Depth Super-Resolution via Deep Controllable Slicing Network

Xinchen Ye\textsuperscript{1,2,*}, Baoli Sun\textsuperscript{1}, Zhihui Wang\textsuperscript{1,2}, Jingyu Yang\textsuperscript{3}, Rui Xu\textsuperscript{1,2}, Haojie Li\textsuperscript{1,2}, Baopu Li\textsuperscript{4}

\textsuperscript{1}International School of Information Science & Engineering, Dalian University of Technology, China
\textsuperscript{2}Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, China
\textsuperscript{3}School of Electrical and Information Engineering, Tianjin University, China
\textsuperscript{4}Baidu Research, USA

ABSTRACT

Due to the imaging limitation of depth sensors, high-resolution (HR) depth maps are often difficult to be acquired directly, thus effective depth super-resolution (DSR) algorithms are needed to generate HR output from its low-resolution (LR) counterpart. Previous methods treat all depth regions equally without considering different extents of degradation at region-level, and regard DSR under different scales as independent tasks without considering the modeling of different scales, which impede further performance improvement and practical use of DSR. To alleviate these problems, we propose a deep controllable slicing network from a novel perspective. Specifically, our model is to learn a set of slicing branches in a divide-and-conquer manner, parameterized by a distance-aware weighting scheme to adaptively aggregate different depths in an ensemble. Each branch that specifies a depth slice (e.g., the region in some depth range) tends to yield accurate depth recovery. Meanwhile, a scale-controllable module that extracts depth features under different scales is proposed and inserted into the front of slicing network, and enables finely-grained control of the depth restoration results of slicing network with a scale hyper-parameter. Extensive experiments on synthetic and real-world benchmark datasets demonstrate that our method achieves superior performance.

CCS CONCEPTS

• Computing methodologies → 3D imaging: Reconstruction.

KEYWORDS

Scene Depth, Super-Resolution, Distance-Aware, Slicing, Controllable

ACM Reference Format:

Xinchen Ye\textsuperscript{1,2,*}, Baoli Sun\textsuperscript{1}, Zhihui Wang\textsuperscript{1,2}, Jingyu Yang\textsuperscript{3}, Rui Xu\textsuperscript{1,2}, Haojie Li\textsuperscript{1,2}, Baopu Li\textsuperscript{4}. 2020. Depth Super-Resolution via Deep Controllable Slicing Network. In Proceedings of the 28th ACM International Conference on Multimedia (MM ’20), October 12–16, 2020, Seattle, WA, USA. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3394171.3413874

1 INTRODUCTION

3D content of a given scene consists of two components, i.e., two-dimensional texture (color) image, and depth information of a third dimension. Scene depth map is essential and widely used as a basic element in many computer vision fields, such as 3D reconstruction \cite{37, 43}, autonomous navigation \cite{1, 11}, monitoring \cite{8, 10}, and so on. However, the acquisition of depth maps is still a challenging task in real conditions, which needs costive specialized equipment, e.g., Time-of-Flight (ToF) depth sensors. Due to the imaging limitation of these depth sensors, high-resolution (HR) depth maps are often difficult or even impossible to be acquired directly. Hence, effective depth super-resolution (DSR) algorithms are needed to yield HR output from the degraded low resolution (LR) counterpart. Recently, CNN-based methods \cite{13, 24, 40, 41} have been proposed to recover depth maps by learning a set of kernels or filters from data instead of hand-designed ones. Although these CNN-based methods present impressive performance, the task of DSR still needs to be improved because of its unsatisfactory performance in terms of accuracy and practicality.

1.1 Motivation

Scene depth recovery depends on scene characteristics, i.e., depth regions on close-view and tiny objects are inclined to be destroyed by downsampling degradation more seriously than the far-view object and background. Meanwhile, most existing DSR methods \cite{13, 40} have unbalanced estimation in one depth map with a same model. Hence, it may cause inaccurate depth recovery for different depth regions. Besides, depth map captured by a depth sensor are usually polluted with distance-dependent Gaussian noise, i.e, the intensity of the noise depends on the scene depth \cite{5}. However, previous studies mainly focus on a single model to process all regions of a depth map without considering the above complex degradation at region-level, which is suboptimal due to the statistical, computational, and representational limitations. Therefore, we expect to develop a new method to discriminately process each region within a depth map by its depth range in the scenario of DSR.

