Towards Handling Sudden Changes in Feature Maps during Depth Estimation

Yao Xue, Yu Cao, Xubin Feng, Meilin Xie, Ke Li, Xingjun Zhang, *Member, IEEE,* Xueming Qian, *Member, IEEE,*

***Abstract*—Depth estimation aims to predict depth map from RGB images without high cost equipments. Deep learning based depth estimation methods have shown their effectiveness. However in existing methods, depth information is represented by a per-pixel depth map. Such depth map representation is fragile facing different kinds of depth changes. This paper proposes a Compressive Sensing based Depth Representation (CSDR) scheme, which formulates the problem of depth estimation in pixel space into the task of fixed-length vector regression in representation space. In this way, deep model training errors will not directly interfere depth estimation, and distortions in estimated depth maps can be restrained in the greatest extent. In addition, we improve depth estimation from two other aspects: model structure and loss function. To capture the features in different scales, we propose a Multiscale Encoder & Multiscale Decoder (MEMD) structure as the vector regression model. To further deal with depth change, we also modify the loss function, where the curvature difference between ground truth and estimation is directly incorporated. With the support of CSDR, MEMD and the curvature loss, the proposed approach achieves superior performance on a challenging depth estimation dataset: NYU-Depth-v2. A range of experiments support our claim that regression in CSDR space performs better than traditionally direct depth map estimation in pixel space.**

***Index Terms*—depth estimation, depth representation, multi-scale feature.**

I. Introduction

DEPTH estimation is the problem of predicting depth map from RGB images. It is attracting more and more research attention, because its wide applications in 3D reconstruction [1] [2], augmented reality[3], virtual reality, autonomous driving and semantic segmentation[4]. For example, in the field of autonomous driving, the depth map can provide the distance information of objects around the vehicle, and assist obstacle detection and avoidance.

Yao Xue (xueyao@xjtu.edu.cn), Ke Li is with the School of Information and Communication Engineering, Xi'an Jiaotong University, Xi'an 710049, China. Yu Cao is with Space Precision Measurement Laboratory, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences. Yu Cao is with University of Chinese Academy of Sciences. Xubin Feng and Meilin Xie are with Space Precision Measurement Laboratory, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences. Xingjun Zhang is with the School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an 710049, China. Xueming Qian (qianxm@mail.xjtu.edu.cn) is with the Ministry of Education Key Laboratory for Intelligent Networks and Network Security, School of Information and Communications Engineering, and SMILES LAB, Xi'an Jiaotong University, Xi'an 710049, China.

This work is supported in part by the NSFC under Grant 62103317, 61772407 and 61732008. This work is partly supported by China Postdoctoral Science Foundation under Grant 2021M702600. This work is partly supported by Natural Science Foundation of Shaanxi Province under Grant 2021JQ-058, Beilin District Science and Technology Program GX2130, and Pazhou Lab, Guangzhou.

The hardware solution to depth map is Kinect or LiDAR. Kinect's depth map estimation is cheaper in price, but not that accurate compared with LiDAR. In addition to hardware solutions, there are also estimation methods that are based on softwares and algorithms. Although the depth estimation accuracy obtained by algorithms may be not as good as that of the hardware, the advantage is that no additional cost is needed to purchase the hardware equipment. In other cases where the space of the equipment is limited, the LiDAR cannot be arranged on the equipment due to the space limitation, while the compact ordinary camera with proper algorithms can meet the space requirement.

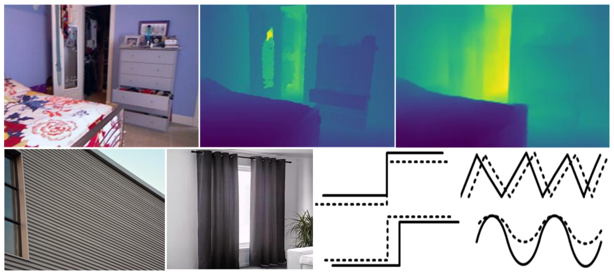


Fig. 1. Existing methods are able to work well in most circumstances. First row depicts an original RGB image, its ground-truth depth map and its estimated depth map. The challenge during depth estimation occurs at the areas with sudden depth changes, such as: (1) the edges between the cabinet and the wall, (2) the edge between the door and the wall, (3) edges of small objects. Second row presents a variety of depth changes, such as: wave type (window curtain), step type (surface of an uneven wall). All of these types correspond to objects in the real world.

With the development of deep learning, convolutional neural network (CNN) is applied in many computer vision tasks, including depth estimation [5], [6]. Usually, when CNNs become deeper, the resolution of feature maps becomes lower. In order to recover the resolution of depth map, an encoder-decoder architecture is usually employed in CNN based methods [5], [6]. Some methods use conditional random field to recover depth map and achieve success [7], [8].

Among existing the deep learning based depth estimation methods, depth information is represented by a per-pixel depth map. Such depth map representation is fragile facing depth changes. Fig.1 illustrates the original images and their ground-truth depth maps, where different kinds of depth changes are shown. The depth map estimated by one of the state-of-the-art method (Fu [9]) does not preserve detail information well. Specially, its estimated depth map is not that reliable when facing edges of objects and mirror-like surfaces.

Deep model training errors have much influence on the estimated depth map. Existing approaches directly estimate a per-pixel depth map for an image. Under this architecture, **the system training noise will directly influence the estimated depth map, especially the depth edges where obvious depth changes occur.** This results in distortions in edge areas of estimated depth maps. For inner areas of an object, such depth distortion is less severe. However for edges of objects, accurate depth estimation is a challenging issue due to sudden changes in depth, model training errors, object scale variations, etc.

In this work, we seek a different route for depth estimation. We propose a Compressive Sensing based Depth Representation (CSDR), which formulates the task of estimating depth maps in pixel space into a task of vector regression in representation vector space. CS theory [10] states that pair wise distances in the original space can be approximately maintained in the compressed space. So, although CSDR performs output space encoding for the purpose of depth representation, the subsequent CNN still aims at the original space under an equivalent distance measure. The principle behind the CSDR scheme is straightforward. CS is responsible for converting depth map to a fixed-length vector. For each training image, there is a vector generated. Then the CNN for vector regression is trained between the training image and its corresponding depth representation vector. During testing, the regression network estimates such vector for every input.

In addition to CSDR, our method is inspired by the following observations. 1) Existing loss functions calculate the difference between the ground truth and the estimated depth map on every single pixel independently. To better capture depth changes, it is necessary to consider the difference of curvature in both horizontal direction and vertical direction between ground truth and estimation. 2) Feature maps of CNNs represent the network's different understanding of an image. Shallow layers have small receptive field and large resolution inputs, they focus more on details of objects. Deep layers have large receptive field and can capture global features of an image. So it is better to use multiscale feature maps. 3) During feature maps decoding, multiple step decoding (between small sizes and original sizes) is smoother than single step decoding. At the same time, shortcut connection can make feature fusion more efficient.

Under the framework of CSDR, this work develops a Multiscale Encoder & Multiscale Decoder (MEMD) based depth estimation model. In existing models that use encoder-decoder architecture, single scale features are used. When decoding, small scale feature maps are decoded to their original sizes by linear interpolation. As mentioned previously, shallow features can better capture details of objects; deep features carry more global information. This work performs feature fusion between deep and shallow features by MEMD to improve the accuracy of depth estimation, especially when sudden depth change occurs. The method behind MEMD is different from U-Net [11]. Unlike U-Net, in the decoding stage of MEMD, the next scale decoder not only uses the features of this scale, but also uses the features of the previous scale. A dense fusion (DF) module is developed to fuse features of different scales. By introducing the shortcut connections mechanism, fusion module can receive the output not only from the adjacent layers, but also from all previous layers, which enriches the input of later layers. Then, we feed fused features of different scales into the refinement module to predict final depth maps.

