IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 29, 2018, accepted January 18, 2019, date of publication January 23, 2019, date of current version February 12, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2894651

Super-Resolution for Monocular Depth Estimation With Multi-Scale Sub-Pixel Convolutions and a Smoothness Constraint

SHIYU ZHAO, LIN ZHANG, (Senior Member, IEEE), YING SHEN, (Member, IEEE), SHENGJIE ZHAO, (Senior Member, IEEE), AND HUIJUAN ZHANG

School of Software Engineering, Tongji University, Shanghai 201804, China

Corresponding author: Lin Zhang (cslinzhang@tongji.edu.cn)

This work was supported in part by the Natural Science Foundation of China under Grant 61672380, in part by the Fundamental Research Funds for the Central Universities under Grant 2100219068, and in part by the National Key Research and Development Project under Grant 2017YFE0119300.

ABSTRACT Depth estimation from a monocular image is of paramount importance in various vision tasks, such as obstacle detection, robot navigation, and 3D reconstruction. However, how to get an accurate depth map with clear details and a fine resolution remains an unresolved issue. As an attempt to solve this problem, we exploit image super-resolution concepts and techniques for monocular depth estimation and propose a novel CNN-based approach, namely $MSCN_{NS}$, which involves multi-scale sub-pixel convolutions and a neighborhood smoothness constraint. Specifically, $MSCN_{NS}$ makes use of sub-pixel convolutions with multi-scale fusions to retrieve a high-resolution depth map with fine details of the scene. Different from previous multi-scale fusion strategies, those multi-scale features come from supervised scale branches of the network. Furthermore, $MSCN_{NS}$ incorporates a neighborhood smoothness regularization term to make sure that spatially closer pixels with similar features would have close depth values. The effectiveness and efficiency of $MSCN_{NS}$ have been corroborated through extensive experiments conducted on benchmark datasets.

INDEX TERMS Monocular depth estimation, multi-scale feature fusion, neighborhood smoothness, sub-pixel convolution.

I. INTRODUCTION

Accurate depth information is vital to many computer vision tasks, such as scene understanding [1]–[3], 3D reconstruction [4], obstacle detection [5], etc. However, collecting depth is expensive or even impossible in some scenarios and in those cases, depth estimation from RGB images is required. Generally, stereo vision approaches [6]–[8] are kind of solutions for this task. Nevertheless, they require binocular images from two cameras with fixed relative locations and thus image quality has a great influence on the performance. Moreover, those methods are usually time consuming to get an accurate disparity map. Therefore, for data consisting of only monocular images, how to predict depth from a single still image becomes profoundly important. However, it is a very challenging task since one captured image may correspond to numerous real world scenes [9] and there are no reliable depth cues available, e.g. stereo correspondences or motions [10].

To handle such a problem, various solutions have been proposed in the literature. Primary methods [11]–[13] in this field usually formulated depth estimation as a Markov random field (MRF) learning problem and resorted to hand-crafted features, such as SIFT, GIST, PHOG, etc. Later, data-driven approaches [14], [15] were explored. Those approaches made use of hand-crafted features to retrieve the most similar candidates in the training set for a given query image. And then, the corresponding depth candidates were warped and fused to produce the final prediction. However, all these methods were usually designed for specific conditions and thus could only achieve reluctantly acceptable results.

With the emergence and popularity of CNNs (Convolutional Neural Networks), recently, researchers have begun exploring CNNs in the context of depth estimation and preliminary better results in terms of both the efficiency and the accuracy have been achieved. Inspired by the

great success already achieved along this direction, in this paper, focusing on how to further explore deep models together with super-resolution concepts and techniques for solving the depth estimation problem, we propose a CNN-based approach with multi-scale sub-pixel convolutions and a neighborhood smoothness constraint, namely $MSCN_{NS}$ (Multi-scale Sub-pixel Convolutional Network with a Neighborhood Smoothness constraint). Specifically, we generate multi-scale features from different branches of our network and fuse those features in several sub-pixel convolutions to get fine details in a high resolution prediction. Moreover, we further improve our model with a neighborhood smoothness constraint as a regularization term during the training phase. Through extensive experiments, we demonstrate that the proposed method is able to outperform state-of-the-art methods in various metrics on benchmark datasets and runs much faster, as well. Besides, the multi-scale sub-pixel convolution and the neighborhood smoothness constraint are proven to be helpful to improve the performance by the ablation study in Sect. V-C.

The remainder of this paper is organized as follows. We first introduce the related work and our contributions in Sect. II and then present the formulation of our method in Sect. III. After that, the details of the proposed method $MSCN_{NS}$ will be described in Sect. IV. In the following, the experimental results and analysis are elaborated in Sect. V. Finally, we conclude the paper in Sect. VI.

II. RELATED WORK AND OUR CONTRIBUTIONS

A. RELATED WORK

In this paper, we focus on how to better explore deep models for solving the problem of depth estimation from monocular images and our method concerns the multi-scale feature fusion and the sub-pixel convolution which is an up-sampling strategy for image super-resolution problem. Therefore, we will briefly review some representative works or concepts into three aspects, namely, CNNs for monocular depth estimation, multi-scale CNNs and image super-resolution approaches.

1) CNNs FOR MONOCULAR DEPTH ESTIMATION

The first depth estimation model exploiting CNN was proposed by Eigen *et al.* [9]. Their model was composed of a coarse-scale CNN and a fine-scale CNN which mapped the input image to the target prediction. The coarse-scale CNN is modified from AlexNet [16] or VGG [17] and outputted a coarse depth map and the fine-scale CNN refined the coarse output with more details of the scene. In their later work [18], Eigen and Fergus extended their model [9] using more CNNs to solve multiple tasks, e.g. depth estimation, surface normal estimation and semantic segmentation. Inspired by Eigen *et al.*'s works [9], [18], Mousavian *et al.* [19] adopted such a multiple CNN style to predict the semantic label and the depth value of each pixel jointly with shared features. In Li *et al.*'s work [20], authors considered preserving local details in the predictions and a two-streamed CNN that could simultaneously predict depth and depth gradients was proposed. Finally, an accurate depth map with local details was acquired by fusing the output information.

Different from multiple CNN approaches, other methods added more spices into CNNs for monocular depth estimation. Liu *et al.* [10] assumed that pixels in one super-pixel own the same depth value and inferred the pixel-level depth through a conditional random field (CRF) whose unary and pairwise potentials were learned by a CNN. Later, they improved their model by introducing a super-pixel pooling operation [21] to remove redundant convolutions and reduce computation costs. Similarly, Li *et al.* [22] and Wang *et al.* [23] involved super-pixel-level predictions and then refined them to the pixel-level predictions via CRFs. Besides, Roy *et al.* [24] combined CNNs with a regression forest, using shallow architectures at each tree node, to avoid the request of large datasets. Xu *et al.* [25], [26] proposed two kinds of sequential deep networks, i.e., the cascade one and the unified graphical one, which fused complementary information derived from multiple side outputs of ResNet by the means of CRFs. In addition, deeper networks for this task have been exploited. Laina *et al.* [27] leveraged the residual learning from ResNet [28] and proposed a fully convolutional architecture with a novel up-sampling method called up-projection.

Although various CNN-based methods have been widely studied in the literature, we are not aware of any work in this field involving image super-resolution techniques or concepts to address the resolution problem and to preserve details of the scene in the prediction.

2) MUTI-SCALE CNNs

Multi-scale fusion strategies for CNNs can be categorized into two main classes, namely, the multi-stream learning and the skip-layer learning, which are illustrated in Fig. 1. In the first class, multiple (or parallel) network streams with different parameters or network architectures are learned. Those networks have different receptive field sizes which refer to multiple scales. Neverova *et al.*'s work [29] for gesture detection and Buyssens *et al.*'s work [30] for image classification are typical samples of this class. The second class has one main stream and adds links between different layers of the main stream. Usually, features from those linked layers are fused by element-wise addition or concatenation. Many CNNs concerning multi-scale fusion for diverse tasks [31]–[35] fell into this class. Our multi-scale fusion strategy belongs to this class, as well. However, we propose supervised structures to acquire multi-scale features which should be more reliable and controllable.

