between predictions and SD maps are used as loss functions [32–34]. These methods are also known as semi-supervised [35].

2.3. Functionally Adaptive Neural Networks

Neural networks are called adaptive when they can adapt themselves to unseen environments i.e., new inputs [36,37]. There are different techniques for designing adaptive networks, among which weight modification and functional adaptation can be mentioned. The former optimizes the network weights for new inputs while the latter modifies the slope and shape of the activation functions usually through a relatively few number of additional parameters [36]. Functional adaptation can be categorized under activation response optimization methods [38–41], in which the aim is to update activation responses while the network weights are fixed. The reason behind keeping the network weights fixed is to preserve the semantics learned by the network during the training process. On the other hand, one or several activation responses are modified to optimize the performance on inevitable unseen objects and scenes so that the network maintains its proficient performance in constantly-changing environments [19].

The adaptation process can happen in either the training stage [20] or the inference stage for some tasks like interactive segmentation or SLAM that some ground truth (even though sparse) is available on the fly [37]. In addition, the networks can adapt to a sequence of images or a single image. In single-image adaptation, core merit is to optimize the prediction for a specific image or even an object and the adaptation is discarded for the next image [37]. Thus, single-image adaptation can be beneficial specially when scenes are prone to varying significantly.

Inspired from the biological neurons, some investigations have been conducted on the adaptive activation functions such as PReLU, which shows that adaptation behaviour in such activation functions can improve the accuracy and generalization of neural networks [20]. In [19], some parameters are introduced to adapt the activation functions to user clicks during the inference of the interactive segmentation task. An adaptive instance normalization layer is proposed in [21] that enables the style transfer networks to adapt to arbitrary new styles with adding negligible computational cost.

3. Theoretical Background

In this section, some theoretical background needed for understanding the proposed scheme is provided.

3.1. Delaunay-based Interpolation

The first step of the interpolation is to conduct triangulation. Considering that there are many different triangulations for a given point set, we should obtain an optimal triangulation method avoiding poorly shaped triangles. Delaunay triangulation method has proved to be the most robust and widely-used triangulation approach. This method connects all the neighboring points in a Voronoi diagram to obtain a triangulation [42].

To find the value of any new point by interpolation, its corresponding triangle in which it lies should be identified. Suppose $P(x, y)$ is a new point that belongs to a triangle with vertices of $P_1(x_1, y_1)$, $P_2(x_2, y_2)$ and $P_3(x_3, y_3)$ with the values of z_1, z_2 and z_3, respectively. To linearly interpolate the value z of P, we should fit a plane (Equation (1)) to the vertices P_1, P_2 and P_3.

$$z = ax + by + c \tag{1}$$

By inserting the known points (x_1, y_1, z_1), (x_2, y_2, z_2) and (x_3, y_3, z_3) in Equation (1) and solving a linear system of equations, the unknown coefficients (a, b, c) of the plane are estimated. Finally, applying Equation (1) and having (a, b, c), the value z for any arbitrary point $P(x, y)$ is interpolated within the triangle (Figure 2).

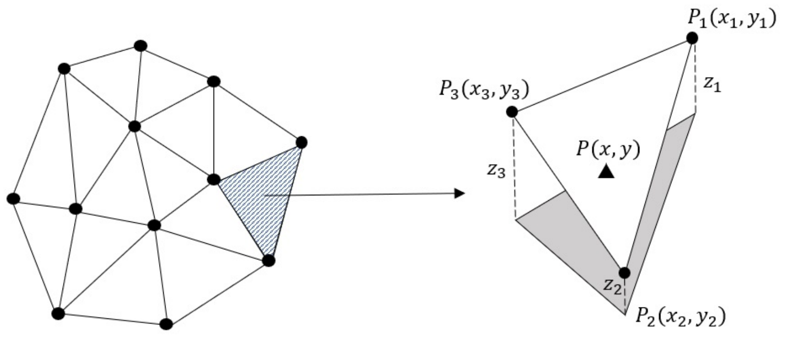


Figure 2. Delaunay-based interpolation on a set of points. First, a Delaunay triangulation is done on the points. Then a plane is fitted to each triangle, and finally the value for points on each of them is obtained based on the fitted plane.

3.2. PSO 154

PSO is a population-based stochastic optimization technique inspired by the social behavior of birds within a flock or fish schooling [22]. PSO has two main components which need to be specifically defined for each application. One component is the introduction of particles, and the other is an objective function for particle evaluation. 155–158

Each particle has the potential of solving the problem; this means they must contain all the arguments needed for the problem in question. 159–160

The velocity and position of each particle are calculated using Equation (2) and Equation (3), respectively [22]. Optimum values of unknown parameters are iteratively updated using the position equation, which is itself dependent on the velocity. 161–163

$$V_i(t+1) = wV_i(t) + c_1 r_1(t)[pbest_i(t) - X_i(t)] + c_2 r_2(t)[gbest_i(t) - X_i(t)] \tag{2}$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \tag{3}$$

In Equation (2), $V_i(t)$ is the velocity of a particle i at time t, and $pbest_i(t)$ and $gbest_i(t)$ are personal and global best positions found by the particle i and all the particles by the iteration t, respectively. The w parameter is an inertia weight scaling the previous time step velocity. Parameters c_1 and c_2 are two acceleration coefficients that scale the influence of $pbest_i(t)$ and $gbest_i(t)$, respectively. In addition, parameters r_1 and r_2 are random variables between 0 and 1 obtained from a uniform distribution. The next position of each particle $(X_i(t+1))$ can be calculated using Equation (3). 164–170