Still under the framework of CSDR, this work develops a new loss function for depth estimation. The most commonly used loss function for depth estimation is to calculate the per-pixel direct difference between ground-truth and estimation (e.g. mean square loss). Some researchers also add the loss function based on the gradient difference and normal vector difference between depth maps. Hu [6] proposed that gradient and normal vector constraint can be applied to improve the accuracy. In this work, the difference of curvature in horizontal direction and vertical direction between ground truth and estimation is added as a part of our overall loss function. Thus, a depth map is cut by the plane in horizontal direction and vertical direction respectively. The curvature loss along the horizontal direction and the curvature loss along vertical direction are calculated respectively and contribute to overall loss independently. In this way, the new loss function helps to recover the depth value and maintain outlines of objects at depth edges.

The *contributions* of this paper are summarized as.

(1) We propose Compressive Sensing based Depth Representation (CSDR), which formulates the problem of depth estimation in pixel space into the task of fixed-length vector regression in representation space.

(2) We propose a Multiscale Encoder & Multiscale Decoder (MEMD) based model, which preserves higher resolution to both details and global patterns of depth maps. A multiscale feature fusion method is also introduced in the decoder.

(3) We further improve the loss function of depth estimation. The proposed method calculates the difference of curvature in both horizontal direction and vertical direction between ground truth and estimation during model training.

II. Related work

Depth estimation is the problem of estimating depth information from RGB images. Because of its wide application, many researchers have been studying in this field in recent years. The depth map obtained by depth estimation can be used in 3D reconstruction [2], semantic segmentation [4], augmented reality [3] and other fields. Depth data carries valuable information. [12] focuses on the problem of 3-D human detection in complex 3-D space using depth data only. LiDAR can not work well on mirror-like surfaces, such as window glasses, water area. If such surface is facing LiDAR at a certain angle, LiDAR can not capture the reflected signal, resulting in huge depth estimation inaccuracy at the corresponding position.

Since the process of getting 2D depth map from 3D object is irreversible, depth estimation is an ill-posed problem. In order to solve the ambiguity caused by the projection process, in recent years, the method based on deep learning usually adopts the encoder decoder structure to fuse the features of different scales. The function of encoder is to extract multiscale features gradually. The decoder uses multistage

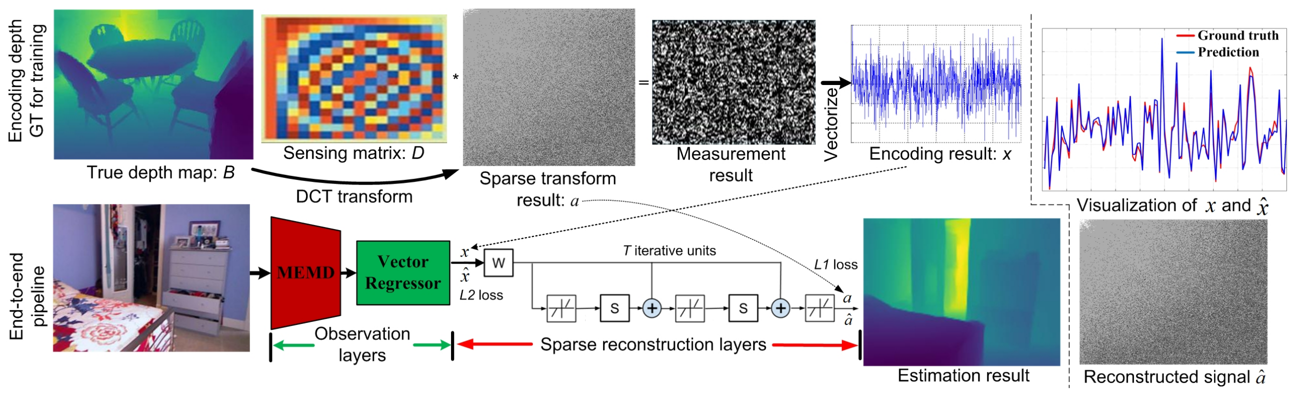


Fig. 2. System overview of the proposed approach. Top row: the proposed compressive sensing based depth representation scheme. Bottom row: the proposed end-to-end depth estimation network and visualizations of critical results.

upsampling to gradually restore feature maps to original sizes. [13] develops a novel depth-guided affine transformation to refine the depth features, since the quality of initial depth features is low. Multiple active depth cameras would interfere with each other, this degrades the quality of depth images significantly. [14] analyzes factors affecting depth accuracy and improves the accuracy of depth images by utilizing multiple projected patterns.

Eig [15] proposed a model using two stacked depth neural networks: one is to make a rough prediction of the whole and get the general trend of the depth map; the other is to further refine the rough depth map of the whole in the previous stage. In terms of the improvement of local details, researchers proposed many methods. Xu [8] integrated conditional random field into convolutional neural network to improve the resolution of output depth map in local detail. Li [7] proposed to use conditional random field as a post-processing method. Xu [16] introduced attention mechanism to improve the result of depth estimation.

Most of the previous methods regard the depth estimation problem as a regression problem. The method of Fu [9] is different from the previous method. They transformed the regression problem into a classification problem and designed corresponding classification loss function. Zheng proposed a new softmax loss function, which is different from the general classification loss function and is order sensitive. Laina [5] proposed a full convolution structure including residual learning, which can model the mapping between monocular image and depth image. They propose a new and effective method of learning feature map upsampling in full convolution network, which can improve the resolution of output. [17] designed an end-to-end trained lightweight convolutional network to infer depths from light fields. An attention module is proposed to better recover details at occlusion areas.

Similar to the observation in [18], we also find that depth maps predicted by existing self-supervised methods tend to be blurry with many depth details lost. To overcome this limitation, [18] paid efforts to obtain per-pixel depth maps with shaper boundaries and richer depth details. They use a multi-level feature extraction strategy to learn rich hierarchical representation. Then, a dual attention strategy is proposed to intensify the obtained features both globally and locally, resulting in improved depth maps with sharper boundaries. [19] also found that most existing approaches treat depth prediction as an isolated problem without considering high-level semantic context information. To ameliorate this issue, [19] develops a scene-aware contextualized convolution neural network, which characterizes the semantic context relationship at the class-level and refines depth at the pixel-level. [20] demonstrates a trade-off between a consistent scene structure and the high-frequency details (e.g. edges of objects). This shares a similar goal as our paper. To do that, [20] presents a double estimation method to merge low-resolution and high-resolution estimations. In addition, [20] designs a patch selection method that adds local details to the final result.

For saliency detection in RGB-D image, depth information also plays a critical role in distinguishing salient objects from cluttered backgrounds. Low depth qualities may result in a degraded detection performance in RGB-D images. To improve saliency detection in RGB-D images, [21] developed a two-phase depth estimation and selective deep fusion scheme. Slightly similar to our motivations, [22] also aims to improve the depth estimation quality in detailed areas and edges. [22] leverages both the low-level and high-level features from multiple input images and generates appealing results, especially with sharp details.

III. The Proposed Method

A. System Overview

The proposed approach is made up of three components: (1) a Compressive Sensing based Depth Representation (CSDR) method, (2) a Multiscale Encoder & Multiscale Decoder (MEMD), (3) a curvature based loss function. Fig.2 presents the overall framework of our approach.