In addition, most existing algorithms treat DSR task of different scale factors (\times 2, \times 4, \times 8, \times 16) as independent problems, and require many scale-specific networks that need
to be trained independently to deal with various scales. However, in real world applications, the truly-wanted upscaling factors of the given scenes are fractional (not integer) or even unknown. Alternatively testing on the current LR input to find a suitable model among all the well-trained scale-specific ones is time-consuming and impractical, or even cannot obtain the desired results. Moreover, perceptual quality of the restored depth map is relatively subjective, and it is necessary for the model to finely-grained control the depth restoration according to image characteristics, which cannot be done using existing deterministic networks. Therefore, a controllable network with high generalization ability to different upscaling factors is needed.

1.2 Scope and Contributions
Based on the above analyses, this paper breaks away the shackles of general paradigms and introduces a distance-aware deep controllable slicing network from a novel perspective, as shown in Fig. 1. Specifically, we propose a slicing network architecture to learn a set of slicing branches to specify some depth range, and this novel network tends to yield accurate depth recovery in a divide-and-conquer manner. To make each branch more accurate at representing different depth, we propose a distance-aware weighting scheme to generate a set of weighting masks from the depth map features, which are applied on the estimated results from the slicing branches to make these branches focus on their specific depth regions, and adaptively aggregate all the slicing branches in the ensemble. Meanwhile, we design a scale-controllable module that extracts different depth features before the slicing network, which realizes to super-resolve LR images with different downscaling factors. The module consists of three branches, i.e., generalized branch (GB), specialized branch (SB), and fusion branch (FB). GB aims to extract the common features from the input, while SB takes the given scale parameter and its corresponding LR depth map as input, then the generated specialized features that contain different scale information are fused with the features of GB through FB in a multi-scale fashion, enabling a richer depth representation. The proposed module can finely-grained control the depth restoration according to different depth map degradation through the pre-defined scale hyper-parameter. Our main contributions are summarized as follows:

1) An end-to-end deep controllable slicing network to realize region-level depth recovery and high generalization ability for the task of DSR.

2) A scale-controllable module (SCM), which realizes the fine-grained control of depth restoration with arbitrary magnification in one united model.

3) A depth slicing module (DSM), which discriminatingly uses depth map features with different depth ranges to super-resolve the depth map in a divide-and-conquer manner.

Note that our model stands on single depth map SR without the aid of color information, but performs better than other color-guided DSR methods on both synthetic and real-world datasets. Besides, our model that is trained on all the scales together is also superior to other state-of-the-art scale-specific models that are trained independently on each scale.

2 RELATED WORK
2.1 CNN-based Depth Super-Resolution
Depending on the input data, DSR methods can be mainly divided into single depth map SR with only LR depth map as input and color-guided DSR with LR depth map and its corresponding HR color image as input.

Riegler et al. [32] integrated the piecewise affine structures into an CNN which combines the advantage of data driven methods and energy minimization models to recover the accurate HR depth map. Hui et al. [17] proposed a low-to-high resolution network to progressively extract features and raise the spatial resolution. Some existing DSR methods use color information as guidance to recover the degraded depth maps. Li et al. [25] employed a two-path CNN to obtain the HR depth map which is designed based on the concept of joint filters. Based on the model of single DSR, Hui et al. [17] also applied a gradual up-sampling method with a hierarchical color guidance module, which further exploits the dependency between color and depth structure to resolve ambiguity in DSR. Wen et al. [40] used the color information as guidance to infer an initial HR depth map, then proposed a coarse-to-fine networks to progressively optimize the depth map. Wang et al. [39] put forward a DSR network to learn a binary map of depth edges and then recovered the HR depth map based on edge-guided filter or cascaded network modules. However, due to treating all depth regions equally without considering depth range variation, there is still room for the above methods to make further improvement.