4. Proposed Method 171

Supervised depth estimation methods suffer from the generalization problem. In other words, they usually need to be retrained for achieving a proficient performance on an unseen dataset. To alleviate this, a double-stage adaptive refinement scheme (DARS) is proposed to equip pre-trained depth estimation networks with an inference-time optimization for improving the performance on both seen and unseen datasets. The proposed scheme (Figure 3) consists of several components including a deep baseline model, a correction module which applies the first stage of refinement, and an activation optimization as the second stage. The tasks and details of each, and the overall proposed scheme are brought in below. In the following subsections s and d superscripts respectively indicate that depth maps are sparse or dense. 172–181

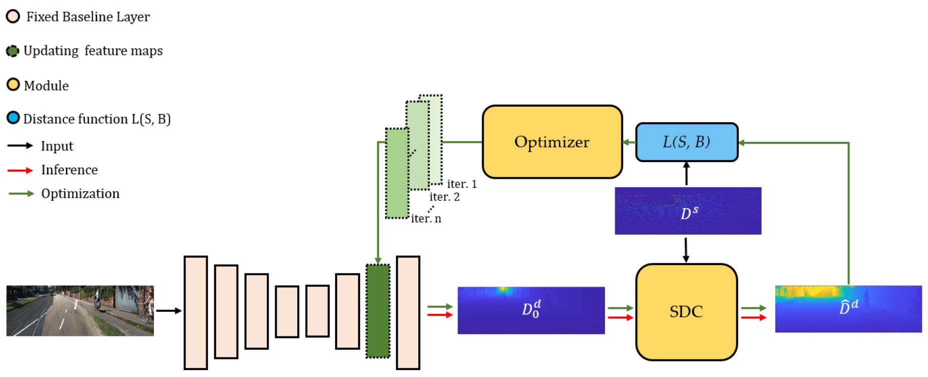


Figure 3. The proposed scheme.

4.1. Baseline 182

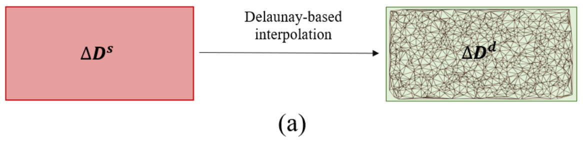
Given an input monocular RGB image $I \in \mathbb{R}^{w \times h \times 3}$, we rely on a depth estimation network $F : I \mapsto D_0^d$ to provide us with an initial depth map $D_0^d \in \mathbb{R}^{w \times h}$. The proposed scheme can utilize any monocular depth estimation network. In this study, Monodepth2 [23] has been selected as the baseline, as one of most widely used depth estimation networks. The baseline is pre-trained and the weights are kept fixed. 183–187

4.2. Correction 188

The depth map D_0^d predicted by the baseline lacks sufficient accuracy, especially for an unseen input. Thus, D_0^d is not a proper initial value for the optimization stage. As a solution, in the first stage of the proposed refinement scheme, a sliced Delaunay correction (SDC) $C : \mathbb{R}^{w \times h} \mapsto \mathbb{R}^{w \times h}$ is used to correct D_0^d, using the available sparse depth map D^s. In SDC, first a correction value $\delta d^s \in \Delta D^s$ for any available depth pixel $d^s \in D^s$ is calculated: 189–193

$$\delta d^s = d_0^s - d^s \tag{4}$$

where $d_0^s \in D_0^d$ are the pixels in D_0^d corresponding to the ones in D^s. Then the sparse correction map ΔD^s is divided into three overlapped slices (see Figure 4). Because neighboring pixels are intuitively assumed to share a similar error pattern, and slices can represent a simplistic segmentation based on the error pattern. 194–197



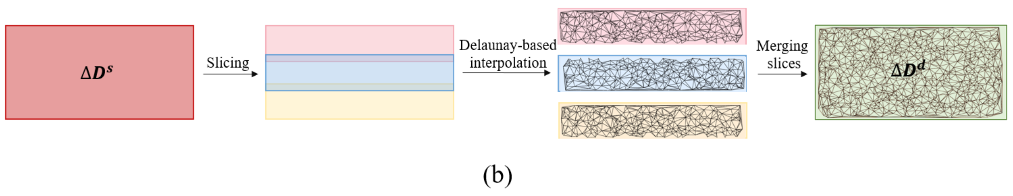


Figure 4. **(a)** Delaunay correction (DC), and **(b)** sliced Delaunay correction (SDC). In DC, the Delaunay-based interpolation is conducted on the whole sparse correction map. While in SDC, each SD map is first divided into three slices with overlap, then the correction value (ΔD^s) is interpolated using Delaunay-based interpolation in each of them, independently. In the overlapped areas, the average of the values coming from the two slices is taken.

In each slice, a Delaunay-based interpolation (see 3.1) $J : \mathbb{R}^2 \mapsto \mathbb{R}$ is utilized to estimate a dense correction map $\Delta D^d = J(\Delta D^s)$, given the sparse one ΔD^s. For the pixels 198–199 in overlapped areas (see Figure 4), the average of the values coming from two adjacent slices is considered as the final depth correction value. As a result of this stage, a corrected depth $\hat{D}^d = D_0^d + \Delta D^d$ is generated, yet with marginal errors.

