Journal of Advances in Mathematics and Computer Science
35(4): 24-45, 2020; Article no.JAMCS.57935
ISSN: 2456-9968
(Past name: British Journal of Mathematics & Computer Science, Past ISSN: 2231-0851)

Fast and Effective Region-based Depth Map Upsampling with Application to Location Map-Free Reversible Data Hiding

Kuo-Liang Chung[1*], Yu-Ling Tseng[1], Tzu-Hsien Chan[1] and Ching-Sheng Wang[1]

[1] *Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, No. 43 Section 4, Keelung Road, Taipei, Taiwan 10672, R.O.C.*

Authors' contributions:

This work was carried out in collaboration among all authors. KLC is responsible for designing the proposed methods for depth map upsampling and location-map free reversible data hiding. YLT is responsible for implementing all the related depth map upsampling methods. THC is responsible for implementing all the related reversible data hiding methods. CSW is responsible for speeding up the proposed depth map upsampling method and its implementation.

Article Information

DOI: 10.9734/JAMCS/2020/v35i430268
Editor(s):
(1) Dr. Rodica Luca, Gheorghe Asachi Technical University, Romania.
Reviewers:
(1) L. M. I. Leo Joseph, S. R. University, India.
(2) Mansi Sharma, Indian Institute of Technology Madras, India.
(3) Poonam Yadav, CCS University, India.
Complete Peer review History: http://www.sdiarticle4.com/review-history/57935

Original Research Article

Received: 08 April 2020
Accepted: 14 June 2020
Published: 25 June 2020

Abstract

In this paper, we first propose a fast and effective region-based depth map upsampling method, and then propose a joint upsampling and location map-free reversible data hiding method, simpled called the JUR method. In the proposed upsampling method, all the missing depth pixels are partitioned into three disjoint regions: the homogeneous, semi-homogeneous, and non-homogeneous regions. Then, we propose the depth copying, mean value, and bicubic interpolation approaches to reconstruct the three kinds of missing depth pixels quickly, respectively. In the proposed JUR method, without any location map overhead, using the neighboring ground truth depth pixels of each missing depth pixel, achieving substantial quality, and embedding capacity merits. The comprehensive experiments have been carried out to not only justify the

**Corresponding author: E-mail: klchung01@gmail.com;*

execution-time and quality merits of the upsampled depth maps by our upsampling method relative to the state-of-the-art methods, but also justify the embedding capacity and quality merits of our JUR method when compared with the state-of-the-art methods.

Keywords: Bicubic interpolation; color plus depth video coding; depth map upsampling; depth no-synthesis-error; quality; reversible data hiding.

1 Introduction

The color plus depth video coding (CDVC) model [1] has been widely used in the 3D video consumer electronics market. To reduce the bitrate requirement in CDVC, one color image is bundled with a synchronized smaller depth map. For example, the color image and one-quarter sized depth map of Kinect 1 are of sizes 640×480 and 320×240, respectively. The smaller depth map is often viewed as a low-resolution depth map. Afterwards, the low-resolution depth map is upsampled, i.e. reconstructed, to the same size as the color image. For convenience, the color image is called the left color image. Nowadays, it is still a challenging problem to design a fast and effective depth map upsampling method. Based on the upsampled depth map, the warping process [2] is applied to map the left color image to a right warped virtual-view. One motivation of this paper is to design a fast and effective depth map upsampling method for solving this challenging problem. The other motivation of this paper is hightlighted in the next paragraph.

At the client side, after upsampling the depth map, because we dont need to consider the compression attack on the upsampled depth map. Therefore, the other motivation of this paper is to deploy the reversible data hiding (RDH) capability into the upsampled depth map. According to the definition of RDH [3], after embedding the hidden data into the upsampled depth map, not only the original upsampled depth map can be recovered completely, but the embedded hidden data also can be extracted correctly. Without needing the location map to indicate where the hidden data are saved in the marked depth map, it is a challenging problem to design an effective RDH method for upsampled depth maps such that the marked upsampled depth map has good quality and high embedding capacity. In this paper, we attempt to design a novel and effective joint depth map upsampling and location map-free RDH method, called the JUR method, for solving this challenging problem.