The MEMD model consists of three parts: (1) an encoder composed of multiscale feature extraction network; (2) a multiscale multistage decoder composed of up sampling layers and convolution layers; (3) a refinement module composed of convolution layers. To encode training labels, CSDR scheme converts depth map to compressed vector representation. When testing, the observation layers of the network estimate a CSDR vector for every test image. Then, sparse reconstruction layers of the network estimate the pixel level depth map.

B. Depth Map Representation and Reconstruction

This section encodes the ground truth depth map into a dense vector, which our MEMD is trained to estimate. The representation scheme consists of Discrete Cosine Transform (DCT) [23] followed by a compressive projection.

Regarding the encoding scheme shown in Fig.2, B denotes a ground truth depth map with size $h \times w$. In the first step of encoding, DCT converts B to a matrix a. Because the matrix a is sparse, we apply compressive sensing based encoding to compresses the sparse matrix a into a much denser vector x with a sensing matrix D by:

$$x = Da, \tag{1}$$

where D is a $m \times n$ random Gaussian distribution matrix (every element is under independently and identically distribution, zero-mean Gaussian with variance $1/m$), with typically $m \ll n$. CS theory [10] claims that given x and D, a convex optimization can reconstruct a, given that the sensing matrix D can satisfy a restricted isometric property (RIP) condition and $m \geq C_m klog(n)$, where C_m is a small constant larger than one and k is the maximum number of non-zero elements in a.

Given D and x, the reconstruction of a depends on a convex optimization of joint L_1 norm and L_2 norm:

$$\min_a \frac{1}{2}\|x - Da\|_2^2 + \lambda\|a\|_1, \tag{2}$$

where the L_1 norm part is a penalty item, λ is a non-negative weight balancing the two items in the cost function (2). Various existing algorithms are able to optimize (2). Examples are orthogonal matching pursuit (OMP) [24] and dual augmented Lagrangian (DAL) [25]. In this paper, we choose to realize the reconstruction process by a recurrent neural network rather than simply using existing algorithm, in purpose of making the whole network end-to-end trainable.

The trainable sparse decoder is able to approximate sparse codes as in (2) over training set using the stochastic gradient descent method. The architecture of the encoder is denoted as $a = f(x, W)$, where W represents all the trainable parameters of the encoder. During training, the stochastic gradient descent method is performed to minimize a loss function $\mathcal{L}(W)$, which measures the squared error between the predicted code and the optimal code averaged over a training set $\left(x^1, \ldots, x^P\right)$:

$$\mathcal{L}(W) = \frac{1}{P}\sum_{p=0}^{p-1} L\left(W, x^p\right) \tag{3}$$

$$L\left(W, x^p\right) = \frac{1}{2}\|\hat{a}^p - f\left(W, x^p\right)\|^2 \tag{4}$$

where $\hat{a}^p = \arg\min_a E\left(x^p, a, D\right)$ is the optimal code for input instance x^p. The training method follows the stochastic gradient descent:

$$W(n+1) = W(n) - \eta(j)\frac{dL\left(W, x^{(n \mod P)}\right)}{dW} \tag{5}$$

where $\eta(n)$ equals to $1/n$ to guarantee convergence.The optimal sparse code is obtained by iterating: $a(k+1) = h_\alpha\left(W_e x - Sa(k)\right)$, with $a(0) = 0$, where x is the input, h_α is a coordinate wise shrinking function with threshold α. $W_e = D^T$ is the transpose of the dictionary matrix D, and $S = D^T D$.

The "sparse reconstruction layers" in Figure 2 illustrates the structure of the trainable sparse decoder, which is based on a time unfolded recurrent neural network (RNN) with 3 iterative units. For the reconstruction layers, we use the differentiable iterative shrinkage and thresholding architecture to approximate the sparse vector by a recurrent neural network with a fixed number of iterations (T). The sparse reconstruction layers include trainable parameters W_e and S.

The matrices W_e and S, are trained to minimize the estimation error to the optimal sparse code. The method allows us to impose restrictions on S in order to further reduce the computational complexity (e.g. keeping many terms at 0, or using a low-rank factorized form). Parameters $W = (W_e, S, \theta)$ are iteratively optimized over a set of training samples. The gradient $dL(W, x^p)/dW$ is computed during the back-propagation phase. We implement the trainable sparse code predictor using the deep learning library Pytorch, on which the CNN based regressor is also implemented.

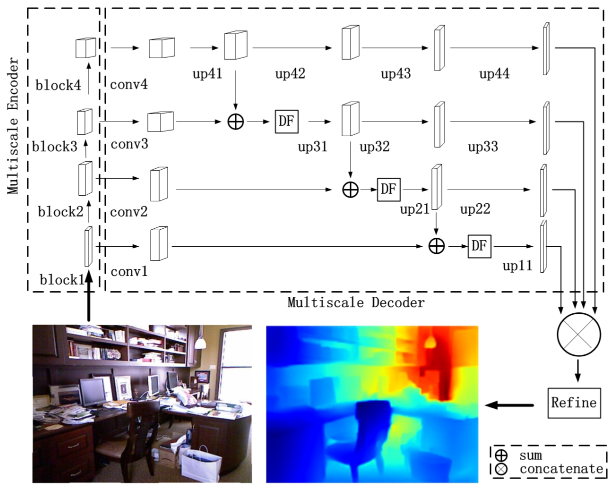


Fig. 3. Network architecture of the proposed multiscale encoder & multiscale decoder. DF stands for the dense fusion module.

C. Multiscale Encoder & Multiscale Decoder

The following part details the MEMD network proposed in this paper. The multiscale encoder can be composed of different types of backbone networks, such as ResNet [26], DenseNet [27], SENet [28], and the performance of models using different backbone networks is tested in Section IV-C. Our MEMD network makes use of the feature maps of different layers of backbone network in order to learn the global features without losing the local detailed features. The MEMD network uses pre-trained weights to initialize backbone network, and then finetune the model in depth estimation dataset. Taking the SENET-154 [28] backbone network as an example, the encoder of MEMD network uses 1/4, 1/8, 1/16, 1/32 size feature maps as the input of the decoder. In the backbone network, the number of channels in the feature maps increases in the order of 256, 512, 1024 and 2048, while their size decreases gradually. The feature maps of different scales extracted by the multiscale encoder are used as the input of the multiscale decoder for further processing. The multiscale decoder consists of convolution layers, upsampling layers [5], and DF module. The feature maps are processed by CNNs with kernel size of 1. The size of these feature maps is fixed while the number of channels is halved.

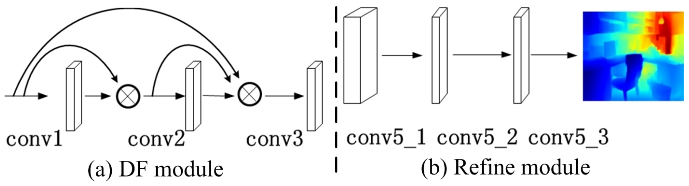


Fig. 4. Structure of Dense Fusion (DF) module and Refine module.

After the output of conv4 is upsampled by up41, on the one hand, it directly passes through up42, up43 and up44 to increase the size gradually to restore the original resolution; on the other hand, it is directly added with the output of conv3 and sent to DF module. In Fig.3, we adopts the upsampling method proposed by Laina [5]. The upsampling layers in Fig.3 first increase the size of feature maps by bilinear interpolation and then smooth them by CNNs.

The structure of DF module is shown in Fig.4, which is based on the design of Densenet [27]. The input of DF module is the the sum of the feature map from this scale and the adjacent smaller scale feature map after passing through the upsampling layer. DF module is composed of several convolutional layers with kernel size of 3. The input of deeper layers in DF module not only includes the output of direct upper convolution layer, but also includes all the output from indirect upper layers. This is equivalent to the introduction of shortcut connection. The later layers can receive not only deep level information, but also shallow level information.