4.3. Activation Optimization

Given the initial value from the first stage (correction), the core part of network adaptation is conducted in the second stage. The technique chosen for the network adaptation is to modify an intermediate set of activation outputs [36]. This is usually done by freezing the weights and optimizing some auxiliary parameters. This way, not only are the valuable learned semantics preserved, but also the network can adapt itself to inputs. Inspired from works like f-BRS [19] in interactive segmentation field, we apply channel-wise scale and bias parameters on intermediate features of the baseline network. The scales are initialized to ones and biases to zeros; they are then optimized based on a cost function. To describe the algorithm of the optimization module better, the overall scheme, i.e., from baseline to optimization module is explained, followed by some details about the optimizer.

4.3.1. Overall Scheme

Given an input RGB image $I \in \mathbb{R}^{w \times h \times 3}$, denote the intermediate feature set as $G(I) \in \mathbb{R}^{m \times n \times c}$ where $G : \mathbb{R}^{w \times h} \mapsto \mathbb{R}^{m \times n \times c}$ is the network body and m, n, and c are respectively width, height, and number of channels. The auxiliary parameters, scales $S \in \mathbb{R}^c$ and biases $B \in \mathbb{R}^c$ are applied on $G(I)$, and the depth $D_0^d = H(S \bigotimes G(I) \bigoplus B)$ is predicted, where $H : \mathbb{R}^{m \times n \times c} \mapsto \mathbb{R}^{w \times h}$ is the network head, and $\bigotimes$ and $\bigoplus$ represent channel-wise multiplication and addition. Afterwards, the correction module $C : \mathbb{R}^{w \times h} \mapsto \mathbb{R}^{w \times h}$ carries out the first refinement stage on D_0^d and returns $\hat{D}^d$:

$$\hat{D}^d = C(D_0^d, D^s) \tag{5}$$

The auxiliary parameters $X \in \mathbb{R}^{2c}$, i.e, channel-wise scales and biases, are learnable. Therefore, the following optimization problem can be formulated as:

$$L(\hat{D}^d(I, X + \Delta X), D^s) \to \min_{\Delta X}., \tag{6}$$

where ΔX is the corrections applied to the parameters and L is the cost function given to the optimizer.

4.3.2. Optimizer

The above optimization problem can be given to any type of optimizers. The default optimizer of f-BRS is limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) [43,44]. This optimizer , due to its local gradient-based nature, is trapped in local optima. To overcome this problem, L-BFGS is replaced with PSO [22]. PSO iteratively updates scale and bias parameters in each particle based on the below distance loss:

$$L = \sqrt{\frac{1}{T} \sum_{i=1}^{T} \left\| \log(\hat{d^s}) - \log(d^s) \right\|}, \tag{7}$$

where T is the total number of pixels with depth values in D^s.

5. Experiments

In this section, we first briefly describe the datasets used in the experiments. Secondly, the metrics are introduced and after that an ablation study is brought to show the effectiveness of each module. Finally, the results by the proposed scheme are compared with those of the state of the art.

5.1. Datasets

Two datasets are used in the experiments, KITTI [45] and NYUv2 [46]. KITTI is a well-known outdoor dataset, on which the baseline is trained. While NYUv2 is an indoor benchmark dataset and the adaptation performance of the scheme is highlighted through testing on that.

5.1.1. KITTI

KITTI dataset [45] consists of stereo RGB images and corresponding SD and DD maps of 61 outdoor scenes acquired by 3D mobile laser scanners. The RGB images have a resolution of 1241×376 pixels, while the corresponding SD maps are of very low density with lots of missing data. The dataset is divided into 23,488 train and 697 test images, according to [47]. For testing, 652 images associated with DD maps are selected from the test split. A sample data has been brought in Figure 5 for KITTI dataset.

5.1.2. NYUv2

NYUv2 dataset [46] contains 120,000 RGB and depth pairs having a size of 640×480 pixels acquired as video sequences using a Microsoft Kinect from 464 indoor scenes. The official train/test split contains 249 and 215 scenes, respectively. Given that NYUv2 does not contain any SD maps, SD maps with 80% sparsity have been randomly synthesized from DD maps for the experiments of the proposed method. Sample data for NYUv2 dataset, including the synthetic SD maps are illustrated in Figure 5.

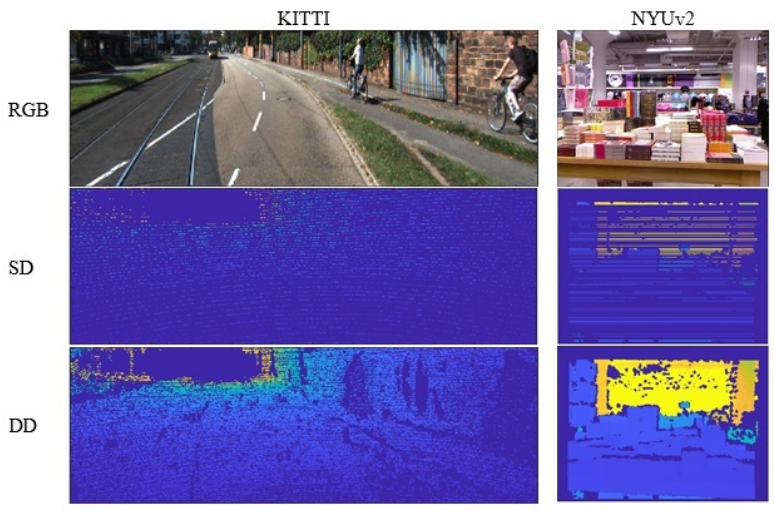


Figure 5. A sample of used datasets.

