sensors

MDPI

Article

Superb Monocular Depth Estimation Based on Transfer Learning and Surface Normal Guidance

Kang Huang [1], Xingtian Qu [1], Shouqian Chen [2], Zhen Chen [1], Wang Zhang [1,*], Haogang Qi [1] and Fengshang Zhao [1]

[1] Department of Mechanical Engineering and Automation, School of Mechanical and Aerospace Engineering, Jilin University, Changchun 130022, China; huangkang18@mails.jlu.edu.cn (K.H.); quxt@jlu.edu.cn (X.Q.); zhenchen18@mails.jlu.edu.cn (Z.C.); qihg18@mails.jlu.edu.cn (H.Q.); 15951264800@163.com (F.Z.)

[2] Research Center for Space Optical Engineering, Harbin Institute of Technology, P.O. Box 307, Harbin 150001, China; shouqian.chen@hit.edu.cn

* Correspondence: wangzhang@jlu.edu.cn

Received: 3 August 2020; Accepted: 25 August 2020; Published: 27 August 2020

check for updates

Abstract: Accurately sensing the surrounding 3D scene is indispensable for drones or robots to execute path planning and navigation. In this paper, a novel monocular depth estimation method was proposed that primarily utilizes a lighter-weight Convolutional Neural Network (CNN) structure for coarse depth prediction and then refines the coarse depth images by combining surface normal guidance. Specifically, the coarse depth prediction network is designed as pre-trained encoder–decoder architecture for describing the 3D structure. When it comes to surface normal estimation, the deep learning network was designed as a two-stream encoder–decoder structure, which hierarchically merges red-green-blue-depth (RGB-D) images for capturing more accurate geometric boundaries. Relying on fewer network parameters and simpler learning structure, better detailed depth maps are produced than the existing states. Moreover, 3D point cloud maps reconstructed from depth prediction images confirm that our framework can be conveniently adopted as components of a monocular simultaneous localization and mapping (SLAM) paradigm.

Keywords: SLAM; SFM; supervised deep learning; multi-task learning; transfer learning; monocular depth estimation; surface normal estimation

1. Introduction

1.1. Background

Compared with depth estimation methods relying on laser rangefinders or other optical instruments, the computer vision method does not require expensive optical equipment and repeated lens calibration. Therefore, image-based depth prediction has been extensively studied and widely applied to 3D scene understanding tasks, such as structure from motion (SFM) [1,2], simultaneous localization and mapping (SLAM) [3,4], 3D object detection [5], etc.

The computer vision method, i.e., image-based depth estimation, defines image depth as the distance from the object point corresponding to each pixel to the camera and exploits clues of images like linear perspective, focus, occlusion, texture, shadow, gradient, etc. for calculation. All the image-based methods can be summarized as two classes: stereo vision methods and monocular methods. The stereo vision methods are heavily dependent on natural light in a natural environment to collect images and is sensitive to changes in illumination angle and changes in illumination intensity. The differences in image matching of the two pictures will result in considerable differences from the matching algorithm. Compared with the stereo vision methods, the monocular vision systems [6–9] rarely encounter the aforementioned problems. Therefore, the monocular depth estimation method is more convenient for calibration or identification than the stereo vision approach.

Despite great achievements [6,10,11] in terms of monocular depth estimation, previous research has not given it enough attention [6,7,10–12]. First, not enough importance is attached to intrinsically consistency such as semantics labels, color clues of the red-green-blue (RGB) images, the surface normal of the depth images, and the linear perspective of the depth images. The state-of-the-art (SOTA) algorithms [7,9,11,13] leverage fairly complex deep neural networks that are too slow for real-time inference.

1.2. Ideas

In terms of the first content above, there are several methods for acquiring image depth based on supervised learning that can achieve ideal depth maps. However, establishing labeled datasets such as NYU depth V2 [14], Scan-Net [15], and Make-3D [16] is such a challenge for supervised learning methods, such that the datasets [14–16] have been constructed for over five years. The ground truth (GT) depth images from these datasets were thus captured by the outdated depth sensors. Therefore, the geometric accuracy of GT images definitely fluctuated, which would cause side effects on the precision of both training and testing processes. Therefore, we first predicted depth maps from RGB images based on an encoder–decoder structure and denoted those images as "coarse depth maps". Then, surface normal maps were estimated and used for applying geometric guidance to the depth estimation process, i.e., the surface normal estimation was introduced in this paper so as to enable accurate regression of geometric structure and complex boundaries of 3D objects.

As for the second content, previous authors [6,17,18] chose to propose a lighter-weight model-based encoder–decoder network, which produced more accurate depth estimation maps and reduced computational complexity. Wofk et al. [17] demonstrated that a well-designed low latency network can maintain real-time depth estimation with economical computation costs. Alhashim et al. [6] proved that a very simple transfer learning-based decoder robustly achieves high-resolution depth maps. Previous studies [6,9,19] proved that the Dense-net is more suitable for depth estimation than models like SE-Net, Res-Net, and Mobile-net. However, experiments in references [6,9] showed that pre-trained deep learning structures based on Densenet-161, Densenet-169, and Densenet-201 models cannot afford real-time depth estimation. As a result, a lighter-weight depth estimation network Densenet-121 was adopted in this paper, which requires fewer parameters and inference iterations than other dense-net models. Moreover, in order to enhance the performance of the Densenet-121 model, the surface normal maps were jointly estimated with depth images.

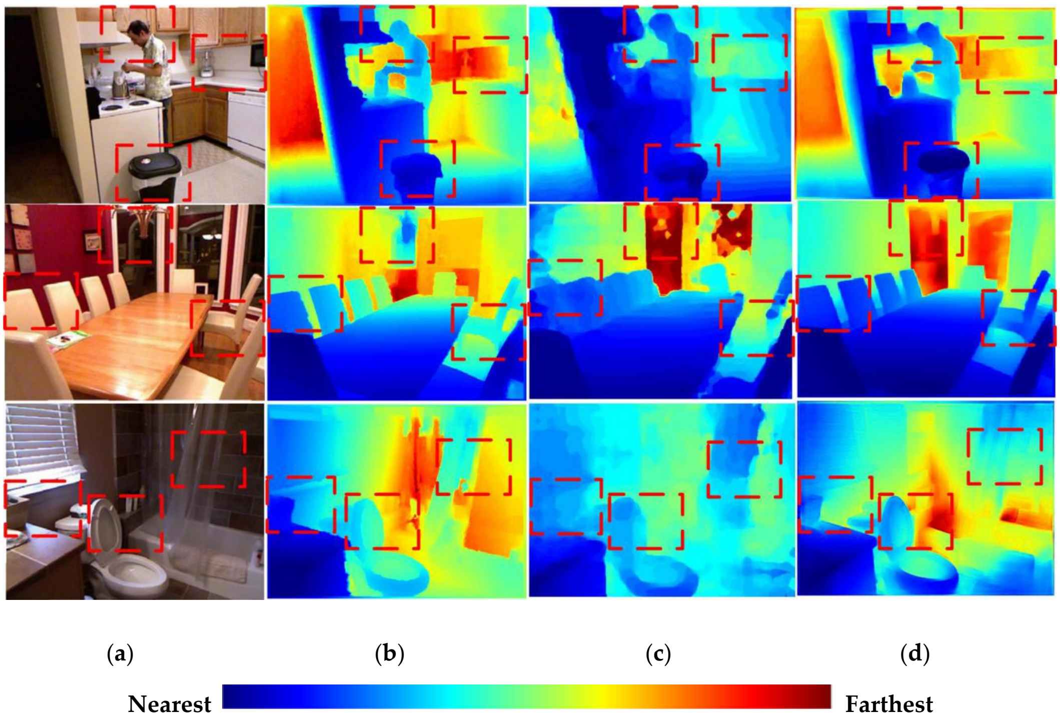
1.3. Approach

The reform of current monocular depth framework can be summarized as follows. First, a novel depth estimation network called the coarse depth estimation (CDE) network was proposed, which leverages an efficient encoder–decoder network architecture. In terms of deep-feature-encoding layers, we introduced a pre-trained Densenet-121 model that was primitively proposed for image classification and object detection. As for the decoder, we chose the 2× bilinear proposed by Alhashim et al. [6] as the up-sampling method. Second, we put forward a network called the red-green-blue-depth (RGB-D) surface normal (RSN) network in order to obtain more accurate surface normal maps. The RSN network was designed as a two-stream encoder–decoder architecture, which produced surface normal maps by hierarchically fusing coarse depth produced by the CDE network and RGB images from a selected dataset. Then, the refinement network exploited surface normal maps produced by the RSN network for ameliorating the quality of coarse depth images.

Three-dimensional features can be represented with different formats, including point clouds, meshes, and volumetric grids. Specifically, the point cloud map is a set of data points preserving the original geometric information of 3D objects without any discretization. Three-dimensional point cloud maps were used as a criterion for evaluating the depth maps produced in our study.

1.4. Contributions

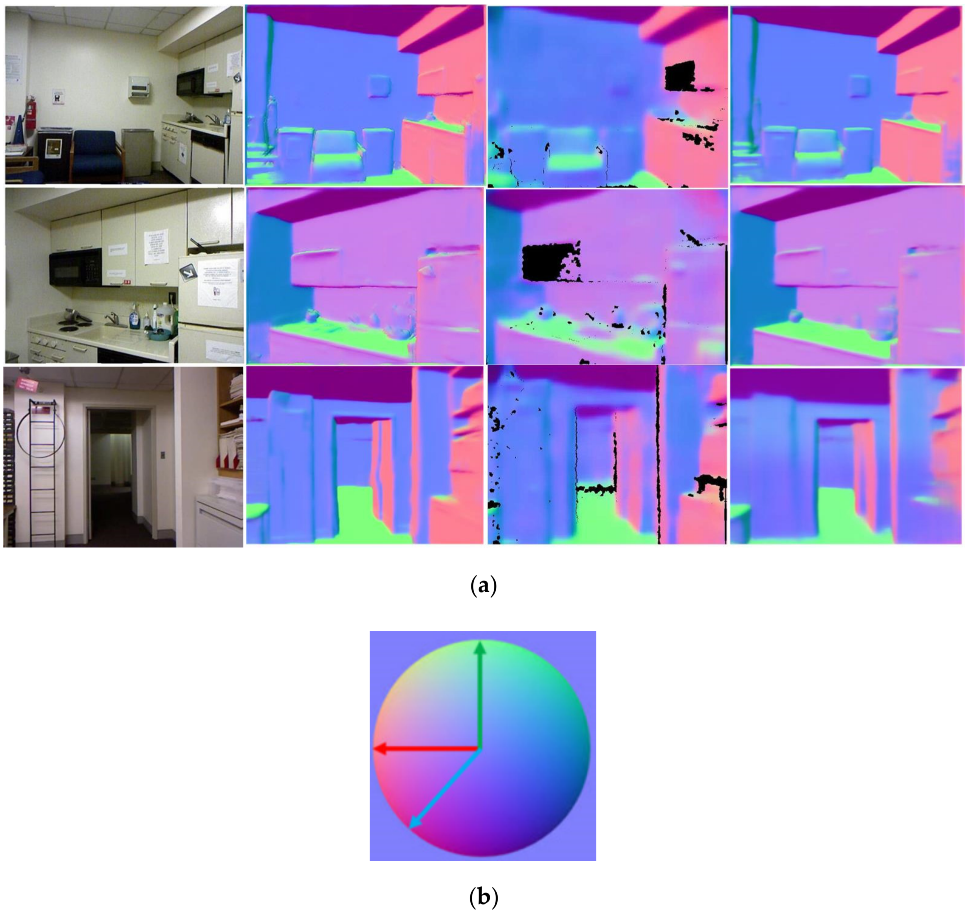
Different from previous studies [10,20], we remodeled the conventional depth estimation network layout. Repeated experiments proved that a deep network based on the low latency Densenet-121 model affords real-time coarse depth estimation and depth refinement depending on a Tesla M40 produced by NVIDA in Santa Clara, CA, U.S. with single 12G memory capacity. Benefiting from guidance of surface normal maps, the Densenet-121 based network obtained better depth maps than SOTA practices [6,7,9]. As shown in Figure 1, our network outperforms SOTA depth estimation on an NYU [14] dataset and produces higher-resolution results that capture object boundaries more faithfully.