The output size of DF module from this scale is doubled by upsampling. On the one hand, the output of DF module will continue to be upsampled step by step and finally it will be recovered to the original size. On the other hand, it will be part of the input of DF module in adjacent larger size branches, as shown in the multiscale decoder in Fig.3. The multiscale decoder in this paper recovers the depth information gradually from coarse to fine. The feature maps obtained from deeper convolutional layers represent more abstract information, and the recovered depth map is rough and more related to the global information. The feature maps obtained by shallow layers have a better understanding of image details, because these feature maps have input of higher resolution.

In the decoding process, the decoder of each scale not only receives the feature map of this scale, but also receives the feature map of the previous scale and fuses it with feature map from this scale, and transmits the fused feature to the next scale. It can be seen that the depth estimation method in this paper is a method that gradually recovers the global and local depth information of image from coarse to fine.

The multiscale decoder outputs four feature maps of the same size. Refine module is the last part of MEMD network and predicts final depth map. The structure of refine module is shown in Fig.4. It consists of three convolutional layers with kernel size of 5, batch normalization layers and activation layers. Table.I shows the number of input and output channels and the size of output feature map of each module.

TABLE I
OUTPUT SIZE OF DIFFERENT MODULES IN MEMD NETWORK.

module name	input channels	output channels	output size
block1	64	256	57x76
block2	256	512	29x38
block3	512	1024	15x19
block4	1024	2048	8x10
conv1	256	128	57x76
up11	128	64	114x152
conv2	512	256	29x38
up21	256	128	57x76
up22	128	64	114x152
conv3	1024	512	15x19
up31	512	256	29x38
up32	256	128	57x76
up33	128	64	114x152
conv4	2048	1024	8x10
up41	1024	512	15x19
up42	512	256	29x38
up43	256	128	57x76
up44	128	64	114x152
Refine	256	1	114x152

D. Limitations of the Existing Loss Function

The most commonly used loss function of early depth estimation method is the difference between the estimated value d_i and ground truth value g_i of depth:

$$l_1 = \frac{1}{n}\sum_{i=1}^{n} e_i \tag{6}$$

where n is the number of training images. e_i is defined as $e_i = |d_i - g_i|$. Under such loss function, the difference of $1mm$ between the predicted value and the ground truth value has the same effect on the final loss at different depths. Lee [29] pointed out that the value of the loss function should be larger in the near place and smaller in the far place. Inspired by Lee [29], Hu [6] modified Equation (6) by logarithmic function:

$$l_{depth} = \frac{1}{n}\sum_{i=1}^{n} F(e_i) \tag{7}$$

where $F(x)$ is defined as following:

$$F(x) = \ln(x + \alpha) \tag{8}$$

where α is manually set parameter.

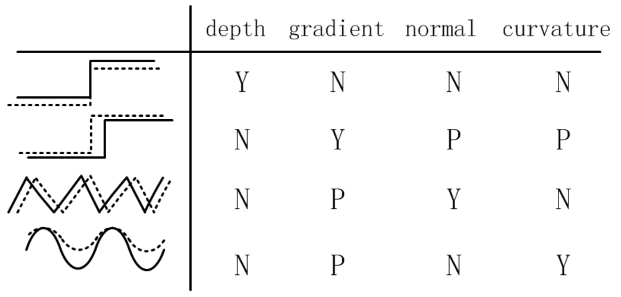
The above loss function is only sensitive to the difference of depth value, not to the sudden change of the depth in the scene, not to the change of the normal vector, not to the different curvature of the surface. Fig.5 shows the sensitivity of different types of loss functions to depth changes. The figure shows one-dimensional depth value difference, while the depth map is two-dimensional, this does not affect the analysis, because the two dimensions can be calculated separately. The solid line and dotted line on the left represent the ground truth and the estimated value of depth map respectively. “Y”, “N” and “P” indicate the responding level of a loss function to a type of depth change. Details can be found in the caption of Fig.5.

Huang [30] made statistics on the natural image, and found that there are many such jumps of depth values in natural images, which reflects the gradient changes in the depth map. Such jumps can also be seen from the true value of dataset. Fig.6 shows the RGB image and corresponding depth map of some NYU depth V2 datasets. It can be seen that there is indeed a jump of depth value in the depth map.

Therefore, it is necessary to improve the loss function of depth estimation. Hu [6] think that the traditional loss function is not sensitive to the error at the edge of the object, so it can not describe the depth value difference at the contour of the object, and propose a gradient loss to punish this type of loss:

$$l_{grad} = \frac{1}{n}\sum_{i=1}^{n}(F(|\nabla_x(d_i)-\nabla_x(g_i)|)+F(|\nabla_y(d_i)-\nabla_y(g_i)|)) \tag{9}$$

In Equation (9), ∇_x and ∇_y represent gradients of depth map in x direction and y direction respectively. This loss function is sensitive to the sharp variation of the depth value at the edge of the object, as shown in Fig.5. The loss functions from Equation (9) and (6) are complementary to each other.



	depth	gradient	normal	curvature
	Y	N	N	N
	N	Y	P	P
	N	P	Y	N
	N	P	N	Y

Fig. 5. Different types of loss are sensitive to different types of depth changes. “Y” indicates that a loss function is able to respond to this type of depth change; “N” means the loss function can not respond to the depth change; “P” stands for that the loss function only partially respond to the depth change.

Besides, Hu [6] proposed to use the difference of normal value between ground truth value and predicted value:

$$l_{normal} = \frac{1}{n}\sum_{i=1}^{n}(1-\frac{<n_i^d, n_i^g>}{\sqrt{<n_i^d, n_i^d>}\sqrt{<n_i^g, n_i^g>}}) \tag{10}$$

where $<\cdot,\cdot>$ is inner product of vectors, n_i^d and n_i^g are normal values of predicted value and ground truth value respectively.

E. Curvature Loss Function

After adding gradient difference and normal vector difference to the loss function, the difference between the true value and the predicted value is better described. However, there is still room for further improvement in the loss function of depth estimation. Neither gradient difference nor normal vector difference can capture the curvature difference between the true depth map and the predicted depth map. In a real scene, the surface of an object has not only a plane, but also a curved surface, such as a spherical object or a cylinder like object. From a geometric point of view, it is the difference in the degree of curvature between the plane formed by the true value and the predicted value. Therefore, another loss function is needed to describe the difference and add it to the final loss function in order to punish such difference. So we add curvature difference loss which is related to the difference between the true value and the predicted value at a certain point to the final loss function.

Because the image pixels are discrete, in the experiment of this paper, we use three points to fit a curve, and then calculate the curvature of the curve. The curvature difference loss proposed in this paper is represented as:

$$l_{curvature} = \frac{1}{n}\sum_{i=1}^{n}(F(|\kappa_x(d_i)-\kappa_x(g_i)|)+F(|\kappa_y(d_i)-\kappa_y(g_i)|)) \tag{11}$$

where κ_x represents the curvature in x direction, κ_y represents the curvature in y direction.

If a plane parallel to the x direction is used to cut the surface formed by the true value depth map, a curve can be obtained from the intersection of two planes; similarly, a plane intersects with the predicted depth map, a curve can also be obtained.