5.2. Assessment Criteria

Assessment criteria proposed by [47] include error and accuracy metrics. The error metrics are root mean square error (RMSE), logarithmic RMSE (RMSE$_{\text{log}}$), absolute relative error (Abs Rel), and square relative error (Sq Rel), whereas the accuracy rate metrics contain $\delta < 1.25^t$ where $t = 1, 2, 3$. These criteria are formulated as follows:

$$RMSE = \sqrt{\frac{1}{T}\sum_{i\in T} ||d_i - d_i^{gt}||^2}, \tag{8}$$

$$RMSE_{log} = \sqrt{\frac{1}{T}\sum_{i\in T} ||log(d_i) - log(d_i^{gt})||^2}, \tag{9}$$

$$AbsRel = \frac{1}{T}\sum_{i\in T} \frac{d_i - d_i^{gt}}{d_i^{gt}}, \tag{10}$$

$$SqRel = \frac{1}{T}\sum_{i\in T}\frac{||d_i - d_i^{gt}||^2}{d_i^{gt}}, and \tag{11}$$

$$accuracies = \% \text{ of } d_i \text{ subject to } max(\frac{d_i}{d_i^{gt}}, \frac{d_i^{gt}}{d_i}) = \delta \tag{12}$$

where d_i and d_i^{gt} are the predicted and target (ground truth) depth respectively at the pixel indexed by i, and T is the total number of pixels in all the evaluated images.

5.3. Network Architecture

As the proposed scheme is by design model agnostic, the network architecture is not the focus of this study. Thus, we used the standard monocular version of Monodepth2 [23] model with the input size of $640 \times 192 \times 3$.

5.4. Implementation Details

We have used monocular Monodepth2 pre-trained on KITTI as our baseline. The input images were resampled to 640×192 and then were fed to the network. The weights were fixed and network was run in inference mode. In SDC, the number of slices were 3 and the overlap between was set to 50%. Moreover, PSO paramters, i.e., c_1, c_2, number of particles, and number of iterations were respectively set to 0.5, 0.3, 10, and 30 in all the experiments. Also, all the implementations were conducted in PyTorch [48].

5.5. Ablation Studies

This ablation study aims to prove the effectiveness of different stages and modules in the proposed scheme. To do this, starting from the baseline, we have enabled the correction and optimization modules in several steps (see Table 1). First of all, the result of Monodepth2 [23] without any kind of post-processing is reported as our baseline. It means that the baseline results are without median scaling by target DD maps. As a result, they suffer from scale ambiguity and low accuracy. In addition, DC is introduced to show the efficacy of slicing in our proposed SDC as the correction module. The difference between SDC and DC is that, in the latter, Delaunay interpolation and correction are carried out on the entire depth maps instead of separately on each slice. For the sake of brevity, these two methods have just been surveyed for KITTI.

Table 1. Ablation Study on KITTI and NYUv2.

Dataset	Modules			Lower is better			Higher is better		
	Baseline	Correction	Optimizer	AbsRel	RMSE	RMSE$_{log}$	$\delta_{1.25}$	$\delta_{1.25^2}$	$\delta_{1.25^3}$
KITTI	Monodepth2	-	-	0.996	19.324	5.715	0.000	0.000	0.000
	Monodepth2	DC	-	0.864	16.888	3.149	0.183	0.330	0.447
	Monodepth2	SDC	-	0.046	1.676	0.091	0.976	0.991	0.995
	Monodepth2	SDC	L-BFGS	0.046	1.676	0.091	0.976	0.991	0.995
	Monodepth2	SDC	PSO	**0.024**	**1.440**	**0.071**	**0.985**	**0.993**	**0.996**
NYUv2	Monodepth2	SDC	-	0.018	0.766	0.747	0.972	0.974	0.975
	Monodepth2	SDC	L-BFGS	0.018	0.766	0.747	0.972	0.974	0.975
	Monodepth2	SDC	PSO	**0.017**	**0.109**	**0.044**	**0.993**	**0.996**	**0.999**

From Table 1, the worst results on KITTI in terms of all the metrics was recorded by the baseline, which was expected because of scale ambiguity. Using DC as the correction module improved the results by 13% in terms of RMSE, while SDC showed a significantly higher improvement over the baseline by 91%. This not only proves the contribution of the correction module but also indicates the effectiveness of the slicing process in SDC. Moreover, this observation supports the assumption that adjacent pixels in depth maps share a similar error pattern. First because adjacent pixels usually belong to same objects.

Second, the error in LiDAR sensor has a correlation with distance from sensor , and as a result pixels which are in an approximately equal distance to the sensor are likely to have close error magnitudes. From another perspective, the proposed slicing proved to be a simplistic segmentation based on the error pattern and was able to remarkably contribute to the correction stage.

According to Table 1, the results obtained when using L-BFGS as the optimizer are equal to ones without optimization on both KITTI and NYUv2 datasets. This means that L-BFGS could not improve the results because, unlike PSO, it does not have the capability for global search. In better words, it seems that it was trapped in local optima, i.e, the depth provided by SDC. Therefore, due to the identical performances and for the sake of conciseness, just one row is dedicated to both SDC and L-BFGS in Figure 6 and Figure 7.