For all depth maps in our work, we set the image color according to distance as the color bar above

Figure 1. The comparison of depth maps were produced by different methods. (**a**) Raw red-green-blue (RGB) images (**b**) Ground truth (GT) depth maps [14], (**c**) Depth maps from the state-of-the-art (SOTA) practice [7], (**d**) Depth maps from our depth prediction network.

Our second contribution is reflected in proposing an RGB-D surface normal network, which effectively captures the geometric relationships between RGB and depth images. Different from previous frameworks [21,22], we proposed a fusion network leveraging both RGB and coarse depth prediction images instead of using RGB images only. Images from different domains can complement each other in the surface normal estimation process, i.e., coarse depth images from the CDE network contribute to enhance geometry details, and RGB images make up the missing depth pixels. As shown in Figure 2, we achieved better surface normal maps than Qi et al. [10] based on the RGB-D surface normal network. Moreover, as we can see in Figure 1, with the geometrical guidance of surface normal, depth maps with distinct boundaries were acquired.



(**a**)

(**b**)

Figure 2. The comparison of surface normal maps: (**a**) from left to right: RGB images, Ground-Truth (GT), surface normal maps produced by Qi et al. [10], ours. (**b**) Color-map definition: red represents left, green represents up, and blue represents outward.

2. Related Work

Stereo depth estimation can be regarded as a well-posed process once ignoring the problem of occlusions and depth discontinuities. Moreover, depth estimation methods based on stereo vision achieved even more accurate and robust depth maps than RGB-D sensors [14,15]. Meanwhile, more precise depth features from our monocular method can also make contributions to better multi-view stereo reconstruction.

Monocular depth estimation has been considered by plenty of researchers [23–29], which usually defined the estimation as a regression of the depth map from a single RGB image. Eigen et al. [21] introduced the application of CNNs in monocular depth estimation, which inspired researchers to explore methods based on deep learning. At present, deep learning methods play a leading role in monocular depth estimation practices. Generally, deep learning methods can be divided into supervised approaches, self-supervised approaches, and unsupervised approaches. Supervised monocular depth methods achieved great breakthroughs relying on well-annotated ground-truth images offered by datasets [14–16]. For example, Liu et al. [28] combined CNNs with CRF to learn the super-pixel-wise connections between depth and RGB images. Different from the supervised method, self-supervised methods usually set up a separate camera pose estimation network [26,27] or a jointly calculated optical flow and camera pose [30]. Unsupervised practices learn the scene depth image synthesis [10] or ego-motion in monocular videos without using ground truth data [24,25]. Similar to the SOTA methods [2,7], in recent years, the supervised learning method was selected in this paper.

Multi-task/cross-task learning was designed based on intrinsic connections of physical elements of the research targets selected. Some recent works attempt to investigate the sharing of image features between different tasks [31–33]. Jiao et al. [34] jointly trained semantic labeling and depth estimation in their encoder–decoder network architecture. Zhang et al. [11] proposed a pattern-affinity propagation method for jointly predicting depth, surface normal and semantic segmentation. As for us, the proposed network first jointly predicts depth and surface normal and then takes advantage of surface normal maps to refine the predicted coarse depth maps.

Surface normal guidance has been introduced by previous studies [10,19,20], where they employed surface normal maps as 3D cues for improving the geometric quality of monocular depth images. Qi et al. [10] jointly calculated depth and surface normal from a single image, making the final estimation geometrically more precise. In work of Zeng et al. [19], a skip-connected architecture was proposed to fuse features from different layers for surface normal estimation. A novel 3D geometric feature virtual normal was proposed by Yin et al. [20] to refine the predicted depth maps. Surface normal estimation was adopted in this paper for calculating the angular difference between predicted depth images and ground-truth maps, which thus applied geometric restriction to depth images.

A transfer-learning-based deep framework was adopted by Zhang et al. [11] to obtain the state-of-the-art semantic segmentation maps. We adopt a pre-trained Densenet-121 model [5] as the backbone of the coarse depth estimation network in depth feature extraction. Our method benefits from the application of transfer learning, where we take advantage of encoders originally designed for 3D object detection by Alhashim [6].

Encoder–decoder networks were widely adopted in various computer vision tasks such as image classification, image segmentation, and 3D objects detection. In recent years, such architectures made significant contributions to both supervised learning and unsupervised learning-based practices of monocular depth estimation [3,30]. We devised a compendious but effective symmetrical encoder–decoder structure with skip connections. Repeated experiments indicated that our encoder–decoder network with simple structure can outperform the SOTA depth synthesis based on more complicated deep learning architecture [7,9].

3. Our Method

This section presents our method for monocular depth estimation. First, the general deep learning framework is introduced. Then, rational loss functions for overall training process are defined. Finally, we discuss the practice of 3D point cloud maps reconstruction.

3.1. Framework Overview

As shown in Figure 3, the whole deep learning structure consists of three parts: a coarse depth estimation network, an RGB-D surface normal network, and a refinement network.

The coarse depth estimation (CDE) network leverages efficient encoder–decoder network architecture. Coarse depth images generated from the CDE network are then fed to the RGB-D surface normal network for RGB-D fusion. Moreover, coarse depth images are also converted to coarse surface normal maps based on a fix-weight network [4]. For convenience, we describe a single RGB input image as I_C, a single in-painted ground truth depth image produced by Levin [35] as D^{GT}, an output coarse depth map as D^*, and a coarse surface normal map recovered from D^* as N^*.

The RGB-D surface normal (RSN) network was designed for obtaining accurate surface normal maps, which functions as the refining coarse depth map (D^*). As shown in Figure 3, the RSN network can be divided into two streams—the RGB stream and the depth stream. The latter can be further divided into depth branch and the confidence map branch. For the general architecture of RSN network, we define a single RGB input as I_F, a corresponding coarse depth input as D^*, and a surface normal output map as N_F.

The RGB stream and depth branch in Figure 3 operate respectively to generate RGB feature $R_1 \sim R_4$ and relative sensor depth feature $D_1 \sim D_4$ with hierarchical resolution. Then, the two branches cooperate to combine and fuse features from each branch.

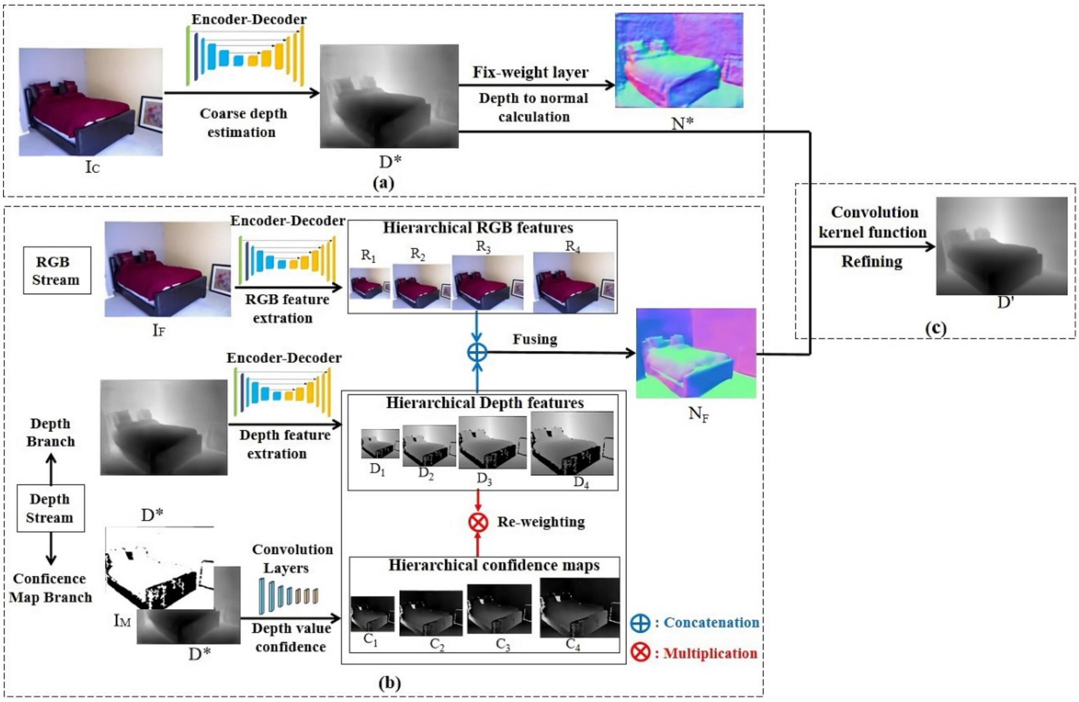


Figure 3. General estimation framework. (**a**) Coarse depth estimation network; (**b**) red-green-blue-depth (RGB-D) surface normal network; (**c**) refinement network.

The ground truth depth images (D^{GT}) used for training the CDE network were produced following Levin et al. [35]. This method greatly contributes to fill pixel holes in sensor depth images from NYU-Depth-V2 [14]. However, this method cannot eliminate the side effects caused by lost pixels on accuracy of ground truth depth images and corresponding coarse depth images (D^*). Therefore, a confidence map network branch was set following the method proposed by Zeng et al. [11], which generates confidence maps to indicate whether side effects resulted from pixel holes on D^* or not. Confidence maps [19] of depth image were produced by combining mask images [21] (I_M) with relative coarse depth images (D^*) and were denoted as $C_1 \sim C_4$ according to resolution.