3) IMAGE SUPER-RESOLUTION APPROACHES

Super-resolution

is a traditional problem in computer vision which aims at restoring a high-resolution image from a low-resolution image. The sparse-coding-based (SC) methods [36], [37]

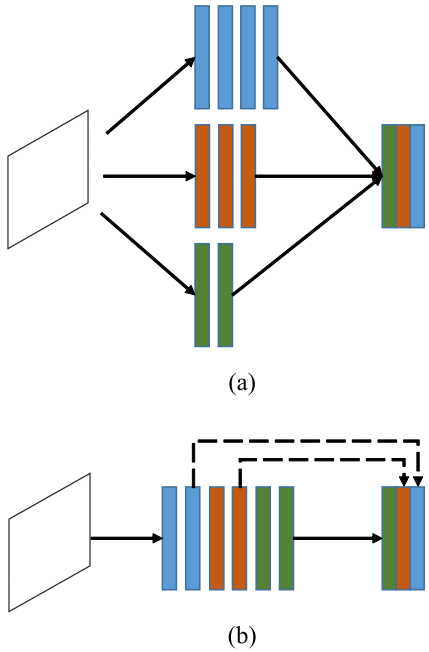


FIGURE 1. Two multi-scale fusion strategies for CNNs. (a) shows the multi-stream learning. (b) outlines the skip-layer learning. Note that the number of streams or skip connections depends on the specific network.

were representative approaches for solving this problem. In Dong *et al.*'s work [38], the authors proved the equivalence between SC methods and CNNs, and thus they proposed a three-layer CNN for this problem. Later, a great number of studies followed this idea and proposed various CNN-based methods. Kim *et al.*'s model [39] involved a deeper network, i.e., VGG [17], and learned the residual between an interpolated low-resolution image and the corresponding high-resolution image. In Kim *et al.*'s another work [40], a recursive structure was proposed to produce several intermediate outputs which were fused to generate the final high-resolution image. Additionally, Shi *et al.*'s work [41] proposed the sub-pixel convolution and made the inference much faster. More recently, Ledig *et al.* [42] involved GAN, a great idea from Goodfellow *et al.* [43], to achieve perceptually good results. In this paper, we exploit some image super-resolution concepts and techniques for the monocular depth estimation and propose a multi-scale sub-pixel convolution to get better results and accelerate the inference.

B. OUR MOTIVATIONS AND CONTRIBUTIONS

Having investigated the literature, we find that in the field of depth estimation based on CNNs, there is still large room for further improvement. First, since CNN usually reduces feature maps' dimensions, many CNN-based methods [9], [18], [25]–[27] can only generate low resolution predictions and adopt bilinear interpolation to restore the original resolution, which leads to blurs in the predictions. Second, even though multi-scale features are involved, there are still many details missing in current methods [9], [18], [19].

In this work, we attempt to solve the aforementioned problems to some extent by proposing a CNN-based approach with multi-scale sub-pixel convolutions and a neighborhood smoothness constraint, namely $MSCN_{NS}$. The advantages and novelties of $MSCN_{NS}$ are highlighted as follows:

(1) We formulate the depth estimation problem as a super-resolution problem on the depth map and up-sample the predicted depth map progressively. Such a formulation is quite different from other depth estimation methods, since those methods up-sample feature maps to get a higher resolution depth map. However, up-sampling feature maps leads to blurs in the prediction. Thus, those methods cannot up-sample feature maps to the dimension of the input and cannot output high resolution depth predictions. It has been corroborated by experiments that our method is able to output a high resolution depth map with sharp edges, runs much faster and achieves state-of-the-art results on popular datasets for the monocular depth estimation task.

(2) For a better prediction, we propose a new up-sampling strategy for depth estimation, namely the multi-scale sub-pixel convolution which involves multi-scale features into Shi *et al.*'s sub-pixel convolution [41]. The proposed structure is quite reasonable in two aspects. First, restoring the resolution of a depth map can be naturally formulated as a super-resolution problem on the depth map. Since the sub-pixel convolution is an efficient way to deal with the image super-resolution problem, it is very likely to be effective on depth estimation. Second, many previous studies [9], [18], [19], [44] have shown that, for depth estimation as well as other pixel-level classification or regression problems, more accurate predictions can be obtained by combining information from multiple scales. The proposed structure is novel, as well. First, it can be regarded as the first attempt to explore image super-resolution techniques on depth estimation. Second, Different from prior studies concerning multi-scale fusions, this work provides a new idea to get multi-scale features from supervised branches of the network. Third, involving multi-scale information in up-sampling is barely studied in previous researches. As one of the most important parts of this work, the multi-scale sub-pixel convolution will be described concretely in Sect. IV-C.

(3) Considering that adjacent pixels with similar features in an image should have close depth values, we have proposed a neighborhood smoothness constraint as a regularization term to train $MSCN_{NS}$. Different from the smoothness in previous studies [10], [19], [21], [24] which use hand-crafted features or take it as a post-processing, our neighborhood smoothness constraint adopts features learned from CNNs automatically to evaluate the similarity of adjacent pixels in order that it can be seamlessly integrated with other parts of our network. In addition, our neighborhood smoothness constraint is highly explainable by regarding it as a pixel level conditional random field (CRF).

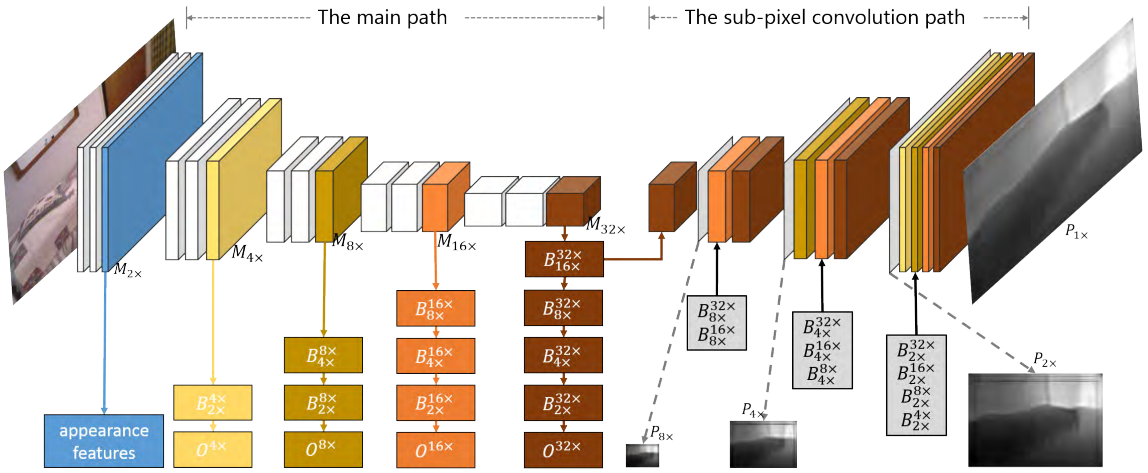


FIGURE 2. The outline of our model. The blue objects represent the smoothness branch and other colored objects connecting to the main path with vertical arrows are scale branches. The black arrow means that features at its starting side will be fused into the corresponding multi-scale sub-pixel convolution.

III. PROBLEM FORMULATION AND OVERIEW

Following previous studies, we formulate the task of monocular depth estimation from monocular images as the problem of learning a non-linear mapping $F : \mathcal{I} \rightarrow \mathcal{D}$, which maps the image space $\mathcal{I}$ to the depth space $\mathcal{D}$. We denote the training set as $\mathcal{T} = (X_i, Y_i), i \in \mathbb{N}$, where $X_i \in \mathcal{I}$ and $Y_i \in \mathcal{D}$ representing the corresponding depth map of X_i, and denote the test set by $\mathcal{Q}$. Thus, our goal is to construct a mapping function F which (a) minimizes the distance of $F(X_j)$ and Y_j, (b) ensures that the resolution of $F(X_j)$ is the same as Y_j's, and (c) preserves fine details of the scene in $F(X_j)$ for a given input $X_j \in \mathcal{Q}$.

For (a), we follow the recent success in CNNs for the image classification and make use of DenseNet [45] as the skeleton of our network. For (b), we involve the super-resolution concepts in this task, which means that our network will generate several intermediate outputs with different resolutions sequentially. Moreover, we adopt an efficient up-sampling technique for super-resolution, i.e., sub-pixel convolution, to accelerate our inference. For (c), we introduce multi-scale features into the sub-pixel convolution. Those multi-scale features comes from different scale branches which are supervised respectively. Thus, our multi-scale strategy is quite different from previous works'. In general, our formulation can be defined as,

$$\begin{aligned} P_{final} &= F_1(X_j) \\ F_n(X_j) &= SP_n(F_{2n}(X_j), \varphi_{2n}(X_j), \varphi_{2^2n}(X_j), \ldots, \varphi_{2^kn}(X_j)) \end{aligned} \quad (1)$$

Here, $n, k \in \mathbb{N}$ and P_{final} refers to our final prediction. n and $2^k n$ refer to the downscaling factors. SP_n refers to a sub-pixel convolution which recovers the resolution to $1/n^2$ of the original input resolution, and $\varphi_{2^t n}(\cdot)$ $(t = 1, 2, \ldots, k)$ refers to a feature extractor to get feature maps related to a downscaling factor of $2^t n$.