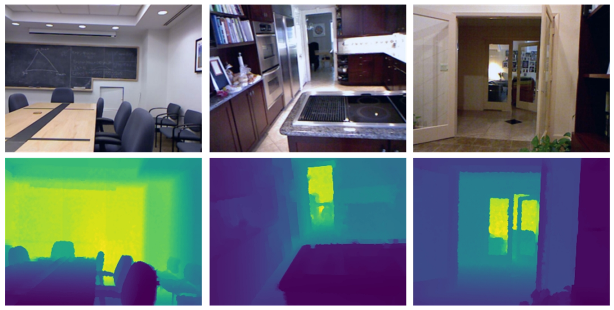


Fig. 6. Gradient changes of depth map in natural scenes.

As shown in Fig.5, two curves may have different curvature at the same point. Solid lines show the true depth edges, dotted lines depict the estimation. Each vertical axis is depth degree, its horizontal axis is along x or y direction of an image. Loss functions have distinct sensitivities to different types of depth changes. The ”depth” loss is sensitive to shifts in depth degree (i.e. vertical axis), but insensitive to shifts in horizontal axis. The ”curvature” loss focuses on the difference between estimation and GT. The ”normal” loss allows a shift in horizontal axis, and is still able to compute the difference.

Equation (11) represents such difference. The Equation includes not only the curvature difference in x direction, but also the curvature difference in y direction. It includes not only the curvature difference at a certain point, but also the curvature difference between predicted depth map and the true value depth map at all points. The final loss function is composed of four kinds of loss functions:

$$L = l_{depth} + \lambda_1 l_{grad} + \lambda_2 l_{normal} + \lambda_3 l_{curvature} \tag{12}$$

where λ_1, λ_2 and λ_3 are weight factor, we will give the value of these weights in section IV-C.

IV. Experiment

A. Data and data preprocessing

We adopt three depth datasets: NYU-Depth-v2 [31], KITTI [32] and Make3D [33]. NYU-Depth-v2 contains 50000 pairs of RGB indoor scene images and depth maps as training data and 654 pairs of images as test data. KITTI contains outdoor scenes with images of resolution about 375×1241 captured by cameras and depth sensors in a driving car. All the 61 scenes from the "city", "residential", "road" and "Campus" categories are used as our training/test sets. We test on 697 images from 29 scenes split by Eigen et al. [15], and train on about 23488 images from the remaining 32 scenes. We train our model on a random crop of size 385×513. Make3D contains 534 outdoor images, 400 for training, and 134 for testing, with the resolution of 2272×1704. We reduce the resolution of all images to 568×426, and train our model on a random crop of size 513×385.

Due to the variety of real scene changes, we perform data augmentation to enhance the generalization ability of the model. The data augmentation method used in this paper includes rotation, translation and color jitter. We downsample the images from 640×480 to 320×240 and the crop images to the size 304×228.

B. Evaluation Criteria

Assuming the number of all effective pixels in the test set is S, we use the following criteria for depth estimation evaluation. Besides, edge accuracy introduced by [6] is also used as another evaluation criteria.

Root mean square error (RMS): $\sqrt{\frac{1}{S}\sum_{i=1}^{S}(d_i - g_i)^2}$.

Mean relative error (REL): $\frac{1}{S}\sum_{i=1}^{S}\frac{|d_i-g_i|}{g_i}$.

Mean log10 error (log 10): $\frac{1}{S}\sum_{i=1}^{S}|\log_{10}d_i - \log_{10}g_i|$.

Thresholded accuracy: ratio of points that satisfy $\max(\frac{d_i}{g_i}, \frac{g_i}{d_i}) = \delta < \theta$, where θ is threshold and its commonly used values are 1.25,1.25^2 and 1.25^3.

C. Comparison with state-of-the-art

We compare our proposed approach with 13 recent state-of-the-art methods including [19], [5], [34], [22], etc on three datasets: NYU-Depth-v2, KITTI and Make3D. Table.II, Table.III and Table.IV present the depth estimation performance comparison of all the approaches.

In all the experiments, our MEMD model uses the Imagenet pre-trained encoder [35]. We use Adam optimizer to train the model in an end-to-end manner, and sets the batch size to 8 during training. We set λ_1, λ_2 and λ_3 in Equation (12) to 1 and α in Equation (8) to 0.5.

From Table.II, it can be seen that the our method is better than the previous methods in terms of REL, log 10 and δ-threshold accuracy. Similar to our method, Laina et al. [5] also use an encoder-decoder model. Compared with [5], our method has smaller error rate and the higher accuracy. Fu [9] is better than our method in terms of RMS. But if considering the estimated depth map, their method is not quite good as ours in detail of depth map. It can be seen from Fig.9 that their depth estimation results are not good enough in small objects such as legs of chair, legs of table and wheels of bicycle. This is because our method uses the network structure of multiscale encoder and multiscale decoder. In the decoder, dense feature fusion module is used to fuse the features of different scales to make full use of the multiscale features. [34] is a strong competitor, which achieves the second best depth estimation results in terms of REL. Similar to our motivation, [34] proposed a robust training objective that is invariant to changes in depth range and scale, advocate the use of principled multi-objective learning to combine data from different sources. In comparison, we use improved loss function to train our model.

TABLE II
Depth estimation results of 9 methods on the testing set of NYU-Depth-v2 dataset. Red and Blue colors label the best and second best localization results in each column.

Method	REL	RMS	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Eigen [15]	0.205	0.607	0.611	0.887	0.971
Laina [5]	0.127	0.573	0.811	0.953	0.988
Li [7]	0.202	0.821	0.621	0.886	0.968
Xu [8]	0.163	0.586	0.811	0.954	0.987
Fu [9]	0.171	0.549	0.828	0.965	0.992
Qi [36]	0.188	0.569	0.834	0.960	0.990
Hu [6]	0.157	0.530	0.841	0.966	0.983
Li [17]	0.125	0.548	0.845	0.969	0.991
Song [19]	0.114	0.541	0.857	0.973	0.993
Zhang [22]	0.115	0.525	0.866	0.975	0.991
Ranftl [34]	0.114	0.535	0.851	0.977	0.993
Ours	0.113	0.527	0.875	0.986	0.997
Ours(noCSDR)	0.164	0.598	0.645	0.869	0.930

TABLE III
Depth estimation results on the testing set of KITTI dataset.

Method	REL	RMS	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Eigen [15]	0.230	8.063	0.713	0.895	0.926
Make3D [33]	0.280	8.73	0.601	0.820	0.926
Fu [9]	0.234	8.205	0.747	0.926	0.923
Qi [36]	0.258	7.349	0.726	0.873	0.914
Hu [6]	0.237	7.578	0.688	0.894	0.916
Li [17]	0.231	7.725	0.694	0.902	0.939
Song [19]	0.190	7.156	0.747	0.945	0.943
Zhang [22]	0.186	8.034	0.703	0.913	0.928
Ranftl [34]	0.201	7.714	0.718	0.932	0.903
Ours	0.183	6.976	0.737	0.953	0.954
Ours(noCSDR)	0.234	8.205	0.632	0.839	0.907

Table.III presents depth estimation results on KITTI dataset. It can be seen that the proposed obtains a slight best performance over all the other comparison approaches including strong competitors [19] [34]. Furthermore, [19] gives quite promising REL value (0.190) among the comparison approaches. This supports their motivation that it is important to characterize a scene-aware context relationship at class-level and refines depth at pixel-level. Another quite competitive approach is [22] (REL=0.186), which leverages both low-level and high-level features and generate appealing results, especially for sharp details. The proposed method achieves REL=0.183 with close to 0.02 REL below [34].

Table.IV presents depth estimation results on Make3D

TABLE IV
DEPTH ESTIMATION RESULTS ON THE TESTING SET OF MAKE3D DATASET.