2.2 Ensemble Strategy
For image restoration tasks, the ensemble strategy has been explored to boost a network’s performance. As studied in [4], a single model or a model with only one branch is usually subject to computational and representational limitations. To alleviate this problem, Zhang et al. [42] proposed a lightweight ensemble method to improve the generalization of negative correlation learning for regression problems. Qin et al. [31] suggested a difficulty-aware image SR method that use a dual-way network to separately recover easy image regions and hard ones. Li et al. [23] applied a learning-based adapting method to ensemble the outputs from multiple models, which can exploit the information among successive video frames. Inspired by the above methods that capture different features, structures or sub-components in an image with separate models or branches, we make use of a slicing network to discriminatively process each region within a depth map by its depth range in the scenario of DSR.

2.3 Generalization Ability
Many techniques have been explored to improve the generalization ability of network for image restoration. Kim et al. [21] and Gao et al. [9] both proposed a joint-training strategy
Figure 1: Overview of the network architecture. SCM is to realize DSR with different downscaling factors in an unified model, which allows to finely-grained control the depth restoration results by using a scale parameter, while DSM aims to learn a set of slicing branches in a divide-and-conquer manner, parameterized by a distance-aware weighting scheme to adaptively aggregate all the branches in the ensemble.

Figure 3: Down- and up-projection units. ‘Deconv’ and ‘Conv’ represent de-convolution and convolution layers, which can be regarded as upsampling and downsampling operators to transform features between HR and LR spatial domains, respectively.

3 PROPOSED METHOD

Fig. 1 outlines the whole architecture and detailed configuration of our proposed deep controllable slicing network. Let \( D_{LR} \) and \( \gamma_m \) be the LR depth map (interpolated to the desired output size) and the upsampling factor as input, respectively. The goal is to predict the corresponding super-resolved depth map \( D_{SR} \). Note that color information can bring improvement for DSR, but may introduce texture-copying and depth bleeding artifacts because of the depth-color inconsistency. Therefore, our method focuses on single depth map SR, and achieves superior performance to other color-guided methods.

\( \gamma_m \) is additionally fed into SB to represent the desired scale. FB aggregates features from SB and GB in a multi-scale fashion, and then outputs the intermediate feature map \( F_i \). In DSM, \( F_i \) is spatially split into \( n \) regions according to different depth ranges, and is sent into the corresponding slicing branches. Then, the intermediate depth map \( D_s \) is obtained by weighted-averaging the outputs from slicing branches. Finally, \( D_s \) is refined by a simple residual structure to output the final super-resolved depth map \( D_{SR} \).

3.1 Scale-Controllable Module

To realize the fine-grained control of depth restoration with arbitrary upsampling factors, we propose a scale-controllable module (SCM) to learn a set of controllable features. As shown in Fig. 2, GB can be regarded as our backbone and contains \( M \) stacked generalized blocks, while SB includes \( M \) stacked specialized blocks together with \( M \) successive fully connected blocks. FB contains \( M \) fusion blocks to aggregate the output features from generalized blocks, specialized blocks and fully connected blocks at different stages in a multi-scale fashion.
where \( G \) vector has different fusion coefficients for the multi-channel at every feature extraction stage, which is defined as follows:

\[
m \rightarrow \text{fusion operation to selectively treat the output features}
\]

\[m \] is the fusion vector of \( m \).

Specifically, we introduce fully connected block to map an output features of generalized blocks and specialized blocks. We use three fully connected layers to implement this mapping operation:

\[
F \rightarrow \text{series of shallow feature extraction operations such as convolution. Then,} F_d \rightarrow \text{LR spatial domain and then mapping the reconstruction errors back into the HR domain. The specialized block is a simple residual architecture consisting of four convolution layers, each followed by a ReLU activation.}
\]

To obtain the generalized features and specialized features discriminately from GB and SB, the whole network is learned with a two-stage training strategy based on different optimization objectives. More training details are described in Sec. 3.3.