As (1) shows, our formulation always up-samples the last result with a factor of 2 until the output owns the same resolution as the input's. Thus, it is different from the end-to-end formulations of previous works. Additionally, since there are no restrictions on $\varphi_{2^t n}(\cdot)$ $(t = 1, 2, \ldots, k)$ and the final target $F_1(\cdot)$, our method can be used as a general architecture for other dense prediction problem, e.g., semantic segmentation and optical flow estimation, as well as monocular depth estimation.

IV. $MSCN_{NS}$: THE PROPOSED METHOD

In this section, we present details of the proposed depth estimation approach $MSCN_{NS}$, including the network architecture, the loss functions, the multi-scale sub-pixel convolution and the neighborhood smoothness constraint. Our approach follows the formulation in Sect. III. Fig. 2 gives the outline of the model, which will be described later.

A. NETWORK ARCHITECTURE

We now describe the details of our model. As shown in Fig. 2, our network contains a main path, a sub-pixel convolution path, a smoothness branch and several scale branches. The main path follows the design of DenseNet [45] without the global pooling and the fully connection. We denote the features at a certain scale from the main path by $M_{n\times}$. "n" in $M_{n\times}$ is the downscaling factor of $M_{n\times}$ whose dimensions are $1/n^2$ the size of the input. Note that $n \in \{2, 4, 8, 16, 32\}$ for $M_{n\times}$ in our method.

Each scale branch uses $M_{n\times}$ with a certain n as inputs to generate features of different scales via a cascade of transposed convolutions and finally outputs a predicted depth map. We denote the scale branch using $M_{n\times}$ as inputs by $B^{n\times}$ and denote the output prediction of $B^{n\times}$ by $O^{n\times}$. Note that "n" in $B^{n\times}$ means that features from the main path with a downscaling factor of n (i.e., $M_{n\times}$) are used as inputs of the scale branch $B^{n\times}$. $O^{n\times}$ has the same resolution as the input. Here, the number "n" in $B^{n\times}$ should be one of {4, 8, 16, 32} and $M_{2\times}$ is not used by any scale branch. **In the whole paper, we use subscripts to represent the real downscaling**

factor of feature maps or an output prediction, and use superscripts to indicate the scale branch to which feature maps or an output prediction belong. Thus, as for different scale features generated by transposed convolutions of the branch $B^{n\times}$, we denote them by $B^{n\times}_{m\times}$, respectively, where m is the downscaling factor of the features and $\forall m \in \{x \mid x \in \{2, 4, 8, 16\}, x < n\}$. The smoothness branch takes $M_{2\times}$ (features from the main path with a downscaling factor of 2) as inputs, and tries to learn features to measure the similarity of neighboring pixels.

The sub-pixel convolution path contains three multi-scale sub-pixel convolutions. Each multi-scale sub-pixel convolution fuses the last low resolution output and all $B^{n\times}_{m\times}$ with a certain m, i.e. $\forall B^{n\times}_{m\times}, n \in \{x \mid x \in \{4, 8, 16, 32\}, x > m\}$. (Note that the first low resolution output is generated by a sub-pixel convolution using $B^{32\times}_{16\times}$.) Then it generates a higher resolution depth map with a downscaling factor of t ($t = m/2$). We call this sub-pixel convolution $SP_{t\times}$ and denote its output by $P_{t\times}$. For example, $P_{4\times}$ is presented as,

$$P_{4\times} = SP_{4\times}(P_{8\times}, B^{16\times}_{8\times}, B^{32\times}_{8\times}). \tag{2}$$

B. MULTI-SCALE TARGET LEARNING AND THE LOSS

During the training phase of our approach, we adopt a multi-scale target learning method to train the sub-pixel convolutions. That is, for a given sample $(X_i, Y_i) \in \mathcal{T}$ we minimize the l_2 distance between Y_i and each of $1\times$ scale predictions which contain $P_{1\times}$ and $O^{n\times}$ ($n = 4, 8, 16, 32$, respectively). And then, we down-sample Y_i to Y^t_i where t is the downscaling factor and minimize the l_2 distance of Y^t_i and the corresponding $P_{t\times}$ to make sure that details of the scene are preserved. Moreover, we fuse $1\times$ scale predictions via weighted average and minimize the l_2 distance between Y_i and the averaged prediction. Thus, our loss function can be defined as a sum of several l_2 distances,

$$\begin{aligned} L_2 = {} & l_2(Y_i, P_{1\times}) + l_2(Y_i, \tilde{Y}_{fs}) \\ & + \lambda_1 \sum_{n\in\mathcal{S}} l_2(Y_i, O^{n\times}) + \sum_{t\in\{2,4,8\}} \lambda^t_2 l_2(Y^t_i, P_{t\times}) \end{aligned} \tag{3}$$

Here, $l_2(\cdot)$ refers to the l_2 distance. λ_1 and λ^t_2 are given constants. $\mathcal{S} = \{4, 8, 16, 32\}$. $\tilde{Y}_{fs}$ refers to the weighted average prediction defined as,

$$\tilde{Y}_{fs} = w_0 P_{1\times} + \sum_{i\in\mathcal{S}} w_i O^{i\times} \tag{4}$$

where w_0 and w_i are weights for fusing $1\times$ scale predictions, and those weights are learned automatically via CNN.

Additionally, we formulate the neighborhood smoothness constraint as a regularization term, L_{smooth}, for training our model, which will be described concretely in Section IV.D. Finally, the whole loss function can be defined as,

$$L = L_2 + L_{smooth}\ . \tag{5}$$

C. MULTI-SCALE SUB-PIXEL CONVOLUTION

Sub-pixel convolution was first proposed by Shi *et al.* [41] as an efficient up-sampling method for the image super-resolution problem. It is originally defined as,

$$I^{SR} = f^L(I^{LR}) = PS(W_L * f^{L-1}(I^{LR}) + b_L) \tag{6}$$

Here, I^{LR} refers to a low resolution RGB image. I^{SR} refers to the high resolution RGB image to be recovered. f^{L-1} refers to a neural network with $L-1$ layers. W_L and b_L are parameters of Layer L, and $PS(\cdot)$ represents a shuffling operator that rearranges the elements of an $H \times W \times (C \cdot r^2)$ tensor to a tensor of the shape $rH \times rW \times C$ where r is the upscaling factor.

Inspired by this, we formulate the prediction of a fine resolution depth map as a multi-stage super-resolution problem on the depth map. In each stage, the predicted depth map is up-sampled by the factor of 2. Therefore, different scale targets should be considered during training and our multi-scale target learning method is born. Furthermore, considering that features from early layers of a CNN carry more detail information of the input image, and the information is very likely to benefit preserving more details of the scene in the prediction, we construct several branches called scale branches in our network and fuse features of different scales from those branches into the sub-pixel convolution. Note that all the scale branches are supervised for extracting better features. Finally, If the resolution of the original input image is $H \times W$, our multi-scale sub-pixel convolution can be defined as,

$$\begin{aligned} P_{n\times} = {} & PS(H([P_{2n\times}, B^{2^2 n\times}_{2n\times}, B^{2^3 n\times}_{2n\times}, \ldots, B^{32\times}_{2n\times}])) \\ & n \in \{1, 2, 4, 8\} \end{aligned} \tag{7}$$

Here, $[P_{2n\times}, B^{2^2 n\times}_{2n\times}, B^{2^3 n\times}_{2n\times}, \ldots, B^{32\times}_{2n\times}]$ refers to the concatenation of input feature maps. $H(\cdot)$ refers to a series of convolutional layers which output a tensor with the shape of $H/(2n) \times W/(2n) \times (1 \cdot 2^2)$, and $PS(\cdot)$ represents a pixelshuffle operation which shuffles the output of $H(\cdot)$ to a predicted depth map of resolution $H/n \times W/n$. Note that the aforementioned notation $SP_n(\cdot)$ in (1) is equivalent to $PS(H(\cdot))$ in (7). Fig. 3 gives illustrations of the original sub-pixel convolution and our multi-scale sub-pixel convolution.