1.1 Related works

In this subsection, the related works for depth map upsampling and RDH for upsampled depth maps are introduced.

1.1.1 Related works for depth map upsampling

Based on the spatial filter kernel idea [4], Kopf *et al.* [5] presented a joint bilateral filtering method for upsampling depth maps. Later, Kim *et al.* [6] proposed the joint trilateral filtering method considering the color value difference. To improve the method in [7], Jung [8] proposed a block truncation coding-based method [9]. Yang *et al.* [10] proposed a sparse representation approach; however, their method is rather time-consuming. Ham *et al.* [11] proposed the majorization-minimization (MM) method by considering the structure relation between the guided color image and the depth map. Based on the Middlebury dataset [12], the MM method outperforms the guided-filtering (GF) [13] and the method in [14]. Based on the color image and the tentative estimated depth map as the guidance, Li *et al.* [15] proposed a hierarchical optimization framework by

progressively performing a cascaded and guided global interpolation (CGI). Based on the ToFMark test dataset, the CGI method outperforms the methods in [5], [13], [14]. Considering the correlation between the objects in the color image and the depth map via joint segmentation, Miguel *et al.* [16] proposed a robust upsampling and noise removal method for depth maps. Choi *et al.* [17] first segmented the depth map into regions of smooth surfaces, and then these regions are used to segment the color image to continuous regions and discontinuous regions. Corresponding to the discontinuous color regions, the depth upsampling is done by the depth-histogram-based method; otherwise, it is done by interpolating from the low-resolution depth map.

Xie *et al.* [18] proposed an edge-guided upsampling (EGU) method. Their EGU method first integrated the Markov random field technique and the training-based patch synthesis technique to generate a temporary upsampled depth map. Then, with the help of the modified version of the joint bilateral filtering, EGU can effectively remedy the jagged artifact and has better quality and edge-preserving effect. Based on the Middlebury dataset, EGU has shown better quality when compared with the related methods [7], [19], [20]. Because of involving the Markov random field and training-based patch synthesis techniques, EGU is time-consuming. Konno *et al.* [21] proposed a self-guided residual interpolation (SRI) method. The techniques used in SRI include the displacement field-guided filtering technique [22], the GF technique [13], and the bicubic interpolation (BIC). Based on the Middlebury dataset, SRI has better quality relative to the methods in [10], [13], [14], [22], [23]. Chang *et al.* [24] compared and analyzed the advantages and disadvantages among the five depth map upsampling methods, namely the bilinear, joint bilateral [5], noise-aware filter [25], guided image filter [13], discontinuity adaptive [26] depth upsampling methods. Furthermore, they applied depth refinement processes to achieve the quality improvement of the upsampled depth map.

Recently, under the deep learning supporting environment, Hui *et al.* [27] proposed a multi-scale guided convolutional end-to-end network (MSG-Net) for depth map upsampling. In MSG-Net, a convolutional neural networks-based (CNN-based) multi-scale fusion strategy was presented to complement the low-resolution depth features with the high-resolution luma features. Unlike some super-resolution networks that require pre-upsampling of input image by BIC in advance, MSG-net learns upsampling kernels inside the CNN. Based on the Middlebury dataset, the MSG-Net outperforms several traditional upsampling methods and Dong *et al.*'s super-resolution CNN (FSRCNN) method [28]. Kim *et al.* [29] learned sparse and spatially-variant kernels for determining the weighted averaging process to transfer structural details to low-resolution depth maps. Then, they propose a fast deformable kernel network to output sparse sets of neighbors and the corresponding weights adaptively for each pixel, leading to depth map upsampling. Based on the normalized convolution operation, Guo and Liu [30] proposed a guided convolutional layer to recover a dense depth map using a sparse and irregular depth map with a depth edge map as guidance. Further, they proposed a convolution network by combining different related methods, achieving better performance.