### 3.2 Depth Slicing Module

We formulate our depth slicing module (DSM) to discriminately mitigate complex degradation at region-level. DSM consists of a set of slicing branches and a distance-aware weighting CNN, in which each has a same network architecture with the specialized block, i.e., a simple residual block. Each slicing branch \( S_n \) that specifies some depth range tends to yield accurate depth estimation \( D_{S_n} \) from the shared depth feature \( F_c \) on the corresponding depth region:

\[
D_{S_n} = G^m_{slice}(F_c),
\]

where \( G^m_{slice}(\cdot) \) denotes the \( n \)-th slicing block. To achieve accurate regression, we also propose a distance-aware weighting scheme to learn the fusion weights for adaptively aggregating all the slicing branches in an ensemble. The output of the weighting CNN is a set of weighting masks \( \{W_n, n \in [1, N]\} \), which are defined as follows:

\[
[W_1, W_2, ..., W_{n-1}, W_n] = G_w(F_c),
\]

where \( G_w(\cdot) \) denotes the weighting CNN. The weighting mask \( W_n \) consists of 0 and 1, where 1 represents that this pixel belongs to corresponding depth ranges. Our proposed distance-aware weighting scheme is automatically aware of

**Figure 2: Network architecture of our SCM. GB aims to extract the common features from the input, while SB takes the given scale parameter and its corresponding LR depth map as input, then generates specialized features to adaptively tune the features of GB through FB in a linear fusion fashion, e.g., \( f^m = g^m + \gamma^m \cdot s^m \).**
multiple depth regions according to different depth ranges (distance within \( D_{S_n} \)), and assists each slicing branch to recover the depth values focusing on its corresponding depth region. Thus, the outputs of all the slicing branches are aggregated with the estimated weighting masks, and generate an accurate recovered depth map \( D_S \) on all the depth regions:

\[
D_S = \frac{\sum_i^n D_{S_i} \ast W_i}{\sum_i^n W_i},
\]

where \( \ast \) denotes the element-wise multiplication.

When fusing slicing branches, the intermediate depth map \( D_S \) may demonstrate some errors at the junction of different depth regions. Therefore, we add a refinement module \( \mathcal{G}_{refine}(\cdot) \) at the end of the DSM module to further enhance the intermediate depth map. The final super-resolved depth map \( D_{SR} \) is reconstructed from \( D_S \) by using a sample residual block with three 3×3 convolution layers interlaced by two ReLU activations.

Note that, by dividing the DSR task into multiple subproblems applied on each specific depth range, our method shares the essence of ensemble strategy and yields more robust estimations than a single DSR network.

### 3.3 Training Algorithm

**Training Data Generation.** Given that we have a collection of LR-HR paired depth maps, which contain HR depth maps \( D_{HR} \in D_{HR} \) and the corresponding LR depth maps \( D_{LR} \in D_{LR} \) (2×2, 4×4, 8×8, 16×16 cases downsampled from HR depth maps). We produce the ground truth depth regions \( D_{S_n}^g \in D_S \) by slicing the corresponding HR depth map \( D_{HR} \) into \( N \) regions with equal depth range according to the maximum distance. Meanwhile, we generate the ground truth weighting masks \( W^{gt}_i \in W \) by setting the non-zero values of the corresponding sliced depth map \( D_{S_n}^g \) as one to form a binary map.

**Loss Functions.** All the slicing branches are trained in a supervised manner such that each one becomes accurate and specialized on a specific depth range. The loss function for training the slicing branches can be formulated as follows:

\[
\mathcal{L}_S = \sum_i^n \| D_{S_i} - D_{S_i}^{gt} \|_1, \tag{8}
\]

The weighting CNN is also trained in a supervised manner, and its loss function is engaged to enforce the regions of concern with regard to a specific depth range, which is defined as

\[
\mathcal{L}_W = \sum_i^n \| W_i - W_i^{gt} \|_1, \tag{9}
\]

For the final depth map reconstruction, we directly measure the pixel-wise difference between the predicted depth map \( D_{SR} \) and its corresponding ground truth \( D_{HR} \) as a task loss to encourage an accurate reconstruction:

\[
\mathcal{L}_T = \| D_{SR} - D_{HR} \|_1, \tag{10}
\]

Combining the above three losses, then we have the overall loss for our model:

\[
\mathcal{L} = \mathcal{L}_T + \lambda_1 \mathcal{L}_S + \lambda_2 \mathcal{L}_W, \tag{11}
\]

where \( \lambda_1 \) and \( \lambda_2 \) are the trade-off parameters.