D. NEIGHBORHOOD SMOOTHNESS

The neighborhood smoothness constraint is based on the prior that neighboring pixels with similar appearances in an image are likely to correspond to close depth values. In Liu *et al.*'s work [10], it is presumed that pixels within a super-pixel have the same depth value and super-pixels with similar appearances have closer depth values. And they make use of hand-crafted features to measure the similarity of adjacent super-pixels. Different from them, we use features automatically learned by CNN to measure the similarity of neighboring pixels and the features of each pixel are extracted in a patch whose center is this pixel. And then, we use the weighted average of features of a pixel to represent the appearance of the pixel and define a constraint term in the

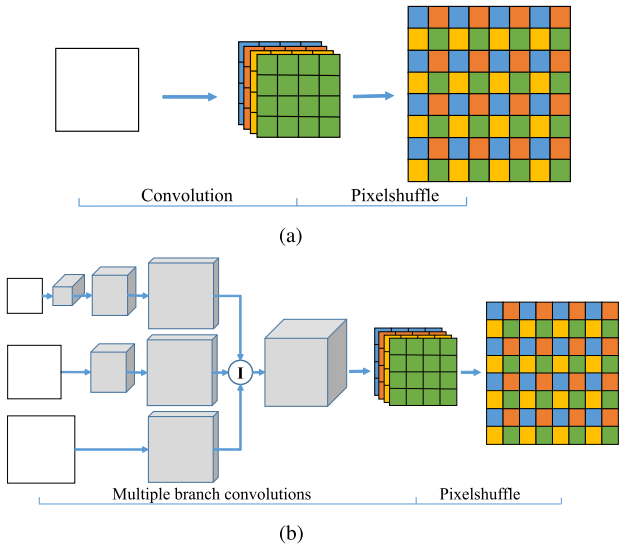


FIGURE 3. (a) illustrates the original sub-pixel convolution proposed by [41]. (b) outlines our multi-scale sub-pixel convolution, which fuses more features from different scale branches. The circle with "I" in it means concatenating features from different branches. And the number of branches before concatenation may be less or more than 3, depending on the output scale of the multi-scale sub-pixel convolution. (a) Sub-pixel convolution. (b) Multi-scale sub-pixel convolution.

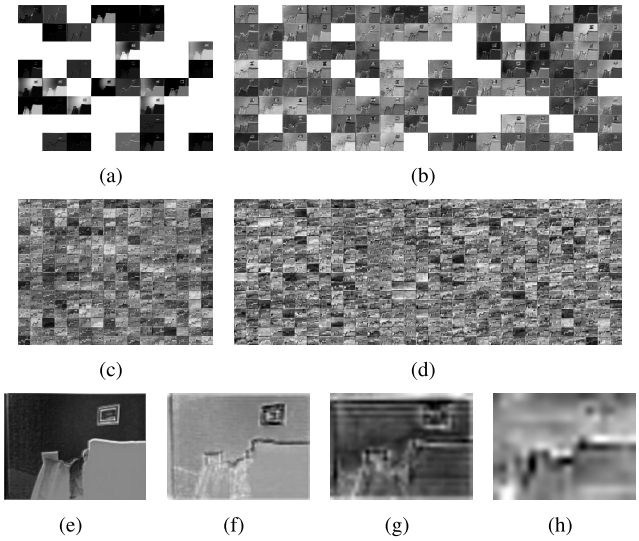


FIGURE 4. Output feature maps from the first convolution layer and the first three DenseBlocks for a typical sample. (a) displays output feature maps from the first convolution layer and the dimension of them is 1/4 that of the input. (b) displays output feature maps from the first DenseBlock and the dimension of them is 1/16 that of the input. (c) displays output feature maps from the second DenseBlock and the dimension of them is 1/64 that of the input. (d) displays output feature maps from the third DenseBlock and the dimension of them is 1/256 that of the input. (e), (f), (g) and (h) refer to the combinations of output feature maps in (a), (b), (c) and (d), respectively. The dimensions of those combinations are 1/4, 1/16, 1/64 and 1/256 that of the input.

loss function, L_{smooth}, as,

$$L_{smooth} = \frac{\lambda_3}{2} \sum_{j-i=1} \left(y_i - y_j\right)^2 e^{-t\left(r_i - r_j\right)^2} \tag{8}$$

Here, λ_3 and t are positive constants. i and j are the locations of two adjacent pixels in a row or a column. r_i and r_j refer to appearances of the two pixels, and y_i and y_j refer to predicted depth values. For better comprehensibility, we will explain why we choose such a formulation, and how we implement it, together with why it should work in the rest of this section.

As (8) shows, there are two parts, namely, L_2 and the weight part. L_2 is used to make the predictions of adjacent pixels closer. However, not all adjacent pixels should own close depth, e.g. adjacent pixels on depth edges. So weights are introduced for such situations. Adjacent pixels which look "different" should own a smaller weight whereas similar pixels should own a larger weight. Therefore, we choose e^x as weights and x should be related to the similarity. We expect to use low level features, such as contours and gradients, to formulate the similarity, since those low level features are different on depth edges but similar on other regions. To our knowledge, the first several layers of CNNs are able to learn various low level features [46], [47]. Therefore, we attempt to explore CNNs to extract the required feature maps.

To get the best feature maps to calculate the similarity of adjacent pixels, we choose a typical sample and visualize outputs from the first convolution layer and the first three DenseBlocks of our network, as well as the combination of outputs from a specific layer or a DenseBlock in Fig. 4. As shown in Fig. 4, (a), (b), (c) and (d) display outputs from the first convolution layer and the first three DenseBlocks, respectively, and the dimension of them is 1/4, 1/16, 1/64 and 1/256 that of the input, respectively. (e), (f), (g) and (h) display the combinations of outputs in (a), (b), (c) and (d), respectively. The dimensions of those combinations are 1/4, 1/16, 1/64 and 1/256 that of the input, respectively. As can be seen, outputs from the first convolution layer and their combination are the most suitable feature maps to formulate the similarity of adjacent pixels. Therefore, we take $2\times$ feature maps ($M_{2\times}$, blue one in Fig. 2) from the main path as the required feature maps for our neighborhood smoothness constraint and feed those feature maps to one DenseBlock with 128 output channels. A convolution layer with a kernel size of 5×5 follows this DenseBlock and generates 4 channel output. The 4 channel output is then shuffled into one channel to up-sample its dimension to the original input's. Finally, each value in the output is regarded as the feature of the pixel in the same location (e.g. r_i or r_j in (8)).

Actually, our neighborhood smoothness constraint can be regarded as a pixel level conditional random field (CRF) with pairwise potentials defined as,

$$\begin{aligned} \varphi(y_i, y_j, r_i, r_j) &= \mu\left(y_i, y_j\right) k\left(r_i, r_j\right) \\ \mu\left(y_i, y_j\right) &= \frac{1}{2}\left(y_i - y_j\right)^2, \quad k\left(r_i, r_j\right) = \lambda e^{-t\left(r_i - r_j\right)^2} \end{aligned} \tag{9}$$

where $\mu(y_i, y_j)$ represents the compatibility function between pixel i and j, and $k(r_i, r_j)$ represents a self-defined weight kernel. Since many works [10], [12], [21], [22], [25], [26], [48] imply that incorporating CRF into dense prediction model is a good strategy, our neighborhood smoothness constraint should work, as well.

V. EXPERIMENTS

In this Section, we will provide various experiments, including ablation studies on different components and comparisons with state-of-the-art methods on popular datasets, to present the superiority of the proposed method. In the following, we will first describe the experimental protocols (or experimental settings) and implementation details, and then provide the experimental results together with analyses.

A. EXPERIMENTAL PROTOCOL

1) DATASETS

In this paper, we involve two popular datasets which are publicly available and widely used in the field of monocular depth estimation, namely, Make3D Range Image Dataset [13] and NYU Depth Dataset V2 [49]. We conduct the ablation study on NYU Depth Dataset V2 and compare our method with state-of-the-art approaches on both datasets. Details about the two datasets are described here.

The **NYU Depth Dataset V2** [49] is an indoor scene RGB-D dataset with 120K frames and corresponding depth maps. It contains 464 scenes captured by a Microsoft Kinect with the resolution of 640×480 and those scenes are officially divided into 249 training scenes and 215 test scenes. Following Laina *et al.*'s work [27], we sample frames with a fixed step out of each training sequence and acquire approximately 12k images. For better performance, we conduct offline data augmentation which will be described later and finally get around 72k samples for training. Since there are obvious black regions in the ground-truth depth maps, we mask out pixels with the depth value of 0 and train our model on the rest pixels. For test, we use the standard test set, which contains 694 images with filled-in depth values, to compare with previous works. During our experiments, all images with the corresponding depth maps are down-sampled to 320×240.