The refinement network actually servers as the convolution kernel function, which optimizes the coarse depth maps from the CDE network guided by surface normal maps from the RSN network. Finally, with the aid of the refinement network, a superbly accurate depth map D' was generated.

3.2. Coarse Depth Estimation (CDE) Network

Figure 4 shows the distinct structure of the encoder–decoder network for getting coarse depth maps (D^*). For the encoder, the raw RGB image (I_C) is encoded into a feature vector based on a Densenet-121 [5] model that has been pre-trained on Image-Net [36]. The feature vector is then transmitted to a sequence of up-sampling layers to produce D^* with half resolution of I_C. As for decoding operation, the decoder network consists of four up-sampling units (BU_1 and $USB_1 \sim USB_3$) and relative concatenation (⊕) skip-connections. In decoding layers, the 2× bi-linear interpolation proposed by Alhashim [6] is adopted as an up-sampling method.

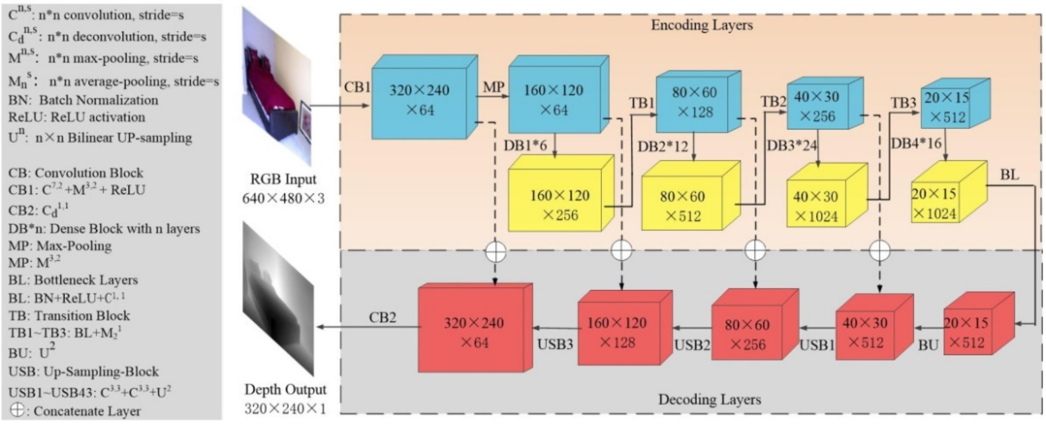
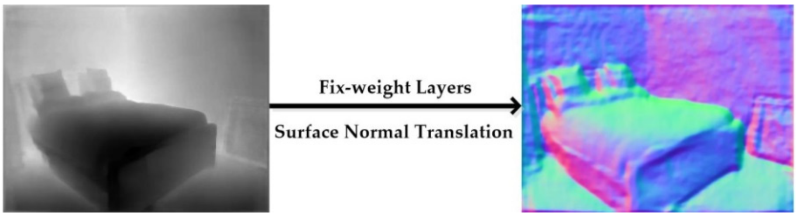


Figure 4. Encoder–decoder coarse depth network.

As shown in Figure 5, the coarse image D^* is then converted to coarse surface normal image (N^*) based on least square algorithm [37], and the inference network is just a fix-weight network [11].



(**a**) (**b**)

Figure 5. Generating coarse surface normal image. (**a**) Coarse depth (D^*); (**b**) N^*.

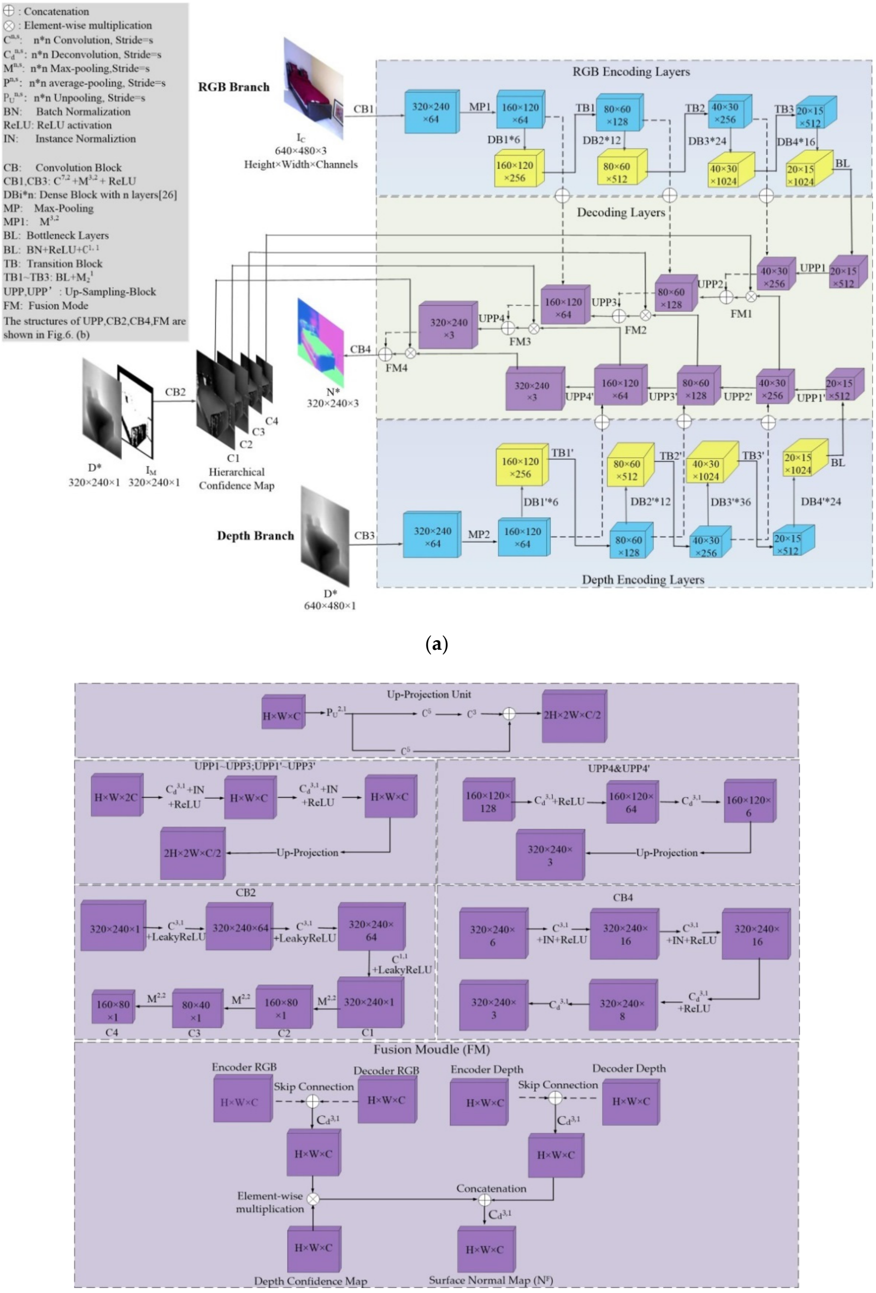
3.3. RGB-D Surface Normal (RSN) Network

Both the RGB stream and depth branch leverage encoder–decoder network architecture. The two branches employ a pre-trained Densenet-121 model as encoding backbone [26], whose detailed structure is illustrated in the upper row of Figure 6. Generally, the encoder consists of same convolution layers with the raw Densenet-121 [5] encoder, except the last convolution blocks. The number of channels was reduced from 1024 to 512 via bottleneck layers aiming to reduce redundant parameters. A symmetric decoder equipped with concatenation connections for the refitted encoder was designed. Multi-scale up-sampling layers were introduced to decoder, which enables RGB-D images fuse in different scales. What's more, common pooling masks were set to let the network learn more image features.

Different from CDE Network, the RSN Network leverages coarse depth images (D^*) instead of in-painted [35] ground truth depth images (D^*) as the depth input. The pre-trained RSN Network can thus be applied for estimating surface normal maps based on custom images.

As is shown in Figure 3, pixel holes in mask images (I_M) suggest that there are lots of missing pixels in ground truth depth images, which inevitably causes deviation to supervised learning. Therefore, we adopted a multi-layer convolution network (CB2) for producing confidence map C_l [19] of input depth images. l stands for scale value of images, i.e., if the resolution of 2D images can be denoted as $H \times W$, then the corresponding l is defined as $\{(l, H \times W)\} = \{(1, 40 \times 30); (2, 80 \times 60); (3, 160 \times 120); (4, 320 \times 240)\}$. The detailed structure of *CB*2 is shown in Figure 6b. The depth branch also adopts Densenet-121 model–based encoding layers, and the fusion calculation takes place at the decoder side. As shown in Figure 6a, the depth features D_l^* are passed into the fusion module in each scale l and re-weighted with the confidence map $C_1 \sim C_4$. Then re-weighted D_l^* are concatenated (⊗) with color features with same resolution and transmitted to a de-convolution layer to produce surface maps based on RGB-D fusion. Finally, the convolution block (*CB*4) of RSN Network generates the surface map N^*. The RGB-D fusion algorithm [19] can be expressed as Equation (1).

$$N_l' = deconv[I_{F_l} \oplus (C_l \otimes D_l^*)] \tag{1}$$


⊕ : Concatenation
⊗ : Element-wise multiplication
C^n,s: n*n Convolution, Stride=s
Cd^n,s: n*n Deconvolution, Stride=s
M^n,s: n*n Max-pooling,Stride=s
P^n,s: n*n average-pooling, Stride=s
PU^n,s: n*n Unpooling, Stride=s
BN: Batch Normalization
ReLU: ReLU activation
IN: Instance Normaliztion
CB: Convolution Block
CB1,CB3: C^7,2 +M^3,2 + ReLU
DBi*n: Dense Block with n layers[26]
MP: Max-Pooling
MP1: M^3,2
BL: Bottleneck Layers
BL: BN+ReLU+C^1,1
TB: Transition Block
TB1~TB3: BL+M2^1
UPP,UPP' : Up-Sampling-Block
FM: Fusion Mode
The structures of UPP,CB2,CB4,FM are shown in Fig.6. (b)
RGB Branch
Ic 640×480×3 Height×Width×Channels
RGB Encoding Layers
Decoding Layers
Depth Encoding Layers
Depth Branch
D* 640×480×1
D* 320×240×1
IM 320×240×1
C1 C2 C3 C4 Hierarchical Confidence Map
N* 320×240×3
CB1 CB2 CB3 CB4
MP1 MP2 TB1 TB2 TB3 TB1' TB2' TB3'
DB1*6 DB2*12 DB3*24 DB4*16 DB1'*6 DB2'*12 DB3'*36 DB4'*24
BL
UPP1 UPP2 UPP3 UPP4 UPP1' UPP2' UPP3' UPP4'
FM1 FM2 FM3 FM4
Up-Projection Unit
UPP1~UPP3;UPP1'~UPP3'
UPP4&UPP4'
CB2
CB4
Fusion Moudle (FM)
Encoder RGB
Decoder RGB
Encoder Depth
Decoder Depth
Skip Connection
Element-wise multiplication
Concatenation
Depth Confidence Map
Surface Normal Map (N^l)

(**a**)

(**b**)

Figure 6. Surface normal adjustment network (Dense-net-121 based). (**a**) The general structure of the RGB-D surface normal network (RSN) network; (**b**) the architectures of up-projection units, fusion module and convolution blocks.