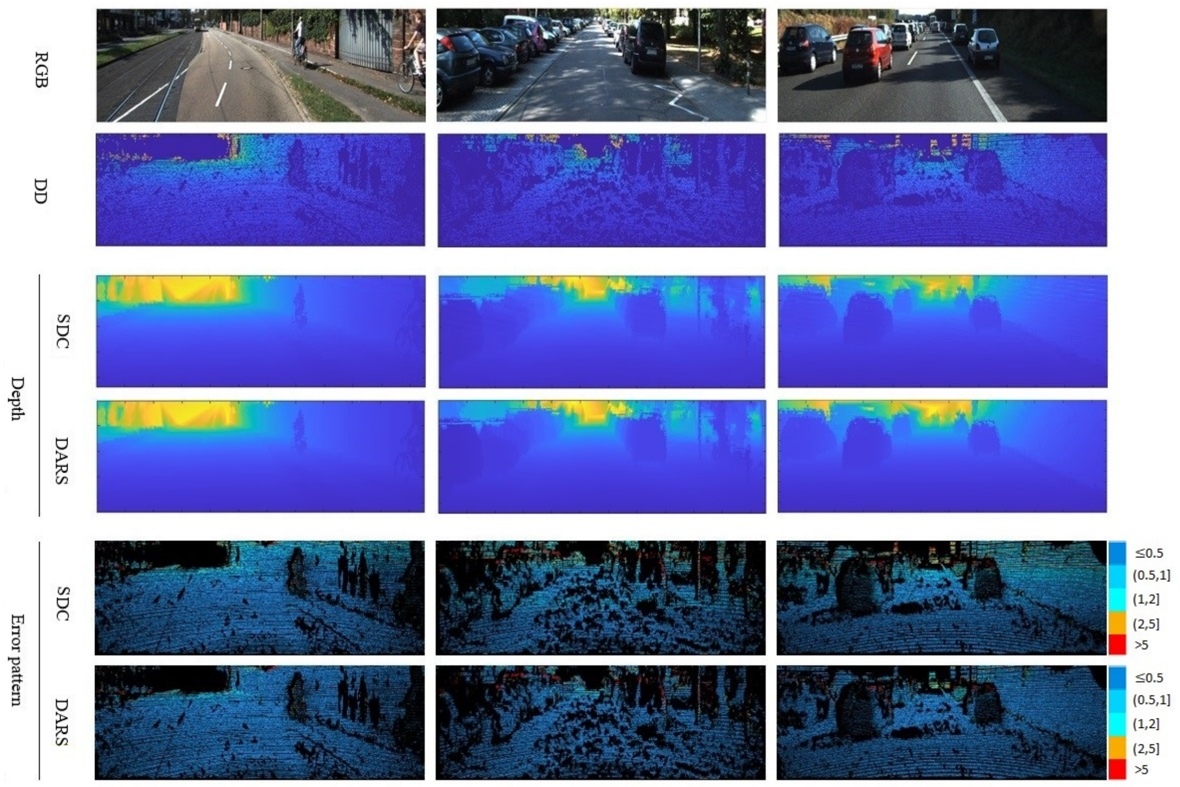


Figure 6. Visual results related to ablation study of KITTI dataset. Numbers on the right side of error patterns are in meters.

In the meanwhile, PSO improved the results significantly in terms of all metrics and on both KITTI and NYUv2 datasets. For instance, PSO showed nearly 50% enhancement in AbsRel and 14% in RMSE on KITTI and 6% and 86% respectively in terms of AbsRel and RMSE on NYUv2.

If we compare the improvement of PSO over L-BFGS on KITTI and that on NYUv2, it can be observed that the improvement was more remarkable on NYUv2. Thus, considering that the baseline was trained on KITTI, one can conclude that the optimization module with PSO as its optimizer, plays a significant role in the adaptation process. This observation also demonstrated the capability and efficacy of the activation optimization used in the proposed scheme.

To conclude, both of the proposed correction and optimization stages in DARS, i.e., SDC and activation optimization using PSO, proved to be effective and led to considerable improvements. Moreover, DARS proved its capability in network adaptation, given its performance on NYUv2.

As is clear from error patterns in Figure 6, related to KITTI and Figure 7 pertaining to NYUv2, the introduction of PSO has led to considerable improvements. The improvements can be specifically observed in more distant pixels which are usually of a higher error magnitude.