The **Make3D** dataset [13] is an outdoor scene RGB-D dataset, containing 534 images with the resolution of 1704×2272. Officially, images of this dataset are split into 400 images for training and 134 images for test. We get 9.6K images via similar offline data augmentation used for NYU Depth Dataset V2 and resize all images to 345×460. Following Laina *et al.*'s work [27], we further reduce the resolution of the images by half as the input of our network. As a result of the limitation of the device for collecting depth map in the open air, the depth map resolution is restricted to 305×55 and depth values above $80m$ are cropped to $80m$ in this dataset. Therefore, we report our results using two kinds of criteria, C1 error and C2 error, as previous works used. C1 errors are calculated only in the regions with the ground-truth less than 70 meters whereas C2 errors are calculated over the entire images.

2) DATA AUGMENTATION

To increase the size of the training sets and reduce overfitting, we follow the method described in [9] and conduct offline data augmentation on both datasets as follows.

- **Scale**: Input images and corresponding depth maps are scaled by $s \in \{1, 1.2, 1.5\}$ and depth values are divided by s.
- **Rotation**: Input images and corresponding depth maps are rotated by a random value $r \in [-5, 5]$ degrees and the empty regions caused by the rotation are masked out during training. Note that this operation is conducted only on Make3D.
- **Translation**: Input images and corresponding depth maps are randomly cropped to the size of the input.
- **Color**: Each channel of input images are multiplied globally by a random value $c \in [0.85, 1.15]$. Note that this operation is conducted only on Make3D.
- **Flips**: The input image and its corresponding depth map in every data pair are flipped to generate a symmetrical pair.

3) EVALUATION METRICS

The same as prior works, three criteria for errors and one criterion for accuracy are considered for quantitative evaluation. Specially, they are defined as,

- Average relative error (rel): $\frac{1}{T}\sum_i \frac{|y_i - y_i^*|}{y_i^*}$.
- Root mean squared error (rms): $\sqrt{\frac{1}{T}\sum_i \left(y_i - y_i^*\right)^2}$.
- Average log10 error (log10): $\frac{1}{T}\sum_i \left|log_{10}y_i^* - log_{10}y_i\right|$.
- Accuracy with threshold (thr): percentage (%) of y_i $s.t.$: $\max\left(\frac{y_i}{y_i^*}, \frac{y_i^*}{y_i}\right) = \delta < thr$.

where y_i and y_i^* are the predicted depth and the ground-truth depth of the pixel i, respectively, and T is the total number of pixels in the evaluated image.

4) BASELINES FOR ABLATION STUDY

We set four baselines in the ablation study to clearly validate the effectiveness of the multi-scale target learning, the multi-scale sub-pixel convolution and the neighborhood smoothness constraint. Specially, they are elaborated as follows.

- **DenseNet-TC** (DenseNet with Transposed Convolution): We trained a DenseNet without the global pooling and the fully connection for depth estimation and extended it with transposed convolutions to get a fine resolution of the predicted depth map.
- **DenseNet-MT** (DenseNet with Multi-scale Targets): We trained a network similar as DenseNet-TC and added the multi-scale target learning to it. Specifically, features from each transposed convolution were fed to an extra convolutional layer to get a predicted depth map of a certain scale and those predictions were supervised respectively.
- **DenseNet-SC** (DenseNet with Sub-pixel Convolution): We replaced the transposed convolution in DenseNet-MT with the original sub-pixel convolution proposed in Shi *et al.*'s work [41]. Note that in this network, the output of each sub-pixel convolution was supervised directly.

TABLE 1. Components and baselines. "–" means that a method does not own the component whereas √ means that a method owns the component. A: Multi-scale target learning; B: Sup-pixel convolution; C: Multi-scale fusion; D: Scale branches; E: Neighborhood smoothness. Note that our multi-scale sub-pixel convolution consists of component B, C and D.

	A	B	C	D	E
DenseNet-TC	–	–	–	–	–
DenseNet-MT	√	–	–	–	–
DenseNet-SC	√	√	–	–	–
DenseNet-SCM	√	√	√	–	–
MSCN	√	√	√	√	–
$MSCN_{NS}$	√	√	√	√	√

- **DenseNet-SCM** (DenseNet with Sub-pixel Convolution and Multi-scale features): We involved multi-scale features into sub-pixel convolutions. However, those features come from early layers of the network instead of scale branches.
- **MSCN** (Multi-scale Sub-pixel Convolutional Network): The proposed method without the neighborhood smoothness constraint.
- $MSCN_{NS}$ (Multi-scale Sub-pixel Convolutional Network with a Neighborhood Smoothness constraint): The proposed method.

For better comprehensibility, we list all related components in Table 1 and show whether a network owns a component.

B. IMPLEMENTATION DETAILS

We implement our method on the popular CNN platform, PyTorch.[1] Training is done on Ubuntu 16.04 with an NVIDIA Titan X Pascal GPU. We choose DenseNet-121 as the main path. We reference the implementation of DenseNet-121 in the vision project[2] and use the pre-trained model to initialize parameters of the four denseblocks. Other layers of the network are initialized by the mean of xavier [55]. We use Adam strategy with $weight\_decay = 0.0005$ and set $\lambda_1 = 0.5$, $\lambda_2^i = 1/i, (i = 2, 4, 8)$, $\lambda_3 = 0.01$ and $t = 2$. The batch size is set to 16 due to the memory limitation. The learning rate is set to 0.001 at the very beginning and reduced 70% every M epochs. The value of M depends on the size of the dataset. Generally, we set it to 6∼10 for NYUv2 Depth and 20∼60 for Make3D.

C. EXPERIMENT RESULTS

1) ABLATION STUDY ON COMPONENTS

We first compare the proposed method $MSCN_{NS}$ with several aforementioned baselines to validate the effectiveness of the multi-scale target learning, the multi-scale sub-pixel convolution and the neighborhood smoothness constraint, respectively. The results are shown in Table 1, from which we could have the following findings. First, all approaches with the multi-scale target learning perform better than DenseNet-TC, indicating that our multi-scale target learning is very effective for depth estimation. Second, DenseNet-SC outperforms DenseNet-MT, which indicates that the sub-pixel convolution is more suitable than the widely used transposed convolution for monocular depth estimation. Third, DenseNet-SCM outperforms DenseNet-SC with a great increase on performance and MSCN outperforms DenseNet-SCM. Such a result clearly demonstrates that multi-scale fusion is a good strategy for this task and our supervised scale branches are able to provide better multi-scale features. This result is quite reasonable, since multi-scale information profoundly benefits keeping details of the scene in the prediction. Fourth, compared with MSCN, $MSCN_{NS}$ gains obvious performance increases on all criteria. Considering that the only difference between $MSCN_{NS}$ and MSCN is that the former is trained with the neighborhood smoothness regularization term, it can be concluded that our neighborhood smoothness constraint further improves the results.

Since there are multiple outputs ($O^{4\times} \sim O^{32\times}$ and $P_{8\times} \sim P_{1\times}$) in our network, we display the qualitative results of those outputs in Fig. 5 and present the quantitative comparisons of them in Table 2. From the results and comparisons, several interesting conclusions can be drawn. First, the performances from $O^{4\times}$ to $O^{32\times}$ increase sequentially whereas the figures become less sharp. Such a phenomenon clearly illustrates that the early layers of the network provide more information about the contour of the object whereas deeper layers provide more useful information for depth values. Thus, there is a trade-off between sharpness and accuracy using single scale features. Second, $P_{8\times} \sim P_{1\times}$ are nearly the same irrespective of the resolution, which indicates that our multi-scale sub-pixel convolution overcomes this trade-off and is able to provide a sharp and accurate prediction. Third, $O^{32\times}$ and $P_{1\times}$ share the same backbone network. But $P_{1\times}$ is much sharper than $O^{32\times}$. As mentioned in Sect. IV-A, $P_{1\times}$ adopts our multi-scale sub-pixel convolution as the upsampling strategy whereas $O^{32\times}$ employs transposed convolutions. Thus, we can conclude that our multi-scale sub-pixel convolution is more suitable than the transposed convolution for depth estimation, and our method is able to get sharp details in a high resolution prediction.