**Training Strategy.** We realize the fine-grained controllable feature learning by a two-stage training strategy. For the first stage, we fix all vectors \( \{\gamma^m, m \in [1, M]\} \) as all-one vectors and remove the fully connected block from SB. We train the rest modules together on the dataset with all the downsampling inputs. Note that, this stage aims to learn the generalization features from diverse degraded depth maps. For the second stage, we fix the parameters of GB and add the fully connected blocks back to SB. We set the scale factor \( \gamma_{in} \) as a designated value that dynamically matches the type of input depth map, i.e., \( \gamma_{in} \) can be set as 1, 2, 3, 4 corresponding to \( \times 2, \times 4, \times 8, \times 16 \) LR depth maps, respectively. The network learns the specialized features by updating the parameters of SB and DSM. At testing phase, the scale factor \( \gamma_{in} \) can be set at any values (from 1 to 4) to tune the restoration performance without knowing the real upsampling factors beforehand, which is very practical for the DSR task tested on real-world inputs.

### 4 EXPERIMENTAL RESULTS

To generate training data, we use 38 depth maps (6, 21, 9 depth maps from 2001, 2003, 2006, and 2014 datasets, respectively) from Middlebury dataset. To test the performance, we conducted experiments on Middlebury 2005 dataset (6 standard test depth maps, i.e., Art, Books, Moebius, Dolls, Laundry, Reindeer), and evaluate the generalization on MPI Sintel dataset (5 depth frames, i.e., Ambush_2-15, Ambush_4-12, Ambush_5-11, Tumble_3-23) and ToFMark dataset (7 (3 real-world depth maps, i.e., Books, Shark, Devil, captured by ToF depth sensors). Another training and testing dataset is NYU v2 RGB-D dataset [36] captured from Kinect. Following the common splitting method, we use the first 1000 images of the NYU dataset as training data, and evaluate on the last 449 images. We randomly extract 15000+ depth patches of a fixed size of 256×256 from HR depth maps and augment the training dataset by 180-degree-rotation. The corresponding LR depth patches are the squared size of 128, 64, 32, and 16 according to 2, 4, 8, and 16 scale factors respectively. The metric of Mean Absolute Difference (MAD) is used for objective evaluation. During training, we set the number of generalized/specialized blocks as \( M = 4 \) and the number of slicing branches as \( N = 5 \). We set the trade-off parameters as \( \lambda_1 = 0.2 \) and \( \lambda_2 = 0.2 \) after trials. For optimization, we used Adam optimizer with momentum = 0.9, \( \beta_1 = 0.9, \beta_2 = 0.99 \), and \( \epsilon = 10^{-8} \). The initial learning rate is set to 0.0001 and decreased by multiplying by 0.1 for every 50 epochs. We implemented our models by PyTorch framework with GPU acceleration.
Table 1: Quantitative depth upsampling results on Middlebury dataset. (lower MADs, better performance)