Among the eight comparative methods denoted by the set symbol "**UP**", BIC [31] and GF [13] are implemented in C++; the other six related upsampling methods, MM [11], EGU [18], SRI [21], FSRCNN [28], MSG-Net [27], and CGI [15], are available codes. Among these comparative methods, MM, EGU, SRI, MSG-Net, and CGI are state-of-the-art depth map upsampling methods.

1.1.2 Related works for reversible data hiding for depth maps

Chung *et al.* [32] presented a depth no-synthesis-error (D-NOSE) [33] based RDH method on upsampled depth maps directly, denoted by the D-RDH method, such that not only the original upsampled depth map can be recovered completely, but the embedded hidden data also can be extracted correctly. In detail, if the prediction error is equal to zero, D-RDH can embed the hidden data h with bit-length $\lfloor \log(upp(d) - p + 1) \rfloor$ into the current depth pixel. Here d denotes the

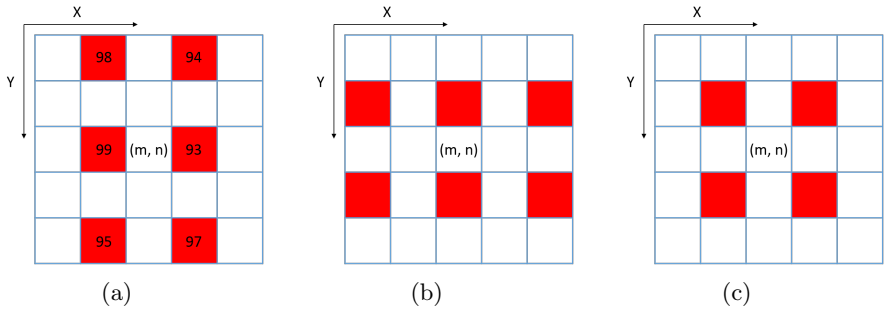


Fig. 1. Three possible 5×5 depth block types of each missing depth pixel $D^{HR}(m,n)$. (a) Type A when m is even and n is odd. (b) Type B when m is odd and n is even. (c) Type C when m is odd and n is odd

depth pixel-value and $upp(d)$ denotes the upper bound of d in the D-NOSE allowable interval $[low(d), upp(d)]$; the detailed definition of this interval is referred to the second paragraph of Subsection 2.2. When the prediction error is equal to -1, D-RDH can embed $\lfloor \log(p - low(d) + 1) \rfloor - 1$ bits; otherwise, D-RDH does nothing. As mentioned above, in the D-RDH method [32] some locations of the upsampled depth map can be embedded by hidden data, but the other locations are prohibited from embedding hidden data. Therefore, a location map is used to record all locations where the hidden data are embedded into the upsampled depth map directly, and then an arithmetic codec [34] is applied to compress the location map for reducing the extra memory requirement; it depletes the embedding capacity.

Recently, Shi *et al.* [35] presented a modified D-RDH (MD-RDH) method to embed much more hidden data for the case when the absolute prediction error is equal to zero. For this case, MD-RDH can embed the hidden data h with bit-length $\lfloor \log(upp(d)low(d) + 1) \rfloor$ into the depth pixel. Like D-RDH, MD-RDH also needs a location map to record all locations where the hidden data are embedded into the upsampled depth map directly, and it also compresses their location maps by the lossless arithmetic codec [34] to reduce the extra memory requirement.

Due to the D-NODE model used in D-RDH and MD-RDH, the test depth maps are collected from the Mobile3DTV dataset since this dataset provides the required parameters information needed by the D-NOSE model. Note that the Middlebury dataset does not provide the related parameters for the D-NOSE model. Because our proposed depth map upsampling method also takes the D-NODE model into account, the same test depth maps in Mobile3DTV are still used to compare the performance among the concerned methods.