2) EVALUATION ON NYU DEPTH DATASET V2

In order to show the superiority of $MSCN_{NS}$, we consider several state-of-the-art methods on this dataset and compare it with them using the standard test with 694 images. The results are listed in Table 4. Moreover, we also make the comparison on time costs with several methods whose code is publicly available and the results are summarized in Table 6. The time cost is divided into two parts, i.e. the preprocessing time and prediction time, to illustrate the advantages of our method. Note that all methods are evaluated on the same computer and accelerated by an NVIDIA Titan X (Pascal) GPU except Karsch *et al.*'s method [50].

From those two tables, it can be found that our method achieves the best or the second best performance on all criteria and runs much faster. It only consumes 1/2 the time

[1] http://pytorch.org/
[2] https://github.com/pytorch/vision

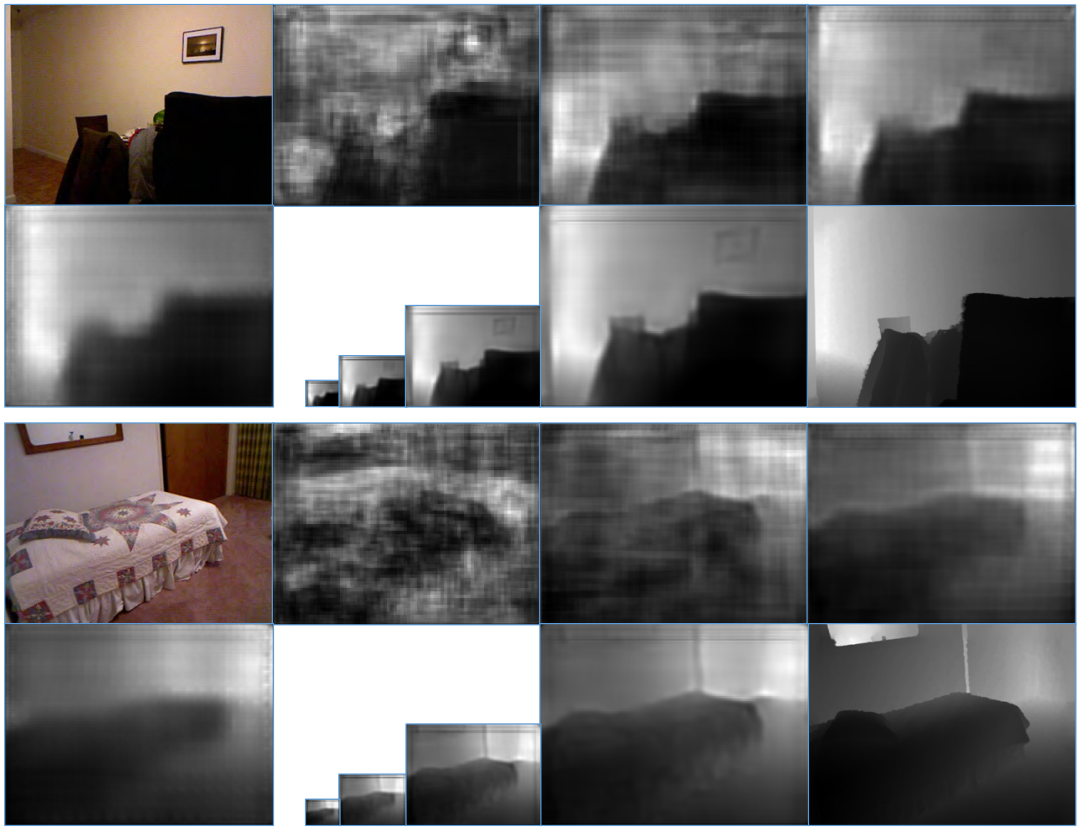


FIGURE 5. Two test samples in NYU Depth V2. For each sample, there are two rows. The first row (from left to right) is the RGB image, $O^{4\times}$, $O^{8\times}$ and $O^{16\times}$ and the second row (from left to right) is $O^{32\times}$, $P_{8\times}$, $P_{4\times}$, $P_{2\times}$, $P_{1\times}$ (the final prediction) and the ground-truth depth map.

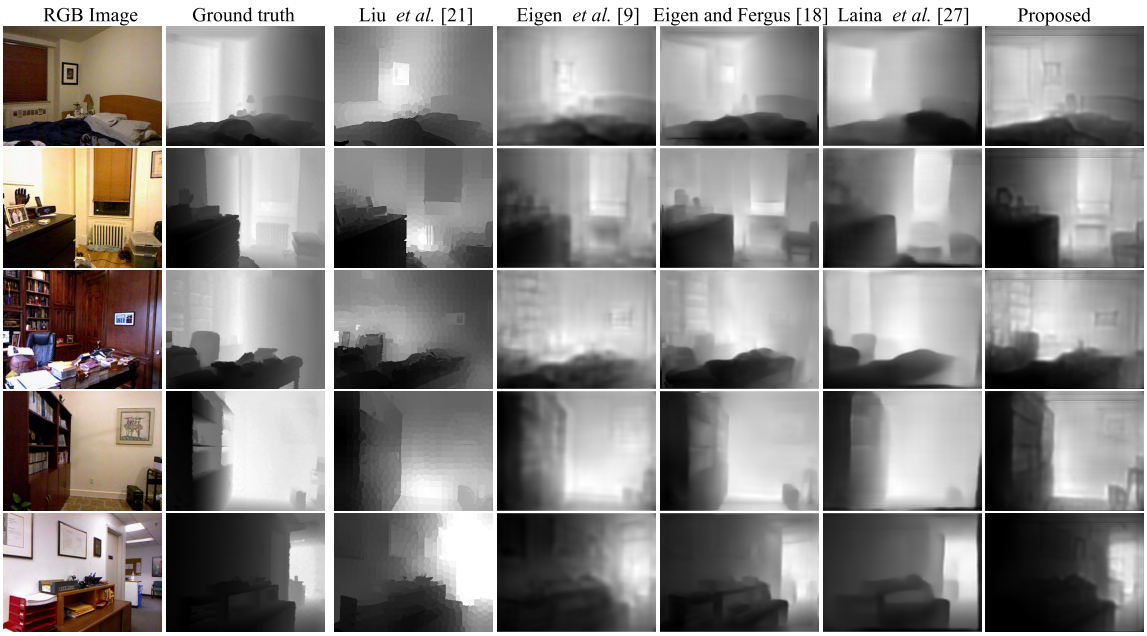


FIGURE 6. Qualitative evaluations on NYU Depth v2. Compared with other state-of-the-art methods. Note that the output resolution of the proposed method is the same as input's, whereas Eigen *et al.*'s [9] is 1/16 of the input's. Eigen and Fergus's [18] is 1/4 of the input's, and Laina *et al.*'s [27] is 1/4 of the input's.

of the runner-up and can meet the requirement of real-time application. Moreover, our training set with 72K images is smaller than Laina *et al.*'s [27] with 95K images, as well as Eigen and Fergus's [18] with 120K images. Besides, the size of our model is approximately 100 M, which is 40% that of Laina *et al.*'s and 1/8 that of Eigen and Fergus's.

FIGURE 7. Qualitative evaluations of our approach on Make3D. Fine boundaries of objects in different environments are provided in our predictions.

TABLE 2. Intermediate output comparison on NYU Depth v2. $O^{4\times} \sim O^{32\times}$ are outputs of corresponding scale branches. $P_{8\times} \sim P_{1\times}$ are outputs of corresponding sub-pixel convolutions. Note that $P_{1\times}$ is the final output of our method.

Output	Error (lower is better)			Accuracy (higher is better)		
	rel	log10	rms	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
$O^{4\times}$	0.289	0.136	1.049	0.426	0.742	0.911
$O^{8\times}$	0.203	0.098	0.772	0.596	0.870	0.966
$O^{16\times}$	0.156	0.076	0.634	0.719	0.914	0.978
$O^{32\times}$	0.154	0.075	0.635	0.726	0.915	0.978
$P_{8\times}$	0.174	0.087	0.748	0.656	0.902	0.974
$P_{4\times}$	0.154	0.075	0.663	0.723	0.932	0.982
$P_{2\times}$	0.137	0.064	0.573	0.781	0.954	0.990
$P_{1\times}$	**0.128**	**0.059**	**0.523**	**0.813**	**0.964**	**0.992**

TABLE 3. Baseline comparison on NYU Depth v2. DenseNet-TC and DenseNet-MT verify multi-scale target learning. DenseNet-SC and DenseNet-SCM verify the effectiveness of the multi-scale fusion. DenseNet-SCM and MSCN verify our multi-scale fusion strategy in sub-pixel convolution. MSCN and $MSCN_{NS}$ verify the neighborhood smoothness constraint.