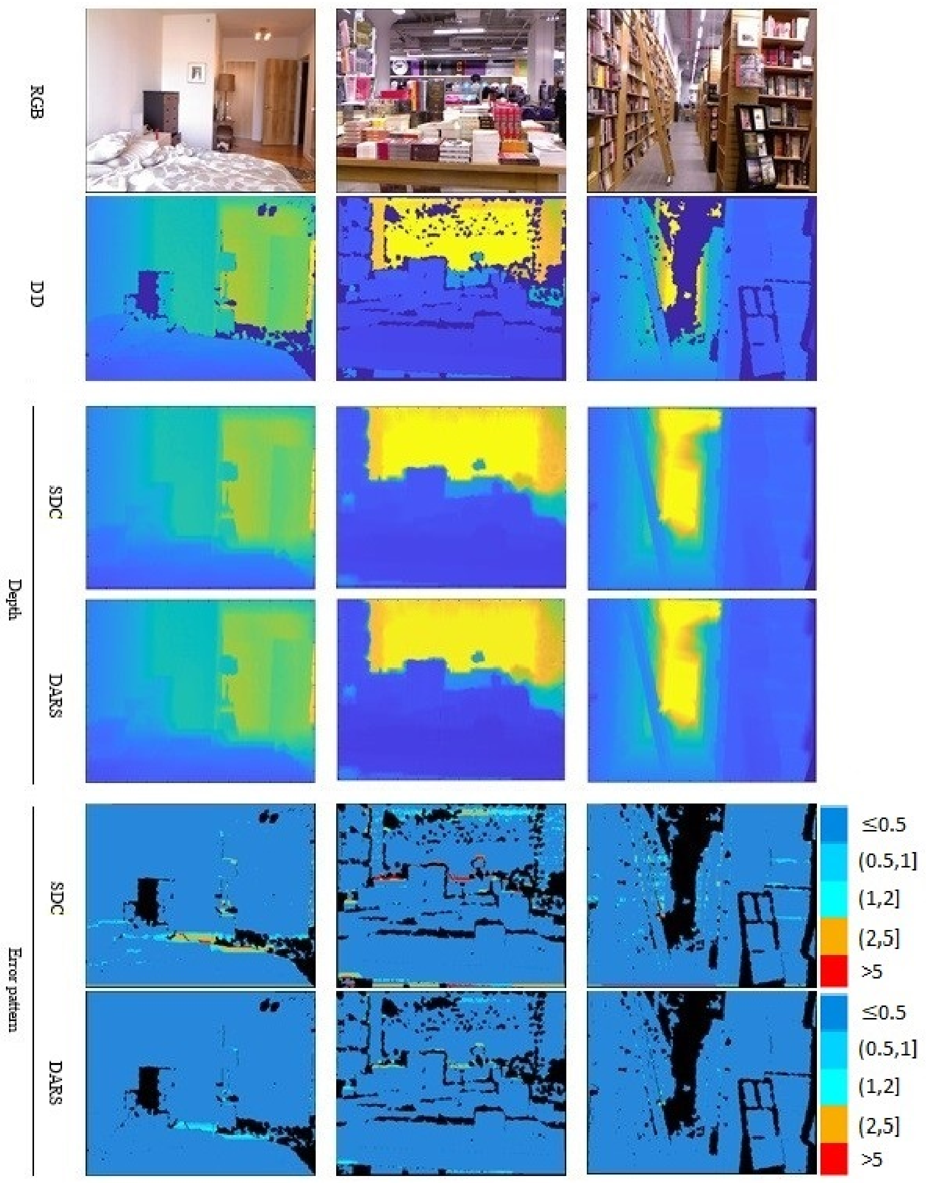


Figure 7. Visual results related to ablation study of NYUv2 dataset. Numbers on the right side of error patterns are in meters.

5.6. Comparison with SOTA

As is clear from Table 2, DARS outperformed competing methods in terms of almost all assessment criteria except for $\delta_{1.25^2}$ and $\delta_{1.25^3}$. From the perspective of these two criteria, the performance of our method was not as good as the second-place rival. However, DARS led to better performance in terms of $\delta_{1.25}$, which is the primary criterion for accuracy assessment. Although DARS utilizes a self-supervised baseline, Monodepth2, it outperformed its supervised rivals by a 39% margin in terms of RMSE on KITTI. This confirms the superiority of the proposed DARS even over supervised approaches and in dealing with harder scenes in a seen dataset.

Table 2. Comparative Study on KITTI.

Method	Lower is better				Higher is better		
	AbsRel	SqRel	RMSE	$\text{RMSE}_{\log}$	$\delta_{1.25}$	$\delta_{1.25^2}$	$\delta_{1.25^3}$
[49]	0.120	0.789	4.755	0.177	0.856	0.961	0.987
[28]	0.132	0.994	5.240	0.193	0.833	0.953	0.985
[26]	0.114	0.898	4.935	0.206	0.861	0.949	0.976
[23]	0.090	0.545	3.942	0.137	0.914	0.983	0.995
[50]	0.090	0.424	3.419	0.133	0.916	0.984	0.996
[51]	0.060	0.231	2.642	0.094	0.958	0.994	0.999
[14]	0.058	0.190	2.360	0.088	0.964	**0.995**	**0.999**
DARS	**0.024**	**0.137**	**1.442**	**0.071**	**0.985**	0.993	0.996

Regarding the second dataset, NYUv2, DARS outperformed the competing methods in terms of all criteria according to Table 3. In terms of AbsRel and RMSE, DARS reached improvements of respectively 83% and 70% with respect to the best competing method. Furthermore, this table indicates how the proposed method successfully adapted to an unseen dataset. Note that unlike DARS, the other methods in Table 3 have been trained on NYUv2. Hence, one can deduce that DARS not only could adapt a network to an unseen dataset but also outperformed the methods trained on the exact same dataset. Also, it suggests DARS as a possible alternative to supervised approaches which suffer from complicated generalization problems in practice. This adaptation capability is extremely advantageous in applications with constantly-changing environments such as SLAM, where the scenes are of an unlimited variety and sparse LiDAR maps are available on the fly.

Table 3. Comparative Study on NYUv2.

Method	Lower is better			Higher is better		
	AbsRel	RMSE	$\text{RMSE}_{\log}$	$\delta_{1.25}$	$\delta_{1.25^2}$	$\delta_{1.25^3}$
[47]	0.158	0.641	-	0.769	0.950	0.988
[15]	0.110	0.392	0.047	0.885	0.978	0.994
[51]	0.107	0.373	0.046	0.893	0.985	0.997
[14]	0.103	0.364	**0.044**	0.903	0.984	0.997
DARS	**0.017**	**0.109**	**0.044**	**0.993**	**0.996**	**0.999**

6. Conclusion

This paper deals with one of the main problems of available deep learning-based depth estimation networks, which is their limited generalization capability. This problem specifically restricts the practical usage of such models in applications with a constantly-changing environment, such as SLAM. To alleviate this problem, a new double-stage adaptive refinement scheme for depth estimation networks, namely, DARS based on the combination of f-BRS and PSO is proposed in this paper. DARS, here, is injected into Monodepth2 as the baseline and adapts the pre-trained network to each input during inference. Experimental results on KITTI and NYUv2 datasets, demonstrated the efficacy of the proposed scheme not only for KITTI but also for NYUv2, while the baseline model was pre-trained only on KITTI. Although our approach is model agnostic by design, this paper did not explore the effects of using different baselines. In future work, we will therefore replace our unsupervised baseline with other networks, ranging from unsupervised to supervised to investigate the effectiveness of our proposed scheme on different baselines.