1.2 Motivations

The common weakness existing in the above-mentioned related works for depth map upsampling is the lack of taking the region classification of missing depth pixels into account. Accordingly, it motivated us to propose a fast method to partition all the missing depth pixels into disjoint regions, and then design a special approach to reconstruct these missing depth pixels in the same region class quickly, also achieving good quality of the upsampled depth maps.

The common weakness existing in the related works on RDH for upsampled depth maps is the location map overhead and the prediction error constraint, depleting the embedding capacity and quality of the marked depth maps. Accordingly, it motivated us to propose an effective joint depth

map upsampling and RDH method, called the JUR method, achieving higher quality and embedding capacity of the marked depth maps.

1.3 Contributions

According to the above two motivations, in this paper, we first propose a fast and effective region-based depth map upsampling method, and then propose a joint upsampling and location map-free RDH method, called JUR. Initially, we move each true depth pixel at location (i, j) in the low-resolution depth map D^{LR} to the location ($2i$, $2j$) in the initial high-resolution depth map D^{HR}. Here, the true depth pixel at the location $(2i, 2j)$ is viewed as a ground truth depth pixel. Throughout this paper, "true depth pixel" and "ground truth depth pixel" denote the same thing. Therefore, 75% of the depth pixels in D^{HR} are the missing depth pixels to be reconstructed. The three contributions of this paper are clarified in the following aspects.

In the first contribution, a fast and effective region-based depth upsampling method is proposed. First, a novel approach is proposed to partition all the missing depth pixels into three regions, namely the homogeneous region which is geometrically flat, the semi-homogeneous region which is flat from the D-NOSE sense, and the non-homogeneous region which is geometrically non-flat. Then, in the proposed region-based depth map upsampling method, we apply the depth copying (DC) approach, the mean value (MV) approach, and the BIC technique to reconstruct the homogeneous, the semi-homogeneous, and the non-homogeneous missing depth pixels, respectively, achieving clear quality, execution-time, and perceptual effect improvement.

In the second contribution, instead of embedding hidden data into the upsampled depth map directly by the previous D-RDH [32] and MD-RDH [35] methods, for each missing depth pixel, according to its region class and its neighboring ground truth depth pixels, we propose a new joint upsampling and the location map-free RDH method, called the JUR method, achieving substantial quality, maximal embedding capacity, and embedding capacity merits relative to the sixteen comparative combinations in $\mathbf{UP} \times \{D-RDH, MD-RDH\}$.

In the third contribution, based on the Mobile3DTV dataset, the comprehensive experimental results demonstrated the good quality, fastest execution-time, good perceptual effect, and non-deep learning supporting environment merits of our depth map upsampling method relative to the eight comparative methods. Relative to the eight comparative depth upsampling methods, except the MSG-Net method, our method has clear PSNR and SSIM merits of the upsampled depth maps. In particular, our depth map upsampling method only needs 0.052, 0.069, and 0.074 seconds to obtain 2x, 4x, and 8x upsampled depth maps, respectively, which is much faster than the eight comparative methods. In addition, the experimental results also justified the quality-embedding capacity tradeoff merit of our JUR method relative to the sixteen comparative combinations in $\mathbf{UP} \times \{D-RDH, MD-RDH\}$. The maximal embedding capacity gain of our JUR method is at least 0.646 bpp (bit per pixel) when compared with the sixteen comparative combinations. When the embedding capacity is selected to be 0.1 bpp, 0.3 bpp, 0.5 bpp, and 0.7 bpp, as the base, the PSNR gains of marked upsampled depth maps by our JUR method are at least 5 dB, 5 dB, 4 dB, and 4 dB, respectively, relative to the sixteen comparative combinations.