Method	REL	RMS	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Eigen [15]	0.305	12.350	0.758	0.874	0.937
Make3D [33]	0.370	11.326	0.799	0.887	0.927
Fu [9]	0.279	10.798	0.782	0.905	0.936
Qi [36]	0.314	11.006	0.777	0.907	0.920
Hu [6]	0.281	11.061	0.781	0.903	0.924
Li [17]	0.276	10.742	0.740	0.917	0.948
Song [19]	0.306	11.085	0.743	0.920	0.915
Zhang [22]	0.253	10.891	0.872	0.915	0.951
Ranftl [34]	0.267	10.552	0.886	0.921	0.968
Ours	0.260	10.491	0.869	0.920	0.955
Ours(noCSDR)	0.313	11.842	0.751	0.836	0.879

dataset. [34], [22] and our method perform quite well on Make3D dataset. [34] achieves the lowest REL=0.253 and demonstrates a trade-off between a consistent scene structure and high-frequency details (e.g. edges of objects. This shares a similar goal as our paper. This partly explains the improvements of our method and [34] over other methods. Besides, [34] presents a double estimation method to merge low-resolution and high-resolution estimations. While, we seek a different route for depth estimation. We perform depth estimation by regression in CSDR space rather than directly predicting depth value for each pixel.

To visualize the depth estimation results, Fig.9 shows the comparison results of depth maps estimated by different methods. It can be seen that the depth maps estimated by Zhang [22], Song [19] are more distorted, and the depth maps obtained by Fu et al. [9], Li [17] are not as delicate as our method at edges of objects. Because our method adopts a dedicated depth change representation module for edges. And our encoder-decoder structure makes full use of multi-scale feature maps. In the decoding phase, the feature map of one scale is sent to DF module for fusion and then decoded step by step to recover the feature of depth map of this scale.

The estimation of the final depth map is completed by the refine module proposed in this paper, so that the model makes full use of the deep and shallow features. Deep features and shallow features focus on the global information and local information of the image respectively. The predicted depth map not only reflects the overall depth trend of the image, but also recovers the texture in detail. At the same time, the curvature difference loss function is added in this paper, which makes depth map obtained by our method can retain the details better when there are small objects.

D. Ablation study

We carry out ablation experiments to better understand the effect of major components of the proposed approach.

1) Effect of CSDR: To explore the contribution of CSDR, we perform an experiment on the effect of CSDR. In our proposed approach, CSDR is responsible for converting depth map representation to fixed-length vectors representation. For a training image, there is a such vector generated. After that, a regression-oriented neural network is trained between training images and their representation vectors. During testing, the regression neural network estimates this vector for every testing image. It is during the vector inference process that inevitable system error occurs.

In Fig.7, ”a” visualizes the effect of system error. The red curve indicates the ground truth according to CSDR, the blue curve represents the vector estimated by the regression network. Please note the differences between the red curve and the blue curve. The differences are obvious between their peaks, bottom areas and general trends. The differences result from the existence of the system error.

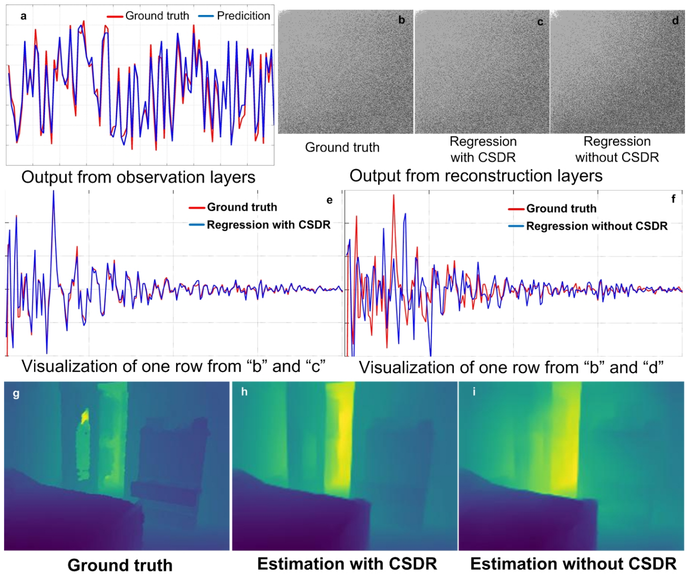


Fig. 7. How CSDR influences depth estimation from the view of CSDR vector space and from the view of depth map space.

The system error exists regardless of whether CSDR is used or not. For existing depth estimation approaches, they do not conduct CSDR-like representation conversion between vector space and pixel space. They purely work in pixel space and directly estimate per-pixel depth map. Thus, the system error can directly influence the pixel values of depth maps. But in the proposed approach, CSDR stops the direct transmit of the system error to depth maps. Fig.7(b)(c)(d) present the reconstructed depth signal by our approach with CSDR and without CSDR. Every reconstructed depth signal corresponds to a depth distribution in pixel space, since DCT transform is used here. Generally speaking, CSDR is able to approximate the ground truth in terms of reconstructed depth signals.

To better understand details, Fig.7(e)(f) illustrates the same row of Fig.7(b)(c)(d). It can be seen that, CSDR maintains the patterns of the ground truth curve and carries valuable information. However without CSDR, the patterns of the predicted curve shows much more distortion compared with the ground truth. At last, Fig.7(h)(i) give the final depth estimation results. As mentioned previously, the patterns of the reconstructed signal in Fig.7(d) have been interfered. Consequently, one can see more obvious depth inaccuracy and distortion in Fig.7(i) than that in Fig.7(h).

With CSDR between pixel space and vector space, the patterns of the curves that carry valuable depth information are maintained. In comparison, without CSDR, the patterns

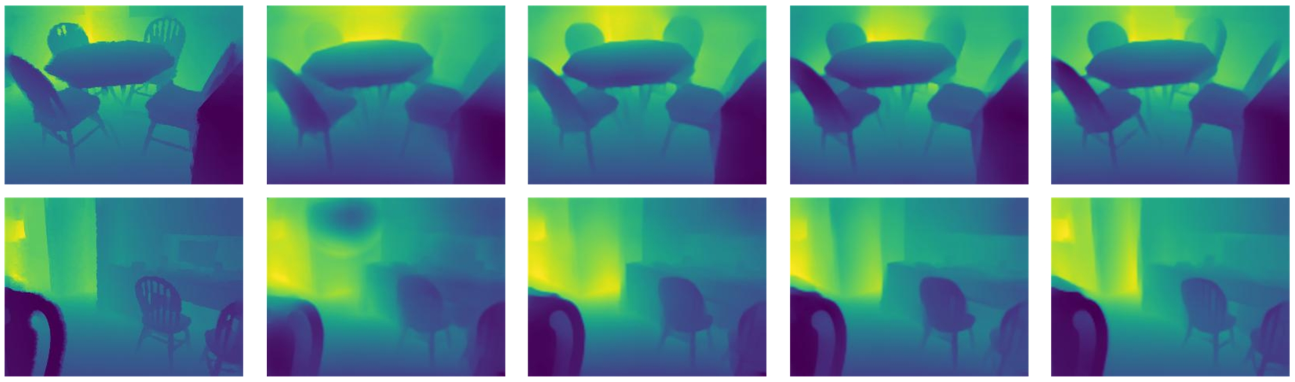


Fig. 8. Depth maps estimated by models trained with different loss functions. The 1st to 5th column depicts the Ground truth, l_{depth}, $l_{depth} + l_{grad}$, $l_{depth} + l_{grad} + l_{normal}$ and complete loss respectively.

of the curves show much more distortion compared with the ground truth. More details regarding the above explanation are available in Fig.7. After a range of experiments and analysis, we claim that CSDR-based depth representation shows more robustness to the system noise, compared with the per-pixel depth map representation. The ”Ours (noCSDR)” row in Table.II presents the localization performance without CSDR. We can observe the big performance gap between ”Ours” and ”Ours (noCSDR)”.