Author Contributions: A.A.: conceptualization, methodology, software, formal analysis, writing-original draft preparation, and visualization; M.S.: methodology, software, and writing—review and editing; G.S.: Conceptualization, Supervision, writing—review and editing, and project administration.

Funding: Please add: "This research received no external funding" or "This research was funded by NAME OF FUNDER grant number XXX." and and "The APC was funded by XXX". Check carefully that the details given are accurate and use the standard spelling of funding agency names at https://search.crossref.org/funding, any errors may affect your future funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Yousif, K.; Taguchi, Y.; Ramalingam, S. MonoRGBD-SLAM: Simultaneous localization and mapping using both monocular and RGBD cameras. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2017, pp. 4495–4502.
2. Li, R.; Wang, S.; Long, Z.; Gu, D. Undeepvo: Monocular visual odometry through unsupervised deep learning. In Proceedings of the 2018 IEEE international conference on robotics and automation (ICRA). IEEE, 2018, pp. 7286–7291.
3. Dimas, G.; Gatoula, P.; Iakovidis, D.K. MonoSOD: Monocular Salient Object Detection based on Predicted Depth. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2021, pp. 4377–4383.
4. Hu, M.; Wang, S.; Li, B.; Ning, S.; Fan, L.; Gong, X. PENet: Towards Precise and Efficient Image Guided Depth Completion. *arXiv preprint arXiv:2103.00783* **2021**.
5. Park, J.; Joo, K.; Hu, Z.; Liu, C.K.; So Kweon, I. Non-local spatial propagation network for depth completion. In Proceedings of the Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16. Springer, 2020, pp. 120–136.
6. Gurram, A.; Tuna, A.F.; Shen, F.; Urfalioglu, O.; López, A.M. Monocular Depth Estimation through Virtual-world Supervision and Real-world SfM Self-Supervision. *arXiv preprint arXiv:2103.12209* **2021**.
7. Hirose, N.; Koide, S.; Kawano, K.; Kondo, R. Plg-in: Pluggable geometric consistency loss with wasserstein distance in monocular depth estimation. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2021, pp. 12868–12874.
8. Liu, J.; Li, Q.; Cao, R.; Tang, W.; Qiu, G. MiniNet: An extremely lightweight convolutional neural network for real-time unsupervised monocular depth estimation. *ISPRS Journal of Photogrammetry and Remote Sensing* **2020**, *166*, 255–267.
9. Hwang, S.J.; Park, S.J.; Kim, G.M.; Baek, J.H. Unsupervised Monocular Depth Estimation for Colonoscope System Using Feedback Network. *Sensors* **2021**, *21*. https://doi.org/10.3390/s21082691.
10. Wang, Y.; Zhu, H. Monocular Depth Estimation: Lightweight Convolutional and Matrix Capsule Feature-Fusion Network. *Sensors* **2022**, *22*, 6344.
11. Cheng, X.; Wang, P.; Guan, C.; Yang, R. Cspn++: Learning context and resource aware convolutional spatial propagation networks for depth completion. In Proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, 2020, Vol. 34, pp. 10615–10622.
12. Kuznietsov, Y.; Stuckler, J.; Leibe, B. Semi-supervised deep learning for monocular depth map prediction. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 6647–6655.
13. Huang, Y.K.; Liu, Y.C.; Wu, T.H.; Su, H.T.; Chang, Y.C.; Tsou, T.L.; Wang, Y.A.; Hsu, W.H. S3: Learnable Sparse Signal Superdensity for Guided Depth Estimation. In Proceedings of the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 16706–16716.
14. Bhat, S.F.; Alhashim, I.; Wonka, P. Adabins: Depth estimation using adaptive bins. In Proceedings of the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 4009–4018.
15. Lee, J.H.; Han, M.K.; Ko, D.W.; Suh, I.H. From big to small: Multi-scale local planar guidance for monocular depth estimation. *arXiv preprint arXiv:1907.10326* **2019**.
16. Lee, S.; Lee, J.; Kim, B.; Yi, E.; Kim, J. Patch-Wise Attention Network for Monocular Depth Estimation. In Proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, 2021, Vol. 35, pp. 1873–1881.
17. Liu, Y.; Yixuan, Y.; Liu, M. Ground-aware monocular 3d object detection for autonomous driving. *IEEE Robotics and Automation Letters* **2021**, *6*, 919–926.
18. Jang, W.D.; Kim, C.S. Interactive Image Segmentation via Backpropagating Refinement Scheme. In Proceedings of the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