Method	Error (lower is better)			Accuracy (higher is better)		
	rel	log10	rms	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
DenseNet-TC	0.248	0.101	0.786	0.585	0.870	0.962
DenseNet-MT	0.240	0.097	0.761	0.601	0.879	0.965
DenseNet-SC	0.227	0.094	0.720	0.613	0.885	0.969
DenseNet-SCM	0.161	0.073	0.600	0.735	0.936	0.985
MSCN	0.154	0.068	0.569	0.755	0.941	0.986
$MSCN_{NS}$	**0.128**	**0.059**	**0.523**	**0.813**	**0.964**	**0.992**

Compared with super-pixel level CRF based approaches [21], [48], our method is much more efficient and effective, which illustrates the merit of our multi-scale sub-pixel convolution and the neighborhood smoothness constraint.

For qualitative evaluation, we provide qualitative results of several methods in Fig. 6. One thing should be mentioned that the output resolution of our method is the same as the output's, whereas Eigen *et al.*'s [9] is 1/16 of the input's.

TABLE 4. Performance evaluation of state-of-the-art methods on NYU Depth v2. The best scores are highlighted in bold font and the runner-up scores are underlined.

Method	Error (lower is better)			Accuracy (higher is better)		
	rel	log10	rms	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Saxena *et al.* [13]	0.349	-	1.214	0.447	0.745	0.897
Karsch *et al.* [50]	0.350	0.131	1.2	-	-	-
Liu *et al.* [48]	0.335	0.127	1.06	-	-	-
Eigen *et al.* [9]	0.215	-	0.907	0.611	0.887	0.971
Liu *et al.* [21]	0.234	0.095	0.842	0.604	0.885	0.973
Mousavian *et al.* [19]	0.200	-	0.816	0.568	0.856	0.956
Roy and Todorovic [24]	0.187	0.078	0.744	-	-	-
Eigen and Fergus [18]	0.158	-	0.641	0.769	0.950	0.988
Li *et al.* [20]	0.143	0.063	0.635	0.788	0.958	0.991
Laina *et al.* (l_2) [27]	0.138	0.060	0.592	0.785	0.952	0.980
Kundu *et al.* [51]	0.136	**0.057**	0.603	0.805	0.948	0.982
Lee *et al.* [52]	0.139	-	0.572	**0.815**	0.963	0.991
$MSCN_{NS}$ (Ours)	**0.128**	0.059	**0.523**	0.813	**0.964**	**0.992**

TABLE 5. Performance evaluation of state-of-the-art methods on Make3D. The best scores are highlighted in bold font and the runner-up scores are underlined.

Method	C1 error(lower is better)			C2 error(lower is better)		
	rel	log10	rms	rel	log10	rms
Saxena *et al.* [13]	-	-	-	0.370	0.187	-
Karsch *et al.* [50]	0.355	0.127	9.20	0.361	0.148	15.10
Liu *et al.* [48]	0.335	0.137	9.49	0.338	0.134	12.60
Liu *et al.* [10]	0.314	0.119	8.60	0.307	0.125	12.89
Zhang *et al.* [53]	0.374	0.127	9.18	0.364	0.141	14.11
Liu *et al.* [21]	0.312	0.113	9.10	0.305	0.120	13.24
Roy and Todorovic [24]	-	-	-	0.260	0.119	12.40
Li *et al.* [22]	0.278	0.092	7.19	0.279	0.102	10.67
Kundu *et al.* [51]	0.452	-	9.559	-	-	-
Atapour and Breckon [54]	0.423	0.122	9.002	-	-	-
$MSCN_{NS}$ (Ours)	**0.254**	**0.080**	**6.92**	**0.249**	**0.088**	**10.47**

TABLE 6. Time evaluation on NYU Depth v2. Our method runs much faster than other competitors and only consumes appropriately 1/2 the time of the runner-up.

Method	Preprocessing	Prediction	Total
Karsch *et al.* [50]	0s	60s	60s
Eigen *et al.* [9]	0s	2.091s	2.091s
Eigen and Fergus [18]	0s	5.622s	5.622s
Liu *et al.* [21]	1.252s	0.316s	1.607s
Laina *et al.* [27]	0s	0.257s	0.257s
Atapour and Breckon [54]	0s	0.036s	0.036s
$MSCN_{NS}$ (Ours)	0s	**0.019s**	**0.019s**

Eigen and Fergus's [18] is 1/4 of the input's, and Laina *et al.*'s [27] is 1/4 of the input's. Thus, our method is able to output a larger resolution prediction with sharp edges, as well as to provide more accurate depth values.

3) EVALUATION ON MAKE3D

We evaluate our method using the official test set with 134 images and list the comparison results with several state-of-the-art approaches in Table 5. It can be observed that our method outperforms all competitors on all criteria. Moreover, it should be noted that Make3D is an outdoor depth dataset. And thus the range of its depth values is much larger than that of indoor dataset like NYU Depth Dataset V2, which brings more challenges for the prediction. Nevertheless, our method makes a great improvement on the metric log10 which refers to an absolute error on logarithm. Therefore, it can be concluded that our method is superior to predict relatively accurate depth value in spite of a large range of ground truth values.

We also provide the qualitative evaluations of our method on Make3D, which are presented in Fig. 7. Since the depth range is large, we provide the reverse depth map for better visualization. As shown, our method provides fine boundaries of objects even in complex environments involving buildings, shrubs and trees.

VI. CONCLUSION

In this paper, we have exploited image super-resolution concepts and techniques for monocular depth estimation and proposed a CNN-based approach with multi-scale sub-pixel convolutions and a neighborhood smoothness constraint. In our model, there is a main path, a sub-pixel convolution path, a smoothness branch and four scale branches. The main path and scale branches are supervised, and generate features of different scales. Then, the sub-pixel convolution path leverages the multi-scale feature fusion and sub-pixel convolutions to get an accurate depth map. During training, a novel multi-scale target learning strategy is adopted to train the sub-pixel convolutions. Moreover, we have proposed a novel neighborhood smoothness constraint that makes use

of features from the smoothness branch to further improve the performance. Our approach achieves the state-of-the-art results, provides high resolution depth maps with details of the scenes and runs much faster.