2) Effect of Depth Change: Table.V shows the edge accuracy of different methods under different thresholds. Because the definition of edge point is to define the point that satisfies $\sqrt{f_x(i)^2 + f_y(i)^2} > \eta$ as edge point, that is to say, the point with large gradient in x direction and y direction is defined as the edge point, so such point is generally the edge of the object, such as the edge of the table, the outline of the chair, etc. Therefore, the higher precision (P), recall (R) and F1 score, the better the depth map can be recovered at the boundary of the object, which shows that the method can better reflect the real depth map at the details. One example is the method of Fu [9]. Although RMS value is relatively low, that is, the error on test set is relatively small, the depth map obtained is not very good in object details, which is shown in Table.V as the corresponding score is relatively low. This also reflects the effectiveness of taking edge accuracy as a complementary of other evaluation criteria.

3) Effect of Curvature Difference Loss: Now we will discuss the function of DF module and curvature difference loss function proposed by us.

Table.VII shows the results with DF module and without DF module. It can be seen from the Table that after DF module is removed, the errors increase and the accuracy decreases. This shows that the DF module proposed in this paper can effectively fuse the features of different scales and improve the performance of the model.

Fig.8 shows the results of training the MEMD model in this paper by using different loss functions and estimating the depth map by using the trained model. It can be seen that with the increase of loss function types, the depth map predicted by the model is more accurate, and the corresponding depth map can be recovered well at places with details, such as the edge of objects and the outline of small objects. The quantitative result in Table.VI is consistent with the visualization result in Fig.8. A combination of loss types bring lower error rates to ? in REL and ? in RMS. Especially the pure introduction of curvature loss reduces ?cha in REL and ?cha in RMS, see the difference between row-4 and row-5 in Table.VI. This is because more kinds of loss functions can more accurately depict difference between the ground truth depth map and predicted depth map. The difference between them can make the model learn more abundant feature expression in the process of model training.

TABLE V
ACCURACY COMPARISON OF DEPTH EDGE ESTIMATION WITH 4 MOST COMPETITIVE METHODS. RED AND BLUE COLORS LABEL THE BEST AND SECOND BEST LOCALIZATION RESULTS IN EACH COLUMN.

Threshold	Method	P	R	F1
0.25	Fu [9]	0.489	0.435	0.454
	Li [17]	0.516	0.400	0.436
	Zhang [22]	0.320	0.583	0.402
	Song [19]	0.644	0.508	0.562
	Ours	0.664	0.529	0.569
0.5	Fu [9]	0.536	0.422	0.463
	Li [17]	0.600	0.366	0.439
	Zhang [22]	0.316	0.473	0.412
	Song [19]	0.668	0.505	0.568
	Ours	0.671	0.511	0.574
1.0	Fu [9]	0.670	0.479	0.548
	Li [17]	0.794	0.407	0.525
	Zhang [22]	0.483	0.512	0.485
	Song [19]	0.759	0.540	0.623
	Ours	0.787	0.545	0.627

TABLE VI
QUANTITATIVE ANALYSIS ON THE EFFECT OF CURVATURE LOSS.

Loss function	REL	RMS	$\delta < 1.25$	$\delta < 1.25^2$
depth	0.195	0.639	0.769	0.850
depth+grad	0.186	0.583	0.799	0.907
depth+grad+normal	0.157	0.552	0.821	0.931
complete	0.113	0.527	0.875	0.986

V. DISCUSSION

[Strength] (1) Our method find a different route (i.e. CSDR), which solves the problem of depth estimation by fixed-length vector regression. So that our method shows more robustness

TABLE VII
THE EFFECT OF DENSE FUSION (DF) MODULE.

Method	REL	RMS	log 10	$\delta < 1.25$	$\delta < 1.25^2$
With DF	0.113	0.527	0.048	0.868	0.976
Without DF	0.118	0.537	0.051	0.859	0.971

to sudden depth change compared with cutting-edge methods that purely estimate depth in pixel space. (2) We study how different types of depth changes affect the result of depth estimation. This is a new study in the research of depth estimation. And our proposed loss function calculates the difference of curvature in both horizontal and vertical direction.

[Limitation] One limitation is our computational complexity. Although our method gives state-of-the-art or mostly the best localization accuracy in experiments. However, our inference speed is relative low (approximately 3.5s per 640*480 size image). This is due to the existence of several iterative units in our sparse reconstruction layers, which bring slower forward inference speed. However, because weights of iterative units are shared, the total number of parameters of our method still remains at a relatively low level. CSDR scheme is a new route for depth estimation. In the future, we plan to improve the design of our sparse reconstruction layers (possible not to use iterative units) to speed up.

VI. CONCLUSION

This paper develops a depth estimation method to deal with the challenges from sudden depth changes, object scale variations, etc. The proposed method contains a Compressive Sensing based Depth Representation scheme, a Multiscale Encoder & Multiscale Decoder based model, and a new loss function based on curvature difference. Various experiments show that our approaches obtain obvious performance improvement over existing methods, including several quite competitive state-of-the-art approaches. Experiments support our insight that it is effective to cast depth map estimation as regression in representation signal space. We hope the insight may prove useful for other depth estimation related works.

REFERENCES

[1] T. Whelan, M. Kaess, H. Johannsson, M. Fallon, J. J. Leonard, and J. McDonald, "Real-time large-scale dense rgb-d slam with volumetric fusion," *The International Journal of Robotics Research*, vol. 34, no. 4-5, pp. 598–626, 2015.

[2] K. Li, Y. Wu, Y. Xue, and X. Qian, "Viewpoint recommendation based on object oriented 3d scene reconstruction," *IEEE Transactions on Multimedia*, vol. 23, no. 1, pp. 257–267, 2021.

[3] W. Lee, N. Park, and W. Woo, "Depth-assisted real-time 3d object detection for augmented reality," in *ICAT*, vol. 11, no. 2, 2011, pp. 126–132.

[4] S.-J. Park, K.-S. Hong, and S. Lee, "Rdfnet: Rgb-d multi-level residual feature fusion for indoor semantic segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4980–4989.

[5] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *2016 Fourth international conference on 3D vision (3DV)*. IEEE, 2016, pp. 239–248.

[6] J. Hu, M. Ozay, Y. Zhang, and T. Okatani, "Revisiting single image depth estimation: Toward higher resolution maps with accurate object boundaries," in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2019, pp. 1043–1051.

[7] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1119–1127.

[8] D. Xu, E. Ricci, W. Ouyang, X. Wang, and N. Sebe, "Multi-scale continuous crfs as sequential deep networks for monocular depth estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5354–5362.

[9] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2002–2011.

[10] D. J. Hsu, S. M. Kakade, J. Langford, and T. Zhang, "Multi-label prediction via compressed sensing," *CoRR*, vol. abs/0902.1284, 2009. [Online]. Available: http://arxiv.org/abs/0902.1284

[11] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[12] L. Tian, M. Li, Y. Hao, J. Liu, G. Zhang, and Y. Q. Chen, "Robust 3-d human detection in complex environments with a depth camera," *IEEE Transactions on Multimedia*, vol. 20, no. 9, pp. 2249–2261, 2018.