The decoder layers of the RSN network were also designed based on the Densenet-121 model. Different from the CDE network, up-projection units [17] ($UPP_1 \sim UPP_4$ and $UPP_1' \sim UPP_4'$) were employed instead of bilinear interpolation for boosting the surface normal estimation process. The detailed structure of up-projection units is shown in the upper row of Figure 6b.

3.4. Refinement Network

For a random pixel i from the coarse depth images, we denote (h_i, w_i) as the location of pixel i in 2D space and (x_i, y_i, d_i^*) as the coordinate of corresponding 3D point, where d_i^* represents the coarse depth value. Similarly, surface normal can be noted as $(n_{x_i}, n_{y_i}, n_{d_i^*})$. Then, a tangent plane p_i [11] can be defined according to Equation (2).

$$n_{x_i}(x - x_i) + n_{y_i}(y - y_i) + n_{\, d_i^*}(d - d^*) = 0 \tag{2}$$

A small 3D neighborhood (H_i) of i was defined in previous studies [10,38]. For a random pixel $j \in H_i$, its depth value d_j. The depth prediction value of pixel i can be computed as Equation (2) according to the pinhole camera model,

$$d'_{ji} = \frac{\left(u_j - c_x\right)n_{x_i}/f_x + \left(v_j - c_y\right)n_{y_i}/f_y + n_{\, d_i^*}}{\left(u_i - c_x\right)n_{x_i}/f_x + \left(v_i - c_y\right)n_{y_i}/f_y + n_{\, d_i^*}} \times d_j \tag{3}$$

where f_x and f_y represent the focal length in x and y directions, respectively; C_x and C_y are coordinates of the lens principal points; and K represents the convolution kernel operation. Then, in order to refine depth value of pixel i, we applied kernel regression [10] to operate estimation on all pixels in H_i as

$$d'_i = \frac{\sum_{j\in H_i}[d_j^* K\left(n_i, n_j\right) d'_{ji}]}{\sum_{j\in H_i}[d'_{ij} K\left(n_i, n_j\right)]} \tag{4}$$

$$K\left(n_i, n_j\right) = n_j^T n_i \tag{5}$$

where d'_i is the refined depth value $n_i = [n_{x_i}, n_{y_i}, n_{d_i^*}]$. According to the linear kernel algorithm, the angle error between surface normal image n_{i} and image n_{j} decides whether pixels i and j are in the same tangent plane p_i or not. Therefore, the smaller angular error contributes to more accurate depth estimation d'_{ji}.

3.5. Loss Function

This section presents loss functions for regression problems existing in depth and surface normal estimation. The loss functions evaluate the difference between the ground truth images and the prediction images generated by the deep learning network. For random pixel i, we define the coarse depth map, refined depth map and ground-truth depth map as d_i^*, d'_i, and d_i^{gt} separately. Similarly, we denoted the coarse surface normal map, RGB-D fusion surface normal, and ground-truth surface normal as n_i^*, n_i^F, and n_i^{gt} separately. The total number of pixels is M, and the loss function for depth value is defined as Equation (6).

$$l_{depth} = \frac{1}{M}\left(\sum_i \| d'_i - d_i^{gt} \|_2^2 + \omega_1 \sum_i \| d_i^* - d_i^{gt} \|_2^2\right) \tag{6}$$

As shown in Equation (6), the loss function l_{depth} is the point-wise loss defined according to depth values. On the one hand, l_{depth} computes the sum of the $L2$ norm of the error vectors between coarse depth maps d_i^* and ground truth images d_i^{gt}. On the other hand, l_{depth} calculates difference between refined depth maps d'_i and ground truth images d_i^{gt} over all pixels. The loss for surface normal training process is defined as Equation (7),

$$l_{normal} = \frac{1}{M}\left(\mu_l \sum_i \| n_i^{F_l} - n_i^{gt} \|_2^2 + \omega_2 \sum_i \| n_i^*(t) - n_i^{gt} \|_2^2\right) \tag{7}$$

where l in Equation (7) stands for the scale value [19] of surface normal features and ω_1, ω_2, and μ_l stand for weight parameter of loss functions. The loss function l_{normal} computes the $L2$ norm of the angular errors of the orientation vectors for each pixel. On the one hand, l_{normal} computes the $L2$ norm of the angular difference between coarse surface normal maps n_i^* and ground-truth normal maps n_i^{gt}. On the other hand, l_{normal} calculates the divergence between fusion normal maps $n_i^{F_l}$ and ground-truth normal maps n_i^{gt}. There are a lot of step edges such as texture crosshatches or object boundaries [39] in natural RGB images, which inevitably interfere with the accuracy of depth estimation. It is necessary to prevent such interference so as to deal with distorted or blurry problems around step edges. The l_{grad} function defined by Hu et.al. [9] was adopted in this paper in order to penalize depth errors between neighboring pixels of depth images,

$$l_{grad} = \frac{1}{M}\sum_{i=1}^{n}(\mathrm{F}(\nabla_x(\tau_i)) + \mathrm{F}\big(\nabla_y(\tau_i)\big)) \tag{8}$$

where $\tau_i = \| d_i^* - d_i^{gt} \|_1$, $\nabla_x(\tau_i)$ and $\nabla_x(\tau_i)$ represent spatial derivation of τ_i along the x and y directions, respectively. $F(x)$ stands for a log algorithm function of depth errors defined by Hu [30].

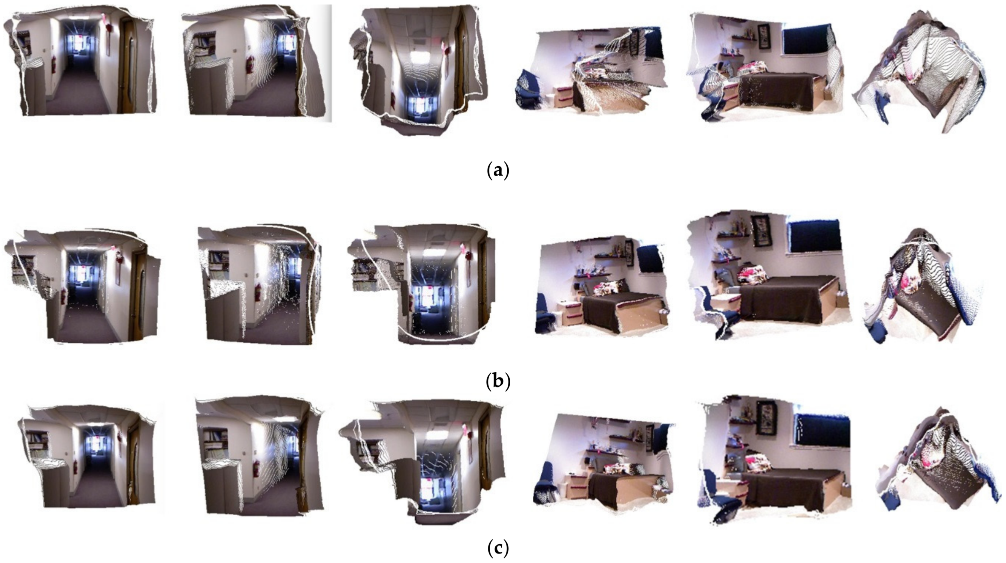
To be concluded, we defined a total loss denoted as l_{total}

$$l_{total} = \lambda_1\, l_{depth} + \lambda_2\, l_{grad} + \lambda_3\, l_{normal} \tag{9}$$

where l_{depth} represents pixel-wise loss for depth values, l_{grad} represents loss function for step edges, and l_{normal} represents function for surface normal. λ_1, λ_2, and λ_3 stand for initialed weights working for balancing effects of back propagation.

3.6. Recovering 3D Features from Estimated Depth

As is often the case [10,21], 3D space geometric constraints contribute to the capability of deep network in terms of depth estimation and corresponding 3D point cloud reconstruction. Therefore, the 3D point cloud maps shown in Figure 7 were reconstructed following Li et al. [40] to visualize the quality of depth maps predicted by our scheme.



(**a**)

(**b**)

(**c**)

Figure 7. Comparison of point clouds from estimated depth maps between Alhashim [6] and ours. (**a**) Alhashim [6], (**b**) GT, (**c**) Ours. GT stands for 3D point cloud maps from ground truth images.

We compare the 3D point-cloud maps produced by our scheme based on Densenet-121 with that of Dense-depth [6], which is based on Densenet-169 and does not introduce geometric constrictions.

As shown in Figures 7 and 8, our scheme outperforms Dense-depth [6] in terms of depth estimation results and the average quality of the relative 3D point-cloud maps. Therefore, the guidance of surface normal will definitely improve the quality of depth maps in terms of features in 3D space. What is more, better 3D point cloud maps can be recovered from depth maps with constriction of geometric features. As a result, the quality of point-cloud maps should also be adopted as a fundamental metric for evaluating the accuracy of depth estimation.

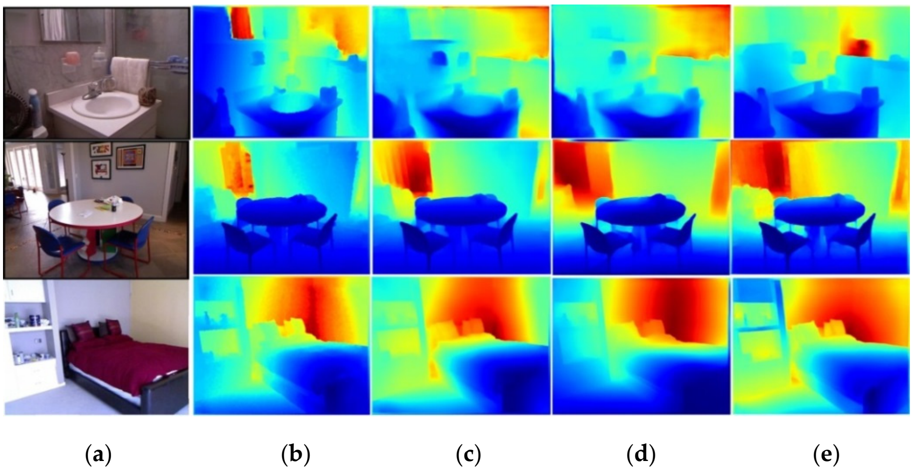


Figure 8. Qualitative results for depth estimation. (**a**) RGB images, (**b**) ground truth (GT), (**c**) Alhashim [2], (**d**) Our Coarse; (**e**) Refined Depth.