<table>
<thead>
<tr>
<th>Method</th>
<th>Art x2</th>
<th>Books x2</th>
<th>Dolls x2</th>
<th>Laundry x2</th>
<th>Moebius x2</th>
<th>Reindeer x2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x4</td>
<td>x8</td>
<td>x16</td>
<td>x4</td>
<td>x8</td>
<td>x16</td>
</tr>
<tr>
<td>Bicubic</td>
<td>0.11</td>
<td>0.25</td>
<td>0.61</td>
<td>1.80</td>
<td>1.33</td>
<td>0.89</td>
</tr>
<tr>
<td>FGI [26]</td>
<td>0.17</td>
<td>0.33</td>
<td>0.73</td>
<td>1.44</td>
<td>1.12</td>
<td>0.89</td>
</tr>
<tr>
<td>MSG [17]</td>
<td>-</td>
<td>0.25</td>
<td>0.51</td>
<td>1.12</td>
<td>0.89</td>
<td>0.73</td>
</tr>
<tr>
<td>DEIN [41]</td>
<td>0.15</td>
<td>0.33</td>
<td>0.73</td>
<td>1.12</td>
<td>0.89</td>
<td>0.73</td>
</tr>
<tr>
<td>CCFN [40]</td>
<td>-</td>
<td>0.25</td>
<td>0.51</td>
<td>1.12</td>
<td>0.89</td>
<td>0.73</td>
</tr>
<tr>
<td>GSPT [3]</td>
<td>0.12</td>
<td>0.25</td>
<td>0.73</td>
<td>1.12</td>
<td>0.89</td>
<td>0.73</td>
</tr>
<tr>
<td>DSR_N [39]</td>
<td>0.12</td>
<td>0.25</td>
<td>0.73</td>
<td>1.12</td>
<td>0.89</td>
<td>0.73</td>
</tr>
<tr>
<td>Ours</td>
<td>0.10</td>
<td>0.23</td>
<td>0.58</td>
<td>1.30</td>
<td>0.96</td>
<td>0.79</td>
</tr>
</tbody>
</table>

Figure 4: Visual comparison of ×8 upsampling results on Art, Dolls: (a) GT; (b) LR; (c) Bicubic, (d) FGI [26], (e) DJF [25], (f) DGDIE [12], (g) DEIN [41], (h) GSPT [3], (i) DSR_N [39], (j) Ours, (k) GT.

Table 2: Quantitative depth upsampling results on real NYU v2 dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>Art x4</th>
<th>Books x4</th>
<th>Dolls x4</th>
<th>Laundry x4</th>
<th>Moebius x4</th>
<th>Reindeer x4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x8</td>
<td>x8</td>
<td>x16</td>
<td>x8</td>
<td>x8</td>
<td>x16</td>
</tr>
<tr>
<td>Bicubic</td>
<td>6.29</td>
<td>6.29</td>
<td>7.24</td>
<td>7.24</td>
<td>6.29</td>
<td>7.24</td>
</tr>
<tr>
<td>MSG [17]</td>
<td>4.00</td>
<td>3.54</td>
<td>1.56</td>
<td>3.35</td>
<td>2.39</td>
<td>1.74</td>
</tr>
<tr>
<td>DGDIE [12]</td>
<td>1.94</td>
<td>3.34</td>
<td>1.29</td>
<td>2.04</td>
<td>1.39</td>
<td>2.05</td>
</tr>
<tr>
<td>DEIN [41]</td>
<td>-</td>
<td>0.33</td>
<td>0.73</td>
<td>0.91</td>
<td>0.99</td>
<td>0.87</td>
</tr>
<tr>
<td>CCFN [40]</td>
<td>-</td>
<td>0.33</td>
<td>0.73</td>
<td>0.91</td>
<td>0.87</td>
<td>0.87</td>
</tr>
<tr>
<td>GSPT [3]</td>
<td>0.12</td>
<td>0.25</td>
<td>0.73</td>
<td>1.12</td>
<td>0.89</td>
<td>0.73</td>
</tr>
<tr>
<td>DSR_N [39]</td>
<td>0.12</td>
<td>0.25</td>
<td>0.73</td>
<td>1.12</td>
<td>0.89</td>
<td>0.73</td>
</tr>
<tr>
<td>Ours</td>
<td>0.10</td>
<td>0.23</td>
<td>0.58</td>
<td>1.30</td>
<td>0.96</td>
<td>0.79</td>
</tr>
</tbody>
</table>

4.1 Performance Comparison

Depth SR under Noiseless Cases: To validate the superiority of our method, we first evaluate noiseless cases on Middlebury and NYU datasets, respectively.