[13] Y. Zuo, Y. Fang, P. An, X. Shang, and J. Yang, "Frequency-dependent depth map enhancement via iterative depth-guided affine transformation and intensity-guided refinement," *IEEE Transactions on Multimedia*, vol. 23, pp. 772–783, 2021.

[14] Y. Pan, R. Liu, B. Guan, Q. Du, and Z. Xiong, "Accurate depth extraction method for multiple light-coding-based depth cameras," *IEEE Transactions on Multimedia*, vol. 19, no. 4, pp. 685–701, 2017.

[15] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Advances in neural information processing systems*, 2014, pp. 2366–2374.

[16] D. Xu, W. Wang, H. Tang, H. Liu, N. Sebe, and E. Ricci, "Structured attention guided convolutional neural fields for monocular depth estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3917–3925.

[17] Y. Li, Q. Wang, L. Zhang, and G. Lafruit, "A lightweight depth estimation network for wide-baseline light fields," *IEEE Transactions on Image Processing*, vol. PP, no. 99, pp. 1–1, 2021.

[18] X. Song, W. Li, D. Zhou, Y. Dai, and L. Zhang, "Mlda-net: Multi-level dual attention-based network for self-supervised monocular depth estimation," *IEEE Transactions on Image Processing*, vol. PP, 2021.

[19] W. Song, S. Li, J. Liu, A. Hao, and H. Qin, "Contextualized cnn for scene-aware depth estimation from single rgb image," *IEEE Transactions on Multimedia*, vol. PP, no. 99, pp. 1–1, 2019.

[20] S. M. H. Miangoleh, S. Dille, L. Mai, S. Paris, and Y. Aksoy, "Boosting monocular depth estimation models to high-resolution via content-adaptive multi-resolution merging," 2021.

[21] C. Chen, J. Wei, C. Peng, W. Zhang, and S. B. University, "Improved saliency detection in rgb-d images using two-phase depth estimation and selective deep fusion," *IEEE Transactions on Image Processing*, vol. PP, no. 99, pp. 1–1, 2020.

[22] Y. Zhang, S. Xu, B. Wu, J. Shi, and X. Zhang, "Unsupervised multi-view constrained convolutional network for accurate depth estimation," *IEEE Transactions on Image Processing*, vol. PP, no. 99, pp. 1–1, 2020.

[23] N. N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Transactions on Computers*, vol. C-23, no. 1, pp. 90–93, 2006.

[24] T. T. Cai and L. Wang, "Orthogonal matching pursuit for sparse signal recovery with noise." *IEEE Transactions on Information Theory*, vol. 57, no. 7, pp. 4680–4688, 2011.

[25] R. Tomioka, T. Suzuki, and M. Sugiyama, "Super-linear convergence of dual augmented-Lagrangian algorithm for sparsity regularized estimation," *Journal of Machine Learning Research*, vol. 12, no. 8, pp. 1537–1586, 2011.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[27] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[28] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[29] J.-H. Lee, M. Heo, K.-R. Kim, and C.-S. Kim, "Single-image depth estimation based on fourier domain analysis," in *Proceedings of the*

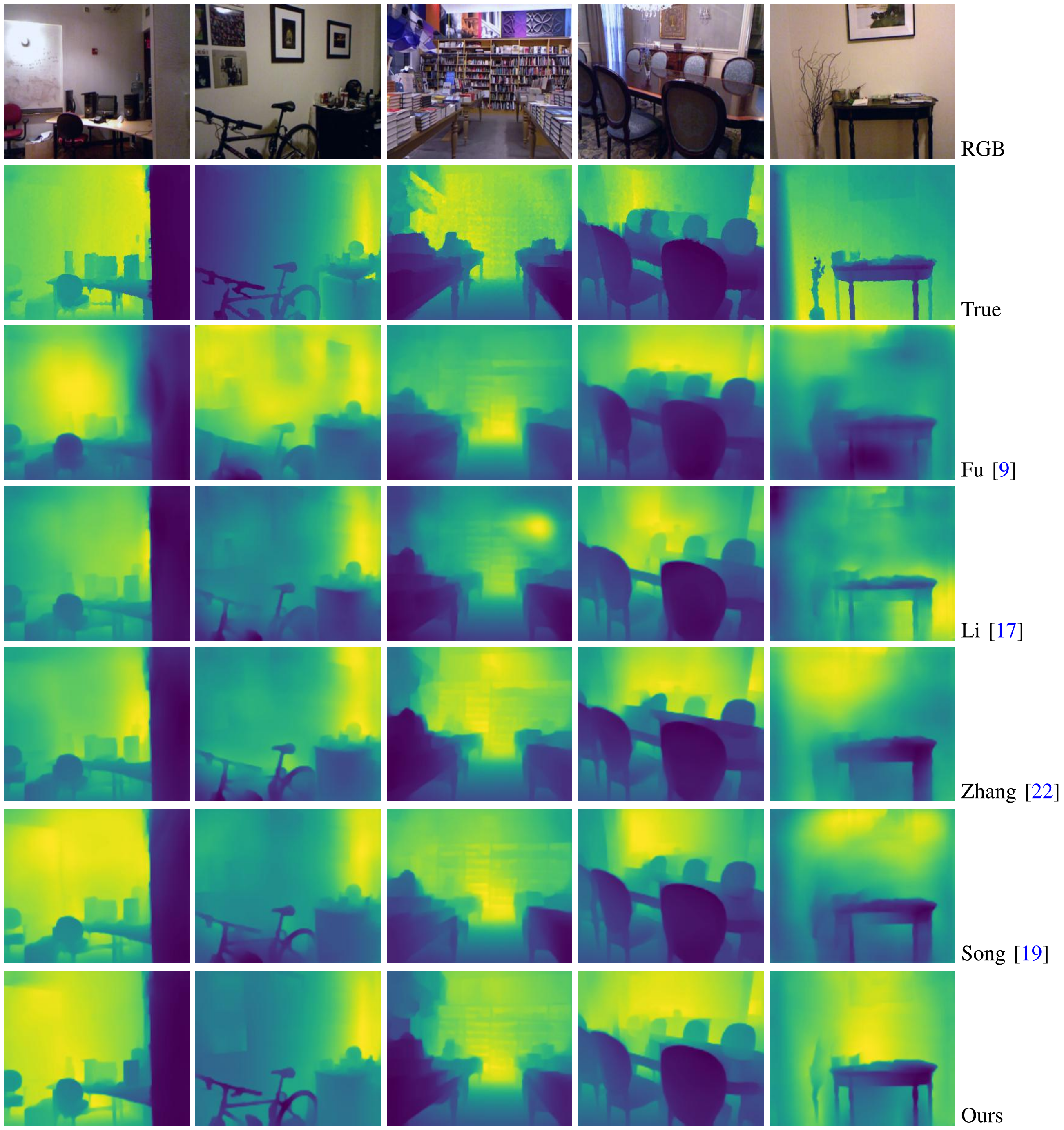


Fig. 9. Depth estimation results comparison with 4 most competitive methods.

IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 330–339.

[30] J. Huang, A. B. Lee, and D. Mumford, "Statistics of range images," in *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, vol. 1. IEEE, 2000, pp. 324–331.

[31] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *European conference on computer vision*. Springer, 2012, pp. 746–760.

[32] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[33] A. Saxena, M. Sun, and A. Y. Ng, "Make3d: Learning 3d scene structure from a single still image," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 824–840, 2009.

[34] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020.

[35] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," 2009.

[36] X. Qi, R. Liao, Z. Liu, R. Urtasun, and J. Jia, "Geonet: Geometric neural network for joint depth and surface normal estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 283–291.