4. Experiments

In this section, we describe the implement details of experiments, evaluate the performance of our depth estimation scheme on NYU-Depth-V2 [14], and compare the prediction results against existing state-of-the-art (SOTA) methods. Moreover, we present results of ablation experiment to analyze the influence of the different parts of our proposed method.

4.1. Dataset

The NYU-Depth-V2 dataset [14] contains 407 K frames taken from 464 different indoor scenes, which were split into 249 training scenes and 215 testing scenes. Specially, 1449 RGB images were accurately labeled with depth images, in which 654 images are annotated for testing phase and others for training phase. All images were collected from videos captured by Kinect RGB-D sensor produced by Microsoft in Redmond, WA, the U.S. NYU-Depth-V2 widely serves as training datasets for supervised monocular depth prediction due to its accurate ground-truth (GT) depth labels and abundant image samples.

In some previous studies [6,9,10,20], depth estimation networks were trained on subsets sampled from the NYU dataset [14]. In practice [10,20], 30 K training frames were sampled from the raw NYU dataset [14]. The fewer frames used, the better-designed the deep learning schemes [4,6], which outperformed SOTA practices [7,21,41] that were trained on the entire NYU dataset. Moreover, the practice by Hu et al. [9] proved that the models are trained on subsets with more frames performing slightly better, but gains in accuracy did not justify the lower learning efficiency and higher system latency. Therefore, instead of the official splits image set, a subset with 30 K frames was used in this paper. All frames were randomly sampled from 249 training subsets.

The NYU dataset [14] does not provide ground-truth surface normal maps (N^{GT}); previous studies [10,21] computed N^{GT} from in-pained depth images, and thus, the quality of the produced N^{GT} were up to the in-painting algorithm proposed by Levin [35]. Instead of the in-painting method, the method proposed by Hickson et al. [42] was leveraged for obtaining N^{GT} in this paper. As for confidence map network, we utilized accurate binary mask images (I_M) offered by Eigen et al. [21].

4.2. Implementation Details

We implemented both the coarse depth estimation (CDE) network and the RGB-D surface normal network (RSN) using the deep learning platform PyTorch by Huang et al. [43] operating on a Tesla M40 GPU produced by NVIDA in Santa Clara, CA, the U.S. with 12 GB capacity.

The encoder of the CDE network was designed based on the Densenet-121 model, which initializes weights for decoder layers. Specially, the last classification layers of the Densenet-121 model were removed.

The Adam [44] optimizer was adopted for training the CDE network, and learning rate was set as 0.0001. The raw RGB images with resolution of $H \times W$ were down-sampled to $H/2 \times W/2$ for boosting training processes and fitting the size of output depth images. We conducted the coarse depth estimation training phase with a batch size of six for 20 epochs. A small subset with 654 samples [14] was employed for testing performance of CDE network, and the batch size for testing phase was set as 32. The network finally produced depth images with resolution of $H/2 \times W/2$ and an error evaluation index for supervising the training algorithm.

In order to avoid interference from over-fitting, four augmentation methods were employed following Cubuk et al. [45] for depth estimation training phase:

(1) The horizontal mirroring operation was applied to both RGB and depth images with a probability of 25%.

(2) Much rotation leads into invalid data for GT depth images [6], so input images in training process were rotated by slight degrees, which ranged from −2 to 2 with a probability of 25%.

(3) Contrast and brightness values of RGB images input were randomly scaled by (0.6, 1.2) with a probability of 50%.

(4) Both RGB and depth images were randomly resized to 320 × 256 with a probability of 50%.

The training process of surface normal was also performed on NYU Depth-V2 [14] and the number of training epochs was also set as 20 with batch size of six. The testing image-set for RSN Network consists of 654 sample RGB images and corresponding ground truth surface normal images produced following the practice by Hickson et al. [42]. For optimizer, the Adam [44] was selected with original learning rate of 1×10^{-4}, initial parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$ and weight decay of 1×10^{-4}. The RSN network also produced surface normal images with the resolution of $H/2 \times W/2$.

The refinement network functioned as the convolution kernel function [10], which does not demand any training process.

In all training experiments, weight ω_1 for l_{depth} was set as 0.5 for balancing the importance of coarse depth estimation and depth refinement. Weight ω_2 for l_{grad} was set as $\omega_2 = 0.5$ to balance the influence of two terms. As is often the case [6,7,9,10,20], the hyper-parameters are empirically set as a reasonable value for loss functions. In our study, parameter ω_2 was set as 0.5 according to validations on an officially annotated subset with 1449 images. Weight μ_l for l_{normal} was set as $\mu_l = 0.2l$, where $l = \{1,2,3,4\}$. The parameter l was defined in Section 3.4, which stands for the scale value of images fed to the RSN network.

As shown in Equations (6)–(8), l_{depth} calculates the depth value errors for each pixel, l_{normal} calculates angular cosine errors for each pixel, and l_{grad} calculates the gradient errors in the log domain. Therefore, the value of l_{depth} will be obviously bigger than l_{normal} and l_{grad}. To mitigate this effect, the parameter λ_1 is set to a small value. According to test results on the annotated small subset, parameters $\{\lambda_1, \lambda_2, \lambda_3\}$ were set as reasonable weights $\{ 0.2, 1, 1 \}$.

As the proceeding of training process, the model gradually converges. The total number of training parameters for entire network based on Densenet-121 model was approximately 32.4 M. Training was performed for 600 K iterations on NYU Depth-V2 [14], which took 18 h to finish. With repeated inference learning, the value of loss functions l_{grad}, l_{normal}, and l_{depth} would converge to zero.

4.3. Evaluation Criteria

For quantitatively evaluation, three error metrics were used by the previous work of Eigen et al. [46]: absolute relative error (*AbsRel*), root mean squared error (*RMSE*), and average (Log_{10}) error $\left(E_{Log_{10}}\right)$. Moreover, threshold accuracy (T_{re}) was selected as the accuracy metric. All metrics can be defined as:

$$max\left(\frac{D_i^{GT}}{D_i'}, \frac{D_i'}{D_i^{GT}}\right) = \delta < T_{re} \tag{10}$$

$$AbsRel = \frac{1}{S}\sum_{i=1}^{S}\frac{\left|D_i^{GT} - D_i'\right|^2}{D_i^*} \tag{11}$$

$$RMSE = \sqrt{\frac{1}{S}\sum_{i=1}^{S}\left|D_i^{GT} - D_i'\right|^2} \tag{12}$$

$$E_{Log_{10}} = \frac{1}{S}\sum_{i=1}^{S}\left|Log_{10}D_i^{GT} - Log_{10}D_i'\right| \tag{13}$$

As is shown in the above metrics, we denote D_i^{GT} as the ground truth depth corresponding to pixel i, D_i' as the relative estimated depth, and S as the total number of pixels with available value in ground truth maps. Here, three different thresholds (δ, δ^2, δ^3) are set as (1.25, 1.25^2, 1.25^3) according to conventional works [2,46].

Three error metrics [21,27] were used for evaluating surface normal maps in this paper: mean of angle error (*Mean*), medians of angle error (*Median*), and root mean square error (*RMSE*). Moreover, three different thresholds (11.25°, 22.5°, 30°) [21] were used for calculating the specific angular error of pixels.

4.4. Benchmark Performance Comparison

We select the SOTA practice Dense-depth [6] for comparison, which adopted Densenet-169 [5] as the encoder of deep network. The pre-trained model functioned by extracting depth feature from RGB image input. As shown in Figure 8, Dense-depth outperforms the coarse depth estimation network based on Densenet-121 [5] proposed in this paper, while performs worse than our entire depth estimation scheme containing the surface normal guidance. Some depth estimation samples are shown in Figure 9, and more depth maps from our framework are listed in the Appendix A. All images are colorized for better visualization.

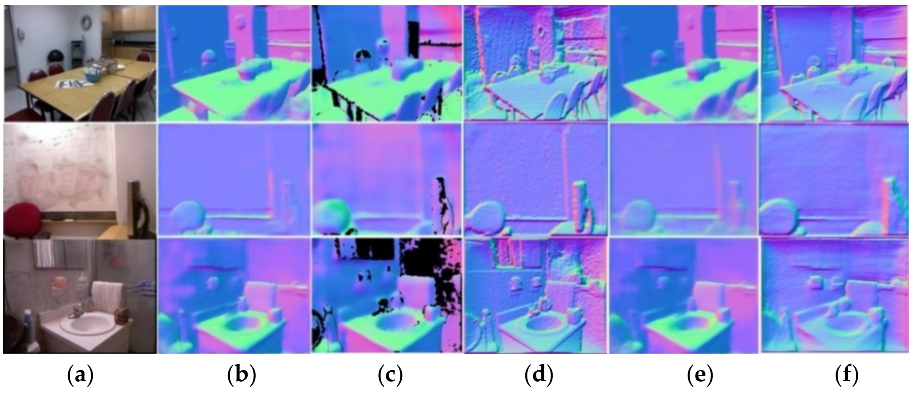


Figure 9. Qualitative results for surface normal estimation. (**a**) RGB images, (**b**) ground truth normal maps, (**c**), Geo-Net [10], (**d**) reconstructed from in-painted ground truth depth (**e**) ours, and (**f**) reconstructed from refined depth. All images are equally scaled for better visualization.

Table 1 lists depth estimation results generated by proposed method and previous masterpieces [6,7,9,21]. The quality of depth maps can be evaluated according to Equations (10)–(13).

Table 1. Comparison of depth estimation accuracy with previous methods on NYU Depth-V2 [14].

Method	**Error (Lower Is Better)**			**Accuracy (Higher Is Better)**		
	AbsREL.	**RMSE.**	$\mathbf{E_{Log10}}$**.**	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Eigen [21]	0.158	0.641	-	0.769	0.950	0.988
Fu [7]	0.115	0.509	0.051	0.828	0.965	0.992
Alhashim [6] (Densenet-169)	0.123	0.465	0.053	0.846	0.974	0.994
Hu [9] (Densenet-161)	0.123	0.544	0.053	0.855	0.972	0.993
Ours (Densenet-121-coarse)	0.137	0.572	0.056	0.839	0.962	0.988
Ours (Densenet-121-refined)	0.122	0.459	0.051	0.859	0.972	0.993

According to results shown in Table 1 and Figure 8, the refined depth images have a better geometric quality than that of Dense-depth, which proves the necessity of surface normal guidance in monocular depth prediction tasks.

Table 2 compares the surface normal calculation results based on RGB image methods [33,37], surface normal with depth consistency method [10], and the RGB-D fusion method (ours). From Table 2, it can be seen that the surface normal scheme leveraging both depth and RGB image features is averagely superior to the schemes that employ depth images or RGB image only.

Table 2. Comparison of surface normal accuracy with previous methods on NYU Depth-V2 [14].