For Middlebury dataset, we compare with DJF [25], MSG [17], DGDIE [12], DEIN [41], CCFN [40], GSPT [3], DSR_N [39], which are learning-based methods based on color.

Table 3: Quantitative depth upsampling results noisy Milddebury dataset.(lower MADs, better performance)

<table>
<thead>
<tr>
<th>Method</th>
<th>Art x4</th>
<th>Books x4</th>
<th>Dolls x4</th>
<th>Laundry x4</th>
<th>Moebius x4</th>
<th>Reindeer x4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x8</td>
<td>x8</td>
<td>x16</td>
<td>x8</td>
<td>x8</td>
<td>x16</td>
</tr>
<tr>
<td>TGV</td>
<td>2.76</td>
<td>4.67</td>
<td>2.54</td>
<td>1.75</td>
<td>3.71</td>
<td>1.89</td>
</tr>
<tr>
<td>MSG [17]</td>
<td>1.57</td>
<td>2.06</td>
<td>1.18</td>
<td>1.12</td>
<td>1.78</td>
<td>1.28</td>
</tr>
<tr>
<td>DGDIE [12]</td>
<td>1.68</td>
<td>3.34</td>
<td>1.29</td>
<td>2.04</td>
<td>1.39</td>
<td>2.05</td>
</tr>
<tr>
<td>DEIN [41]</td>
<td>2.44</td>
<td>4.24</td>
<td>1.44</td>
<td>2.38</td>
<td>1.55</td>
<td>2.45</td>
</tr>
<tr>
<td>GSPT [3]</td>
<td>1.33</td>
<td>2.67</td>
<td>1.37</td>
<td>1.26</td>
<td>2.03</td>
<td>1.24</td>
</tr>
<tr>
<td>DSR_N [39]</td>
<td>1.60</td>
<td>3.25</td>
<td>1.21</td>
<td>1.98</td>
<td>1.31</td>
<td>2.16</td>
</tr>
<tr>
<td>Ours</td>
<td>1.20</td>
<td>2.13</td>
<td>1.96</td>
<td>1.53</td>
<td>2.28</td>
<td>1.67</td>
</tr>
</tbody>
</table>
guidance. Table 1 presents the $\times 2$, $\times 4$, $\times 8$ and $\times 16$ upsampling performance of different methods. Our network almost obtains the best objective scores in all cases, especially for the $\times 8$ and $\times 16$ cases which are more difficult to restore. As shown in Fig. 4, obviously, our method can recover more pleasing structures and depth details, e.g., less jaggy artifacts around the stick in Art, more sharper and cleaner results on the toy’s head in Dolls. Note that our unified model performs better than these color-guided and scale-specific models that are trained independently on each scale.

Additionally, we evaluate on NYU dataset to demonstrate the effectiveness of our method. These state-of-the-art methods (JBU [22], DJF [25], DGDIE [12], GbFT [1], PAC [37], SVLRM [28], DKN [20]) that are also evaluated on NYU dataset are compared. As illustrated in Table 2, our method obtains the best objective results for all the upsampling cases. Fig. 5 further shows the visual performance under the $\times 8$ case. Focusing on the highlighted regions, we achieve the sharpest and clearest results.

Depth SR under Noisy Cases: Following [29], to simulate the acquisition process of a ToF depth sensor, we also add depth-dependent Gaussian noise to the training data, and then downsample the polluted depth maps at $\times 8$ and $\times 16$ scales. As shown in Table 3, our method achieves the best objective performance. We further provide perceptual comparisons in Fig. 6. The results of other methods present excessive cotton-like blurring, while our method can remove the noise and keep the sharpest depth boundaries on each depth range thanks to our distance-aware slicing network.

### 4.2 Evaluation on Generalization

We validate the generalization ability of our method on MPI Sintel dataset, the degraded depth inputs with unseen scale factors, and real ToFMark dataset.