REFERENCES

[1] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik, "Indoor scene understanding with RGB-D images: Bottom-up segmentation, object detection and semantic segmentation," *Int. J. Comput. Vis.*, vol. 112, no. 2, pp. 133–149, 2015.
[2] X. Ren, L. Bo, and D. Fox, "RGB-(D) scene labeling: Features and algorithms," in *Proc. CVPR*, 2012, pp. 2759–2766.
[3] A. Eitel, J. T. Springenberg, L. Spinello, M. Riedmiller, and W. Burgard, "Multimodal deep learning for robust RGB-D object recognition," in *Proc. IROS*, 2015, pp. 681–687.
[4] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser, "3DMatch: Learning local geometric descriptors from RGB-D reconstructions," in *Proc. CVPR*, 2017, pp. 199–208.
[5] Z. Wang, H. Liu, X. Wang, and Y. Qian, "Segment and label indoor scene based on RGB-D for the visually impaired," in *Proc. MMM*, 2014, pp. 449–460.
[6] H. Hirschmuller, "Accurate and efficient stereo processing by semi-global matching and mutual information," in *Proc. CVPR*, 2005, pp. 807–814.
[7] J. Žbontar and Y. LeCun, "Stereo matching by training a convolutional neural network to compare image patches," *J. Mach. Learn. Res.*, vol. 17, pp. 1–32, Apr. 2016.
[8] A. Seki and M. Pollefeys, "SGM-Nets: Semi-global matching with neural networks," in *Proc. CVPR Workshop*, 2017, pp. 21–26.
[9] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Proc. NIPS*, 2014, pp. 2366–2374.
[10] F. Liu, C. Shen, and G. Lin, "Deep convolutional neural fields for depth estimation from a single image," in *Proc. CVPR*, 2015, pp. 5162–5170.
[11] B. Liu, S. Gould, and D. Koller, "Single image depth estimation from predicted semantic labels," in *Proc. CVPR*, 2010, pp. 1253–1260.
[12] A. Saxena, S. H. Chung, and A. Y. Ng, "Learning depth from single monocular images," in *Proc. NIPS*, 2006, pp. 1161–1168.
[13] A. Saxena, M. Sun, and A. Y. Ng, "Make3D: Learning 3D scene structure from a single still image," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 5, pp. 824–840, May 2009.
[14] K. Karsch, C. Liu, and S. B. Kang, "Depth extraction from video using non-parametric sampling," in *Proc. ECCV*, 2012, pp. 775–788.
[15] J. Konrad, M. Wang, and P. Ishwar, "2D-to-3D image conversion by learning depth from examples," in *Proc. CVPR Workshop*, 2012, pp. 16–22.
[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012, pp. 1097–1105.
[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. ICLR*, 2015, pp. 1–14.
[18] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *Proc. ICCV*, 2015, pp. 2650–2658.
[19] A. Mousavian, H. Pirsiavash, and J. Košecká, "Joint semantic segmentation and depth estimation with deep convolutional networks," in *Proc. 3DV*, 2016, pp. 611–619.
[20] J. Li, R. Klein, and A. Yao, "A two-streamed network for estimating fine-scaled depth maps from single RGB images," in *Proc. CVPR*, 2017, pp. 3372–3380.
[21] F. Liu, C. Shen, G. Lin, and I. Reid, "Learning depth from single monocular images using deep convolutional neural fields," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 10, pp. 2024–2039, Oct. 2016.
[22] B. Li, C. Shen, Y. Dai, A. van den Hengel, and M. He, "Depth and surface normal estimation from monocular images using regression on deep features and hierarchical CRFs," in *Proc. CVPR*, 2015, pp. 1119–1127.
[23] P. Wang, X. Shen, Z. Lin, S. Cohen, B. Price, and A. L. Yuille, "Towards unified depth and semantic prediction from a single image," in *Proc. CVPR*, 2015, pp. 2800–2809.
[24] A. Roy and S. Todorovic, "Monocular depth estimation using neural regression forest," in *Proc. CVPR*, 2016, pp. 5506–5514.
[25] D. Xu, E. Ricci, W. Ouyang, X. Wang, and N. Sebe, "Multi-scale continuous CRFs as sequential deep networks for monocular depth estimation," in *Proc. CVPR*, 2017, pp. 161–169.
[26] D. Xu, E. Ricci, W. Ouyang, X. Wang, and N. Sebe, "Monocular depth estimation using multi-scale continuous CRFs as sequential deep networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published, doi: 10.1109/TPAMI.2018.2839602.
[27] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *Proc. 3DV*, 2016, pp. 239–248.
[28] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016, pp. 770–778.
[29] N. Neverova, C. Wolf, G. W. Taylor, and F. Nebout, "Multi-scale deep learning for gesture detection and localization," in *Proc. ECCV Workshop*, 2014, pp. 474–490.
[30] P. Buyssens, A. Elmoataz, and O. Lézoray, "Multiscale convolutional neural networks for vision–based classification of cells," in *Proc. ACCV*, 2012, pp. 342–352.
[31] G. Bertasius, J. Shi, and L. Torresani, "DeepEdge: A multi-scale bifurcated deep network for top-down contour detection," in *Proc. CVPR*, 2015, pp. 4380–4389.
[32] Y. Ganin and V. Lempitsky, "N^4-fields: Neural network nearest neighbor fields for image transforms," in *Proc. ACCV*, 2014, pp. 536–551.
[33] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Hypercolumns for object segmentation and fine-grained localization," in *Proc. CVPR*, 2015, pp. 447–456.
[34] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. CVPR*, 2015, pp. 3431–3440.
[35] P. Sermanet, S. Chintala, and Y. LeCun, "Convolutional neural networks applied to house numbers digit classification," in *Proc. ICPR*, 2012, pp. 3288–3291.
[36] J. Yang, J. Wright, T. Huang, and Y. Ma, "Image super-resolution as sparse representation of raw image patches," in *Proc. CVPR*, 2008, pp. 1–8.
[37] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Trans. Image Process.*, vol. 19, no. 11, pp. 2861–2873, Nov. 2010.
[38] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *Proc. ECCV*, 2014, pp. 184–199.
[39] J. Kim, J. K. Lee, and K. Mu Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proc. CVPR*, 2016, pp. 1646–1654.
[40] J. Kim, J. K. Lee, and K. M. Lee, "Deeply-recursive convolutional network for image super-resolution," in *Proc. CVPR*, 2016, pp. 1637–1645.
[41] W. Shi *et al.*, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proc. CVPR*, 2016, pp. 1874–1883.
[42] C. Ledig *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proc. CVPR*, 2017, pp. 105–114.
[43] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. NIPS*, 2014, pp. 2672–2680.
[44] S. Xie and Z. Tu, "Holistically-nested edge detection," in *Proc. ICCV*, 2015, pp. 1395–1403.
[45] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. CVPR*, 2017, pp. 2261–2269.
[46] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. ECCV*, 2014, pp. 818–833.
[47] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson. (2015). "Understanding neural networks through deep visualization." [Online]. Available: https://arxiv.org/abs/1506.06579
[48] M. Liu, M. Salzmann, and X. He, "Discrete-continuous depth estimation from a single image," in *Proc. CVPR*, 2014, pp. 716–723.
[49] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *Proc. ECCV*, 2012, pp. 746–760.
[50] K. Karsch, C. Liu, and S. B. Kang, "Depth transfer: Depth extraction from video using non-parametric sampling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 11, pp. 2144–2158, Nov. 2014.
[51] J. N. Kundu, P. K. Uppala, A. Pahuja, and R. V. Babu, "AdaDepth: Unsupervised content congruent adaptation for depth estimation," in *Proc. CVPR*, 2018, pp. 2656–2665.
[52] J. H. Lee, M. Heo, K. R. Kim, and C. S. Kim, "Single-image depth estimation based on Fourier domain analysis," in *Proc. CVPR*, 2018, pp. 330–339.

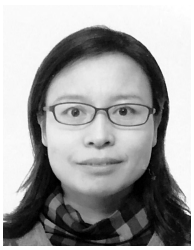
[53] Y. Zhang *et al.*, "Search-based depth estimation via coupled dictionary learning with large-margin structure inference," in *Proc. ECCV*, 2016, pp. 858–874.
[54] A. Atapour-Abarghouei and T. P. Breckon, "Real-time monocular depth estimation using synthetic data with domain adaptation via image style transfer," in *Proc. CVPR*, 2018, pp. 2800–2810.
[55] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. AISTATS*, 2010, pp. 249–256.

SHIYU ZHAO received the B.S. degree from the School of Software Engineering, Tongji University, Shanghai, China, in 2017, where he is currently pursuing the master's degree. His research interests are visibility enhancement for bad weather images, scene understanding, and machine learning.

LIN ZHANG (M'11–SM'15) received the B.S. and M.S. degrees from the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2003 and 2006, respectively, and the Ph.D. degree from the Department of Computing, The Hong Kong Polytechnic University, Hong Kong, in 2011, where he joined the Department of Computing as a Research Assistant, in 2011. In 2011, he joined the School of Software Engineering, Tongji University, Shanghai, where he is currently an Associate Professor. His current research interests include the environment perception of intelligent vehicle, pattern recognition, computer vision, and perceptual image/video quality assessment.

YING SHEN (M'13) received the B.S. and M.S. degrees from the Software School, Shanghai Jiao Tong University, Shanghai, China, in 2006 and 2009, respectively, and the Ph.D. degree from the Department of Computer Science, City University of Hong Kong, Hong Kong, in 2012. In 2013, she joined the School of Software Engineering, Tongji University, Shanghai, where she is currently an Associate Professor. Her research interests include bioinformatics and pattern recognition.

SHENGJIE ZHAO (SM'09) received the B.S. degree in electrical engineering from the University of Science and Technology of China, Hefei, China, in 1988, the M.S. degree in electrical and computer engineering from the China Aerospace Institute, Beijing, China, in 1991, and the Ph.D. degree in electrical and computer engineering from Texas A&M University, College Station, TX, USA, in 2004. He is currently a Professor with the School of Software Engineering, Tongji University, Shanghai, China. In previous postings, he conducted research at Lucent Technologies, Whippany, NJ, USA, and the China Aerospace Science and Industry Corporation, Beijing. His research interests include big data, wireless communications, image processing, and signal processing. He is a Fellow of the Thousand Talents Program of China.



HUIJUAN ZHANG received the B.S. and M.S. degrees from the Department of Computer Science, Xidian University, Shaanxi, China, in 1993 and 1999, respectively, and the Ph.D. degree from the College of Computer Science and Technology, Xidian University, in 2006, where she joined the Department of Computing, as a Lecturer, in 1996, and as an Associate Professor, in 2003. In 2007, she joined the School of Software Engineering, Tongji University, Shanghai, where she is currently an Associate Professor. Her current research interests include mobile computing, pattern recognition, and virtual reality.