Method	**Error (Lower Is Better)**			**Accuracy (Higher Is Better)**		
	Mean.	**Median.**	**RMSE.**	11.25°	22.5°	30°
3DP(MW) [37]	36.3	19.2	46.6	39.2	52.9	57.8
Wang [22]	26.9	14.8	-	42.0	61.2	68.2
Qi [10]	19.0	11.8	26.9	48.4	71.5	79.5
Ours (RGB-D fusion)	20.6	11.0	25.6	47.9	73.2	81.8

Instead of directly comparing errors between pixel values, Table 2 compares angular difference between orientation vectors for each pixel. The qualitative performance of previous research [10,22,37] is cited from the original papers. As listed in Table 2, with the aid of high-order geometric RGB-D fusion, the surface normal maps from our method outperforms Geo-Net [10]. From Table 2 and Figure 8, we can conclude that the method proposed in this paper can recover better shape from RGB images, which contributes to supply more accurately geometric details with depth images.

4.5. Computational Performance

Table 3 compares the computational efficiency of proposed depth estimation algorithm with that of state-of-the-art (SOTA) methods [6,7,9]. It is seen in Table 3 that our model achieves SOTA results on the RMSE metric. What is more, our model requires fewer training parameters, fewer training iterations, and fewer image samples in terms of the same training epochs. Furthermore, our model consumed less training time while operating on similar platforms.

Table 3. Comparison of computational efficiency and performance on the NYU dataset [14]. The "Parm.'"stands for training parameters, and "Iters." represents the number of training iterations.

Method	**RMSE.**	**Frames**	**Epochs**	**Training Time (h)**	**Iters.**	**Inference Time (s)**	**Parms.**
Fu [7]	0.509	120K	-	-	3M	-	110M
Alhashim [6]	0.465	50K	20	20	1M	0.265	42.6M
Hu [9] (Se-Net)	0.530	50K	20	-	1M	0.352	-
Ours	0.459	30K	20	18	600k	0.217	32.4M

In additional, we tested our pre-trained model and testing models released by references [6,9] on an Tesla M40 produced by NVIDA in Santa Clara, CA, the U.S. with single 12G memory capacity. As shown in Table 3, our model performed lower latency and higher accuracy.

All the data shown in Table 3 were cited from the original paper and corresponding released models [6,9].

4.6. Ablation Study

In this section, ablation studies are performed to verify each part of the proposed architecture in terms of performance on depth estimation.

In this experiment, the DenseNet-121 model was substituted with pre-trained DenseNet-161 both in the coarse depth estimation (CDE) network and the RGB-D surface normal (RSN) network. As shown in Table 4, the network based on the DenseNet-161 model [5] outperforms that of the DenseNet-121 model [5] in terms of qualitative accuracy metrics. However, according to the training parameters listed in Table 4, the growth of encoding layers in deep learning deep structure will introduce superfluous training costs.

Table 4. Ablation study of encoder layers. Parm. stands for training parameters; DN stands for Dense-net.

Method	**Parm.**	**Error (Lower Is Better)**			**Accuracy (Higher Is Better)**		
		Abs-REL	**RMSE.**	**E_{Log10}.**	**$\delta < 1.25$**	**$\delta < 1.25^2$**	**$\delta < 1.25^3$**
DN-121-Refined	32.4 M	0.122	0.459	0.051	0.859	0.972	0.993
Dense-depth [2] (DN-169)	42.6 M	0.123	0.465	0.053	0.846	0.974	0.994
DN-161-Refined	67.2 M	0.116	0.446	0.049	0.867	0.976	0.994

Furthermore, as shown in Table 4, with the geometric constrictions from surface normal maps, the CDE network based on Densenet-121 model outperforms Dense-depth [6], whose decoder was designed based on deeper-model Densnet-169.

In this experiment, Mobilenet-V2 [47] was adopted as the backbone of depth prediction and surface normal estimation, which utilizes less training weight parameters and lower computational complexity than the Densenet-121 model. As shown in Table 5, the network based on Mobilenet-V2 apparently required less training cost and thus performed lower latency. However, Mobilenet-V2 based deep learning scheme produced worse depth maps than Densenet-121 model [5] at the same time. Therefore, Mobilenet-V2 [47] model-based networks can be conveniently embedded into simple mobile platforms such as a mobile phone and lite drone. The detailed structure of coarse depth estimation network based on Mobilenet-V2 [47] model was shown in Appendix B.

Table 5. Ablation study of encoder layer numbers. Parm. stands for training parameters.

Method	**Parm.**	**Error (Lower Is Better)**			**Accuracy (Higher Is Better)**		
		AbsREL	**RMSE.**	**E_{Log10}.**	**$\delta < 1.25$**	**$\delta < 1.25^2$**	**$\delta < 1.25^3$**
Ours(121-refined)	32.4 M	0.122	0.459	0.051	0.859	0.972	0.993
Ours(V2-Refined)	4.7 M	0.196	0.519	0.146	0.811	0.951	0.979

In this experiment, up-projection units in decoder layers of the RGB-D surface normal (RSN) network was substituted with the 2× bilinear up-sampling and up- and down-projection unit [48], respectively, for ablative comparison.

The up-projection unit [17] was designed for embed platforms. As shown in Figure 10, it achieves the lowest latency in this experiment. Therefore, it is suitable to be employed as the up-sampling strategy of the RSN network for boosting surface normal inference process.

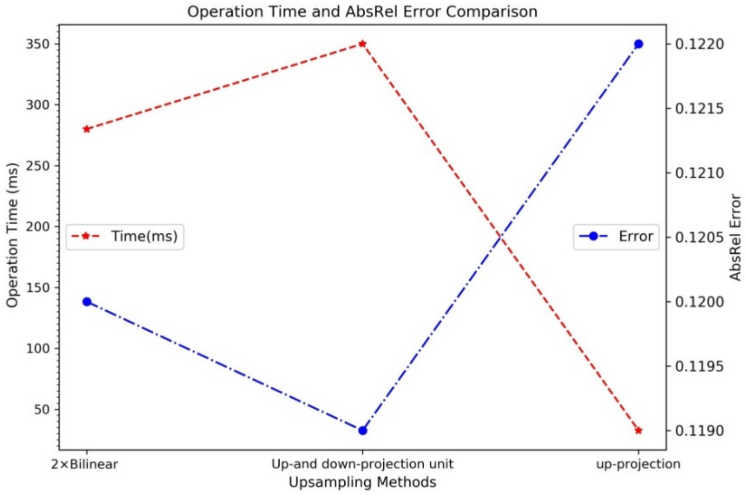


Figure 10. Comparison of time consumption: Runtimes of different up-sampling methods including. 2× bilinear interpolation [6], up and down projection [48], and up-projection [17].

The up- and down-projection unit applies super-resolution (SR) techniques to up-scaling the input image to a higher resolution. As shown in Figure 10, the value of absolute relative error (*AbsRel*) suggests that the refined depth maps obtained by the up and down projection unit slightly outperforms 2× bilinear interpolation [6]. When using the up- and down-projection projection unit [48] in decoder layers, we found that the gains in performance did not justify the slow learning time and the extra GPU memory required. Therefore, the 2× bilinear up-sampling proposed by Alhashimet al. [6] functioned as an up-sampling method in our coarse depth estimation network.

4.7. Custom Results

To verify the deep learning network proposed herein, we took some videos of different indoor scenes with the monocular camera of a smart phone. Then, we randomly captured some RGB images from videos and resized them as the resolution of 640 × 480 for depth estimation.

As shown in Figure 11, depth maps predicted from our network perform distinct boundary and robust geometric shapes.

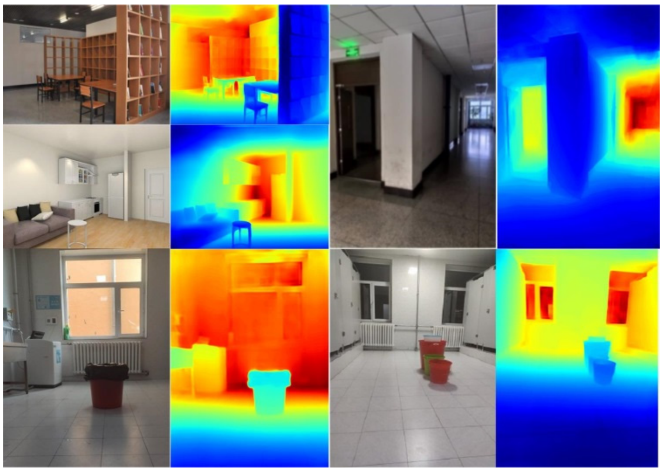


Figure 11. Refined depth images generating from custom images (densenet-161 model).

5. Conclusions

In this work, we designed a lighter-weight encoder–decoder deep learning network for depth estimation from monocular RGB images. The encoder layers were designed based on pre-trained deep learning model originally for image classification. Experiments proved that an effective encoder based on transfer learning and geometric guidance outperforms previous methods [7,49] employing complex feature extraction layers. Ablation studies suggested that employing different pre-trained models enabled our networks adapt to different platforms. For example, Mobilenet-V2 [3] can be a suitable model for simple mobile platforms such as smart phones and light micro drones, the Densenet-121 model can be used for mobile platforms equipped with a powerful GPU device, while a denser decoder based on Densenet161 can only be applied to platforms that can afford expensive training costs.

What is more, surface normal estimation was introduced to improve the quality of depth images. Benefiting from geometric guidance offered by surface normal maps, our network obviously achieved better depth images on benchmark NYU depth V2 [14] than state-of-the-art methods [4,7,9]. With the geometric constrictions from surface normal maps, superb 3D point-cloud maps were reconstructed from refined depth images.

Because this work greatly benefits from surface normal estimation, we believe that there are still many other possible geometric features that can be used to improve monocular depth estimation. Therefore, we will further study the effects of geometric features such as 3D objects boundary, semantic label, and image defocus.

Further research will also focus on how to apply pre-trained depth estimation network to 3D vision practices, such as augment reality (AR), simultaneous localization and mapping (SLAM), and structure from motion (SFM).

Author Contributions: Conceptualization, K.H. and X.Q.; Methodology, W.Z.; Software, K.H.; Validation, S.C., K.H. and W.Z.; Formal analysis, X.Q.; Data duration, F.Z., Z.C.; Writing—original draft preparation, K.H., S.C., and W.Z.; Writing—review and editing, K.H., H.Q., S.C., W.Z., and X.Q.; Funding acquisition, W.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: The authors want to express gratitude to the editor and anonymous reviewers for their professional comments and valuable suggestions.