**MPI Sintel Dataset** As demonstrated in Table 4, we almost achieve the best results on each case. Fig. 7 shows the visual comparison on Ambush $\times 12$. We obtain more accurate and clear depth details in the recovered results.

**ToFMark Dataset** Different from other methods that need to synthesize the suitable training datasets for testing on ToFMark or revise the inputs from ToFMark to adapt to...
Figure 9: Visual comparison of ×8 (γin = 3) upsampling results at different γin values.

Table 5: Generalization on real ToFMark dataset.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Books</td>
<td>12.26</td>
<td>12.78</td>
<td>12.31</td>
<td>13.21</td>
<td>11.15</td>
<td>11.03</td>
</tr>
<tr>
<td>Shark</td>
<td>14.11</td>
<td>15.11</td>
<td>14.06</td>
<td>15.03</td>
<td>13.26</td>
<td>12.08</td>
</tr>
</tbody>
</table>

Figure 11: Visualization of the weighting masks.

Table 6: Ablation Study on ×8 downsampling cases.

<table>
<thead>
<tr>
<th>Method</th>
<th>MAD Values (the lower the better )</th>
</tr>
</thead>
<tbody>
<tr>
<td>w/o γin</td>
<td>0.591 0.280 0.317 0.365 0.278 0.349</td>
</tr>
<tr>
<td>Ours</td>
<td>0.578 0.263 0.310 0.342 0.272 0.334</td>
</tr>
<tr>
<td>N=0</td>
<td>0.607 0.279 0.322 0.370 0.288 0.355</td>
</tr>
<tr>
<td>N=3</td>
<td>0.588 0.270 0.307 0.356 0.275 0.350</td>
</tr>
<tr>
<td>N=5 (Ours)</td>
<td>0.578 0.263 0.310 0.342 0.272 0.334</td>
</tr>
<tr>
<td>N=5 (w/o Weight)</td>
<td>0.619 0.286 0.327 0.403 0.299 0.363</td>
</tr>
<tr>
<td>N=7</td>
<td>0.563 0.247 0.302 0.344 0.239 0.322</td>
</tr>
</tbody>
</table>

the well-trained models, we directly send the depth maps into our model to acquire the results. As shown in Table 5 and Fig. 10, through controlling the scale parameter, we achieve the best performance, which demonstrates our better generalization ability on real data.

4.3 Ablation Study

Scale-Controllable Module (SCM). Previous experiments have already validated our SCM on the recovery of designated scales and the generalization of unseen datasets or scales. As shown in Table 6, we further demonstrate the effectiveness of SB according to different training stages (‘w/o γin’ for the first training stage, while ‘Ours’ for the complete training) through ablation study. Without the control of γin, the performance decreases a lot. Besides, we also provide perceptual comparisons in Fig. 9 to illustrate the fine-grained control of SCM. For the same ×8 LR input, we change γin from 0 to 5 with an interval of 0.5 to obtain different visual results. Our network presents a mild performance variation, and achieves the best performance at γin = 3.

Depth Slicing Module (DSM). We investigate our DSM via ablation study on two aspects, i.e., the distance-aware weighting scheme and the number of slicing branches. As listed in Table 6, removing the whole DSM module (N = 0) or the weighting scheme (N = 5 (w/o Weight)) will decrease the performance a lot compared to our final configuration (N = 5 (Ours)). Meanwhile, as the number of slicing branches N increases, the performance improves obviously, but saturates at N = 5 (our final choice). In addition, we visualize the probabilistic weighting masks to further validate the capability of our distance-aware weighting scheme as shown in Fig. 11. Given a test input, the proposed scheme can automatically distinguish different depth regions depending on the depth range, and generate the corresponding weighting mask on each specific depth range.

5 CONCLUSION

We propose a novel distance-aware deep controllable slicing network, which learns a set of slicing branches parameterized by a distance-aware weighting scheme to adaptively aggregate all the branches in an ensemble. In addition, a scale-controllable module is designed to realize the fine-grained control of depth restoration objectives. Comprehensive experiments demonstrate the superiority of our model.

REFERENCES