19. Sofiiuk, K.; Petrov, I.; Barinova, O.; Konushin, A. f-brs: Rethinking backpropagating refinement for interactive segmentation. In Proceedings of the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 8623–8632.
20. Lau, M.M.; Lim, K.H. Review of adaptive activation function in deep neural network. In Proceedings of the 2018 IEEE-EMBS Conference on Biomedical Engineering and Sciences (IECBES). IEEE, 2018, pp. 686–690.
21. Huang, X.; Belongie, S. Arbitrary style transfer in real-time with adaptive instance normalization. In Proceedings of the Proceedings of the IEEE international conference on computer vision, 2017, pp. 1501–1510.
22. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the Proceedings of ICNN'95-international conference on neural networks. IEEE, 1995, Vol. 4, pp. 1942–1948.
23. Godard, C.; Mac Aodha, O.; Firman, M.; Brostow, G.J. Digging into self-supervised monocular depth estimation. In Proceedings of the Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 3828–3838.
24. Zhou, T.; Brown, M.; Snavely, N.; Lowe, D.G. Unsupervised learning of depth and ego-motion from video. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1851–1858.
25. Ye, X.; Ji, X.; Sun, B.; Chen, S.; Wang, Z.; Li, H. DRM-SLAM: Towards dense reconstruction of monocular SLAM with scene depth fusion. *Neurocomputing* **2020**, *396*, 76–91.
26. Godard, C.; Mac Aodha, O.; Brostow, G.J. Unsupervised monocular depth estimation with left-right consistency. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 270–279.
27. Bian, J.; Li, Z.; Wang, N.; Zhan, H.; Shen, C.; Cheng, M.M.; Reid, I. Unsupervised scale-consistent depth and ego-motion learning from monocular video. *Advances in neural information processing systems* **2019**, *32*, 35–45.
28. Yin, Z.; Shi, J. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 1983–1992.
29. Cai, H.; Matai, J.; Borse, S.; Zhang, Y.; Ansari, A.; Porikli, F. X-Distill: Improving Self-Supervised Monocular Depth via Cross-Task Distillation. *arXiv preprint arXiv:2110.12516* **2021**.
30. Feng, T.; Gu, D. Sganvo: Unsupervised deep visual odometry and depth estimation with stacked generative adversarial networks. *IEEE Robotics and Automation Letters* **2019**, *4*, 4431–4437.
31. Ji, P.; Li, R.; Bhanu, B.; Xu, Y. Monoindoor: Towards good practice of self-supervised monocular depth estimation for indoor environments. In Proceedings of the Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 12787–12796.
32. Fei, X.; Wong, A.; Soatto, S. Geo-supervised visual depth prediction. *IEEE Robotics and Automation Letters* **2019**, *4*, 1661–1668.
33. dos Santos Rosa, N.; Guizilini, V.; Grassi, V. Sparse-to-continuous: Enhancing monocular depth estimation using occupancy maps. In Proceedings of the 2019 19th International Conference on Advanced Robotics (ICAR). IEEE, 2019, pp. 793–800.
34. Ma, F.; Cavalheiro, G.V.; Karaman, S. Self-supervised sparse-to-dense: Self-supervised depth completion from lidar and monocular camera. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA). IEEE, 2019, pp. 3288–3295.
35. Ming, Y.; Meng, X.; Fan, C.; Yu, H. Deep learning for monocular depth estimation: A review. *Neurocomputing* **2021**, *438*, 14–33.
36. Palnitkar, R.M.; Cannady, J. A review of adaptive neural networks. In Proceedings of the IEEE SoutheastCon, 2004. Proceedings. IEEE, 2004, pp. 38–47.
37. Kontogianni, T.; Gygli, M.; Uijlings, J.; Ferrari, V. Continuous adaptation for interactive object segmentation by learning from corrections. In Proceedings of the European Conference on Computer Vision. Springer, 2020, pp. 579–596.
38. Gatys, L.; Ecker, A.S.; Bethge, M. Texture synthesis using convolutional neural networks. *Advances in neural information processing systems* **2015**, *28*.
39. Gatys, L.A.; Ecker, A.S.; Bethge, M. Image style transfer using convolutional neural networks. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2414–2423.
40. Simonyan, K.; Vedaldi, A.; Zisserman, A. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034* **2013**.
41. Zhang, J.; Bargal, S.A.; Lin, Z.; Brandt, J.; Shen, X.; Sclaroff, S. Top-down neural attention by excitation backprop. *International Journal of Computer Vision* **2018**, *126*, 1084–1102.
42. Amidror, I. Scattered data interpolation methods for electronic imaging systems: a survey. *Journal of electronic imaging* **2002**, *11*, 157–176.
43. Zhu, C.; Byrd, R.H.; Lu, P.; Nocedal, J. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on mathematical software (TOMS)* **1997**, *23*, 550–560.
44. Byrd, R.H.; Lu, P.; Nocedal, J.; Zhu, C. A limited memory algorithm for bound constrained optimization. *SIAM Journal on scientific computing* **1995**, *16*, 1190–1208.
45. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research* **2013**, *32*, 1231–1237.
46. Silberman, N.; Hoiem, D.; Kohli, P.; Fergus, R. Indoor segmentation and support inference from rgbd images. In Proceedings of the European conference on computer vision. Springer, 2012, pp. 746–760.
47. Eigen, D.; Puhrsch, C.; Fergus, R. Depth map prediction from a single image using a multi-scale deep network. *arXiv preprint arXiv:1406.2283* **2014**.

48. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*; Wallach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E.; Garnett, R., Eds.; Curran Associates, Inc., 2019; pp. 8024–8035.
49. Luo, C.; Yang, Z.; Wang, P.; Wang, Y.; Xu, W.; Nevatia, R.; Yuille, A. Every pixel counts++: Joint learning of geometry and motion with 3d holistic understanding. *IEEE transactions on pattern analysis and machine intelligence* **2019**, *42*, 2624–2641.
50. Zhang, J.; Li, W.; Gou, H.; Fang, L.; Yang, R. LEAD: LiDAR Extender for Autonomous Driving. *arXiv preprint arXiv:2102.07989* **2021**.
51. Lee, M.; Hwang, S.; Park, C.; Lee, S. EdgeConv with Attention Module for Monocular Depth Estimation. *arXiv preprint arXiv:2106.08615* **2021**.