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

Appendix A

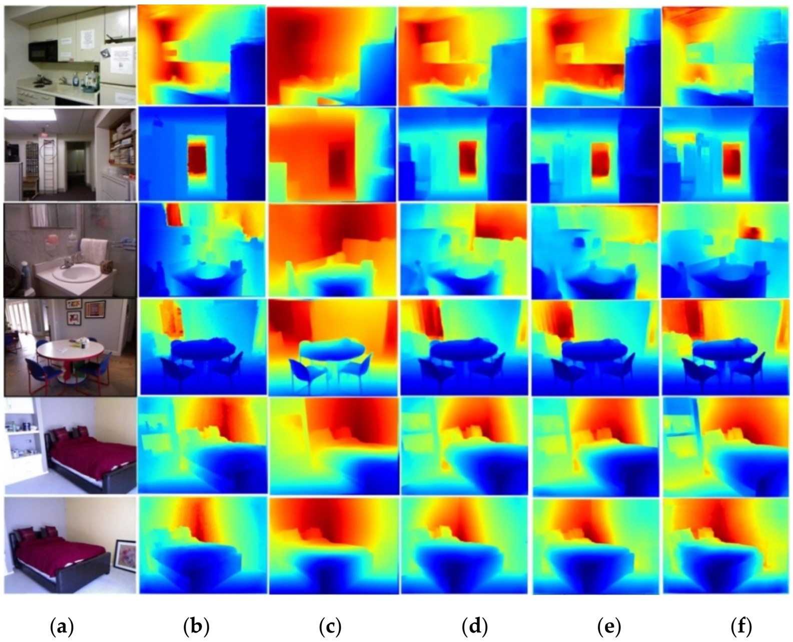


Figure A1. Results of different methods on NYU V2 Depth. (**a**) Red-green-blue (RGB) images, (**b**) ground truth (GT), (**c**) Ranftl [29], (**d**) Hu [30], (**e**) Alhashim [3], (**f**) ours.

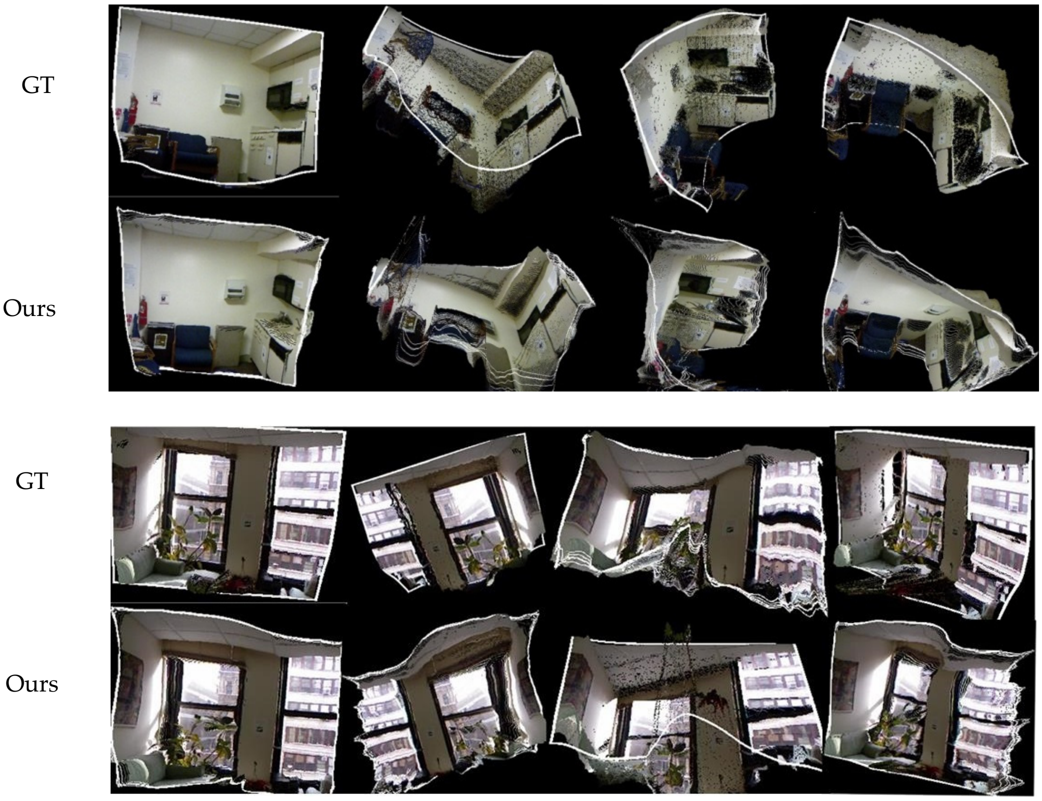


Figure A2. Three-dimensional point cloud maps comparison on NYU Depth V2.

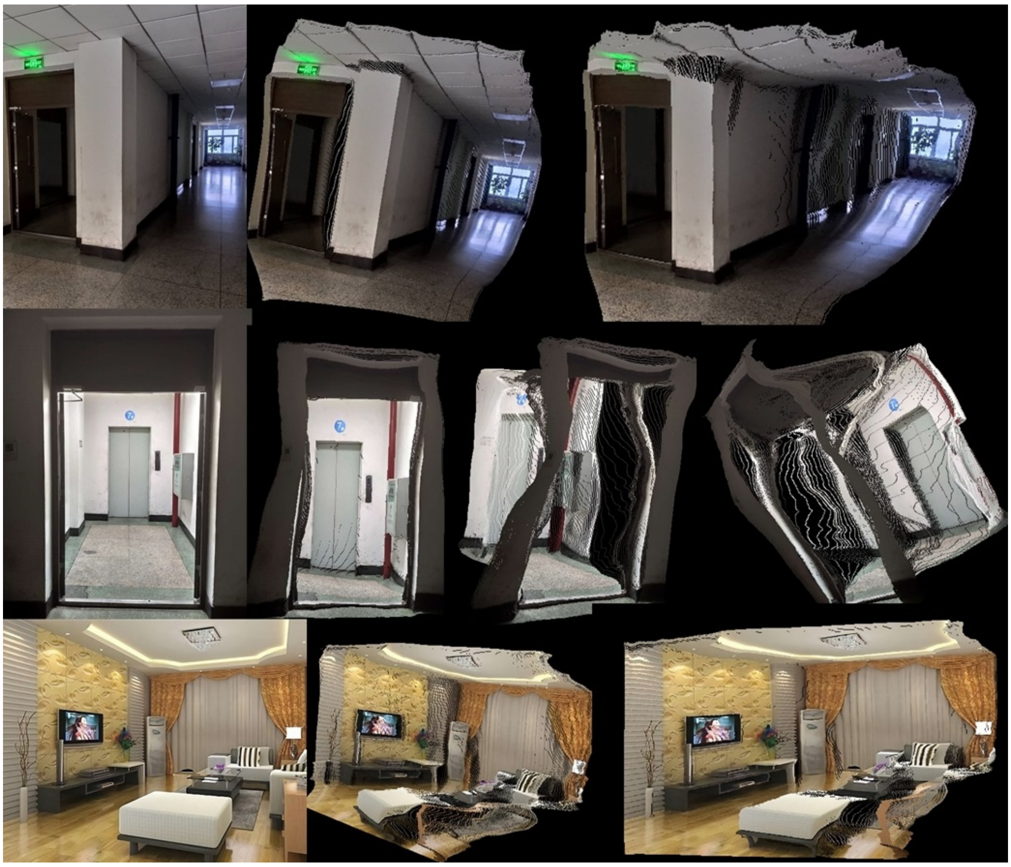


Figure A3. Point cloud maps were reconstructed based on custom images.

Appendix B

Table A1. Details of coarse depth estimation architecture based on Densenet-161.

Layer	Output	Operator
Input	640 × 480 × 3	-
Convolution1 (CONV. 1)	320 × 240 × 96	CONV. 7 × 7
DenseBlock1 [26] (DB.1)	160 × 120 × 96	Avg. Pooling 2 × 2
DB2	80 × 60 × 192	Avg. Pooling 2 × 2
DB3	40 × 30 × 384	Avg. Pooling 2 × 2
DB4	20 × 15 × 1056	Avg. Pooling 2 × 2
CONV. 2	20 × 15 × 2208	CONV. 1 × 1
Bottle Layer	20 × 15 × 1024	Batch-Normal + ReLU + CONV. 1 × 1
UP-Sample1 (US1)	40 × 30 × 1024	Up-sample 2 × 2
Concatenating 1	40 × 30 × 1408	Concatenating DB3 Avg. Pooling
US1-CONV. 1 [2]	40 × 30 × 512	CONV. 3 × 3
US1-CONV. 2	40 × 30 × 512	CONV. 3 × 3
UP-Sample2 (US2)	80 × 60 × 512	Up-sample 2 × 2
Concatenating2	80 × 60 × 704	Concatenating DB2 Avg. Pooling
US2-CONV. 1	80 × 60 × 256	CONV. 3 × 3
US2-CONV. 2	80 × 60 × 256	CONV. 3 × 3
UP-Sample3 (US3)	160 × 120 × 256	Up-sample 2 × 2
Concatenating3	160 × 120 × 352	Concatenating DB1 Avg. Pooling
US3-CONV. 1	160 × 120 × 128	CONV. 3 × 3
UP3-CONV. 2	320 × 240 ×128	CONV. 3 × 3
UP-Sample4 (US4)	320 × 240 × 128	Up-sample 2 × 2
Concatenating4	320 × 240 × 224	Concatenating CONV. 1 Avg. Pooling
US4-CONV. 1	320 × 240 × 64	CONV. 3 × 3
US4-CONV. 2	320 × 240 × 64	CONV. 3 × 3
CONV. 3	320 × 240 ×1	CONV. 3 × 3
Output	240 × 320 × 1	-

Table A2. Details of coarse depth estimation architecture based on Mobilenet-V2.

Layer	Output	Operator
INPUT	640 × 480 × 3	-
Convolution1 (CONV.1)	320 × 240 × 32	CONV. 3 × 3
Inverted Residual [3] Block1 (IRB1)	320 × 240 × 16	Bottleneck × 1
Inverted Residual Block2 (IRB2)	160 × 120 × 24	Bottleneck × 6
Inverted Residual Block3 (IRB3)	80 × 60 × 32	Bottleneck × 6
Inverted Residual Block4 (IRB4)	80 × 60 × 64	Bottleneck × 6
Inverted Residual Block5 (IRB5)	40 × 30 × 96	Bottleneck × 6
Inverted Residual Block6 (IRB6)	40 × 30 × 160	Bottleneck × 6
Inverted Residual Block7 (IRB7)	20 × 15 × 320	Bottleneck × 6
CONV. 2	20 × 15 × 1280	CONV. 1 × 1
CONV. 3	20 × 15 × 768	CONV. 1 × 1
UP-Sample1 (US1)	40 × 30 × 768	Up-sample 2 × 2
Concatenating1	40 × 30 × 928	Concatenating IRB6
US1-CONV. 1	40 × 30×384	CONV. 3 × 3
US1-CONV. 2	40 × 30 × 384	CONV. 3 × 3
UP-Sample2 (US2)	80 × 60 × 384	Up-sample 2 × 2
Concatenating2	80 × 60 × 448	Concatenating IRB4
US2-CONV. 1	80 × 60 × 192	CONV. 3 × 3
US2-CONV. 2	80 × 60 × 192	CONV. 3 × 3
UP-Sample3 (US3)	160 × 120 × 192	Up-sample 2 × 2
Concatenating3	160 × 120 × 216	Concatenating IRB2
US3-CONV. 1	160 × 120 × 96	CONV. 3 × 3
UP3-CONV. 2	160 × 120 × 96	CONV. 3 × 3
UP-Sample4 (US4)	320 × 240 × 96	Up-sample 2 × 2
Concatenating4	320 × 240 × 112	Concatenating IRB1
US4-CONV. 1	320 × 240 ×64	CONV. 3 × 3
US4-CONV. 2	320 × 240 × 64	CONV. 3 × 3
CONV. 4	320 × 240 × 1	CONV. 1 × 1
Output	320 × 240 × 1	-

References

1. Guizilini, V.; Ambrus, R.; Pillai, S.; Gaidon, A. Packnet-sfm: 3d packing for self-supervised monocular depth estimation. *arXiv* **2019**, arXiv:1905.02693.
2. Ummenhofer, B.; Zhou, H.; Uhrig, J.; Mayer, N.; Ilg, E.; Dosovitskiy, A.; Brox, T. Demon: Depth and motion network for learning monocular stereo. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 5622–5631.
3. Zhou, H.; Ummenhofer, B.; Brox, T. Deeptam: Deep tracking and mapping. In Proceedings of the European Conference on Computer Visio (ECCV), Munich, Germany, 8–14 September 2018; pp. 822–838.
4. Yu, C.; Liu, Z.; Liu, X.; Xie, F.; Yang, Y.; Wei, Q.; Fei, Q. DS-SLAM: A Semantic Visual SLAM towards Dynamic Environments. In Proceedings of the 2018 IEEE International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 1168–1174.
5. Huang, G.; Liu, Z.; Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2261–2269.
6. Alhashim, I.; Wonka, P. High quality monocular depth estimation via transfer learning. *arXiv* **2018**, arXiv:1812.11941.
7. Fu, H.; Gong, M.; Wang, C.; Batmanghelich, K.; Tao, D. Deep ordinal regression network for monocular depth estimation. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, 18–22 June 2018; pp. 2002–2011.
8. Chen, L.; Tang, W.; Wan, T.R.; Nigel, W.J. Self-supervised monocular image depth learning and confidence estimation. *arXiv* **2018**, arXiv:1803.05530. [CrossRef]
9. Hu, J.J.; Ozay, M.; Zhang, Y.; Okatani, T. Revisiting Single Image Depth Estimation: Toward Higher Resolution Maps with Accurate Object Boundaries. *arXiv* **2019**, arXiv:1803.08673.
10. Qi, X.; Liao, R.; Liu, Z.; Urtasun, R.; Jia, J. GeoNet: Geometric Neural Network for Joint Depth and Surface Normal Estimation. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, 18–22 June 2018; pp. 283–291.
11. Zhang, Z.; Cui, Z.; Xu, C.; Yan, Y.; Sebe, N.; Yang, J. Pattern-affinitive propagation across depth, surface normal and semantic segmentation. In Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 4106–4115.
12. Lee, J.-H.; Kim, C.-S. Monocular Depth Estimation Using Relative Depth Maps. In Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 9729–9738.
13. Chen, L.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 834–848. [CrossRef] [PubMed]
14. 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, Florence, Italy, 7–13 October 2012; pp. 746–760.
15. Dai, A.; Chang, A.X.; Savva, M.; Halber, M.; Funkhouser, T.; Nießner, M. ScanNet: Richly annotated 3D reconstructions of indoor scenes. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2432–2443.
16. Chang, A.; Dai, A.; Funkhouser, T.; Halber, M.; Niessner, M.; Savva, M.; Song, S.; Zeng, A.; Zhang, Y. Matterport3D: Learning from RGB-D data in indoor environments. In Proceedings of the International Conference on 3D Vision (3DV), Qingdao, China, 10–12 October 2017; pp. 667–676.
17. Wofk, D.; Ma, F.C.; Yang, T.J.; Karaman, S.; Vivienne, S. FastDepth: Fast Monocular Depth Estimation on Embedded Systems. *arXiv* **2019**, arXiv:1903.03273.
18. Nekrasov, V.; Shen, C.H.; Reid, I. Light-Weight RefineNet for Real-Time Semantic Segmentation. *arXiv* **2018**, arXiv:1810.03272.
19. Zeng, J.; Tong, Y.; Huang, Y.; Yan, Q.; Sun, W.; Chen, J.; Wang, Y. Deep Surface Normal Estimation With Hierarchical RGB-D Fusion. In Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 6146–6155.

20. Yin, W.; Liu, Y.; Shen, C.; Yan, Y. Enforcing Geometric Constraints of Virtual Normal for Depth Prediction. In Proceedings of the 2019 IEEE International Conference on Computer Vision, Seoul, Korea, 27–28 October 2019; pp. 5683–5692.
21. Eigen, D.; Fergus, R. Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-scale Convolutional Architecture. In Proceedings of the 2015 IEEE International Conference on Computer Vision, Santiago, Chile, 13–16 December 2015; pp. 2650–2658.
22. Wang, X.; Fouhey, D.; GuptaIn, A. Designing deep networks for surface normal estimation. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 8–10 June 2015; pp. 539–547.
23. Garg, R.; BG, V.K.; Carneiro, G.; Reid, I. Unsupervised CNN for single view depth estimation: Geometry to the rescue. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; pp. 740–756.
24. Wang, C.; Miguel Buenaposada, J.; Zhu, R.; Lucey, S. Learning depth from monocular videos using direct methods. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, 18–22 June 2018; pp. 2022–2030.
25. Yang, Z.; Wang, P.; Wang, Y.; Xu, W.; Nevatia, R. Every Pixel Counts: Unsupervised Geometry Learning with Holistic 3D Motion Understanding. *arXiv* **2018**, arXiv:1806.10556.
26. Zhou, T.H.; Brown, M.; Noah, S.; David, L. Unsupervised Learning of Depth and Ego-Motion from Video. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 6612–6619.
27. Li, R.; Wang, S.; Long, Z.; Gu, D. UnDeepVO: Monocular Visual Odometry through Unsupervised Deep Learning. *arXiv* **2017**, arXiv:1709.06841.
28. Liu, F.; Chunhua, S.; Guosheng, L. Deep convolutional neural fields for depth estimation from a single image. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 5162–5170.
29. Lee, J.H.; Han, M.K.; Kong, D.W.; Suh, H. From Big to Small: Multi-Scale Local Planar Guidance for Monocular Depth Estimation. *arXiv* **2019**, arXiv:1907.10326.
30. Janai, J.; Guney, F.; Ranjan, A.; Black, M.; Geiger, A. Unsupervised Learning of Multi-Frame Optical Flow with Occlusions. In Proceedings of the European Conference on Computer Visio (ECCV), Munich, Germany, 8–14 September 2018; pp. 713–731.
31. Misra, I.; Shrivastava, A.; Gupta, A.; Hebert, M. Cross-stitch networks for multitask learning. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 3994–4003.
32. Kwak, D.-H.; Lee, S.-H. A Novel Method for Estimating Monocular Depth Using Cycle GAN and Segmentation. *Sensors* **2020**, *20*, 2567. [CrossRef] [PubMed]
33. Xu, D.; Ouyang, W.; Wang, X.; Sebe, N. PAD-Net: Multi-tasks Guided Prediction and Distillation Network for Simultaneous Depth Estimation and Scene Parsing. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, 18–22 June 2018; pp. 675–684.
34. Jiao, J.B.; Cao, Y.; Song, Y.B.; Lau, R. Look Deeper into Depth: Monocular Depth Estimation with Semantic Booster and Attention-Driven Loss. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Cham, Switzerland, 2016; pp. 55–71.
35. Levin, A.; Lischinski, D.; Weiss, Y. Colorization using optimization. *ACM Trans. Graph.* **2004**, *23*, 689–694. [CrossRef]
36. Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; Li, F.-F. ImageNet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), Miami, FL, USA, 20–25 June 2009.
37. Fouhey, D.F.; Gupta, A.; Hebert, M. Data-driven 3d primitives for single image understanding. In Proceedings of the 2013 International conference on computer vision, Sydney, Australia, 1–8 December 2013; pp. 3392–3399.
38. Sun, L.H.; Wang, J.S.; Yun, H.; Zhu, Q.; Yin, B.C. Surface Normal Data Guided Depth Recovery with Graph Laplacian Regularization. In Proceedings of the 2019 ACM Multimedia Asia (MMAsia ’19), Beijing, China, 16–18 December 2019; pp. 1–6.
39. Huang, J.; Lee, A.B.; Mumford, D. Statistics of range images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Hilton Head, SC, USA, 13–15 June 2000; pp. 324–331.

40. Xiaoyang, L. Depth2pointCloud. Available online: https://github.com/ZJULiXiaoyang/depth2pointCloud (accessed on 19 July 2020).
41. Laina, I.; Rupprecht, C.; Belagiannis, V.; Tombari, F.; Navab, N. Deeper Depth Prediction with Fully Convolutional Residual Networks. In Proceedings of the 2016 Fourth International Conference on 3D Vision (3DV), Stanford, CA, USA, 25–28 October 2016; pp. 239–248.
42. Hickson, S.; Raveendran, K.; Fathi, A.; Murphy, K.; Essa, I. Floors are Flat: Leveraging Semantics for Real-Time Surface Normal Prediction. *arXiv* **2019**, arXiv:1906.06792.
43. Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z. Automatic differentiation in PyTorch. In Proceedings of the Advances in Neural Information Processing Systems Workshops, Long Beach, CA, USA, 4–9 December 2017; pp. 1–4.
44. Kingma, D.P.; Ba, J.L. Adam: A method for stochastic optimization. In Proceedings of the 2015 International Conference on Learning Representations, San Diego, CA, USA, 7–9 May 2015; pp. 1–41.
45. Cubuk, E.; Zoph, B.; Mane, D.; Vasudevan, V.; Le, Q.V. Autoaugment: Learning augmentation policies from data. *arXiv* **2018**, arXiv:1805.09501.
46. Eigen, D.; Puhrsch, C.; Fergus, R. Depth Map Prediction from a Single Image Using a Multi-scale Deep Network. In Proceedings of the 27th International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 2366–2374.
47. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, M.; Chen, L. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4510–4520.
48. Haris, M.; Shakhnarovich, G.; Ukita, N. Deep back-projection networks for super-resolution. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 1664–1673.
49. Ranftl, R.; Lasinger, K.; Hafner, D.; Schindler, K.; Koltun, V. Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-shot Cross-dataset Transfer. *arXiv* **2019**, arXiv:1907.01341.