The rest of this paper is organized as follows. In Section II, we propose a new and fast method to partition all missing depth pixels into three disjoint regions. In Section III, our fast and effective region-based map upsampling method is presented. In Section IV, our joint depth map upsampling and location map-free RDH method, JUR, is presented. In Section V, the thorough experiments are carried out to demonstrate the quality merit of our depth map upsampling method and the embedding capacity as well as the quality merits of our JUR method. In Section VI, some concluding remarks are addressed.

2 Partition All Missing Depth Pixels into Three Disjoint Regions

Before presenting the proposed region-based depth map upsampling method, which will be described in the next section, we first classify the missing depth pixels in the initial high-resolution depth map to three regions, such as the flat, i.e. homogeneous, region, semi-flat, i.e. semi- homogeneous, region, and non-flat, i.e. non-homogeneous, region. The motivation of this region classification is that for each region with special geometrical property, we can apply special approach to effectively reconstruct the missing depth pixels in that region, achieving low computational cost, image quality, and embedding capacity merits.

Following the above-mentioned motivation, we propose a fast linear-time, i.e. O(WH)-time, method to partition all missing depth pixels of the initial high-resolution depth map D^{HR} into three disjoint regions: the homogeneous region, the semi-homogeneous region, and the non-homogeneous region. Here, the size of D^{HR} is assumed to be W×H. The reader is suggested to refer to [36] for the definition of Big-O complexity.

2.1 Fast identify the Homogeneous missing depth Pixels

For each missing depth pixel $D^{HR}(m,n)$, we put a 5×5 window W centered at the location (m,n) to cover its neighboring true depth pixels in order to determine the region class of $D^{HR}(m,n)$.

2.1.1 Definition of homogeneous missing depth pixel

There are three possible block types, as shown in Fig. 1, where the true depth pixels covered by W are marked in red. Let k and l denote non-negative integers. The block type of $D^{HR}(m,n)$ is determined by

$$\text{Type} = \begin{cases} \text{type A when } m = 2k \text{ and } n = 2l+1 \\ \text{type B when } m = 2k+1 \text{ and } n = 2l \\ \text{type C when } m = 2k+1 \text{ and } n = 2l+1 \end{cases} \tag{2.1}$$

Definition 1. *For one missing depth pixel $D^{HR}(m,n)$, when all the true depth pixel values covered by the 5×5 window W centered at the location (m,n) are the same, $D^{HR}(m,n)$ is in a flat region and is identified as a homogeneous missing depth pixel.*

As shown in Fig. 1(c), for type C block, by Definition 1, $D^{HR}(m,n)$ is identified as a homogeneous missing depth pixel because of $D^{HR}(m1,n1) = D^{HR}(m1,n+1) = D^{HR}(m+1,n1) = D^{HR}(m+1,n+1)$. Considering Figs. 1(a)-(c), in the worst case, it takes 4.3 (= (5 + 5 + 3)/3) comparisons to check whether Condition 1 is held or not.

2.1.2 Speed up the checking of Condition 1

We propose an early jumping technique and a look-ahead pruning technique to speed up the checking of Condition 1.

2.1) Early jumping technique: Based on the column-major and right-to-left scanning order, by the early jumping technique, if the first two neighboring true depth pixels are not equal, Condition 1 is false for $D^{HR}(m,n)$ and we stop the subsequent checking step; otherwise, we continue with the next checking step. For the example in Fig. 1(a), because the number of possible checking steps is 1, 2, 3, 4, or 5, the average number of checking steps for block type A is 3 (= (1 + 2 + 3 + 4

+ 5)/5). By the same argument, the average number of checking steps for block type B is 3 (= (1 + 2 + 3 + 4 + 5)/5); the average number of checking steps for block type C is 2 (= (1 + 2 + 3)/3). Consequently, based on the early jumping technique, on average, it takes only 2.7 (= (3 + 3 + 2)/3) comparisons to determine whether Condition 1 is held for one missing depth pixel.

2.2) Look-ahead pruning technique: Furthermore, we adopt a look-ahead pruning technique to determine whether we should discard the checking of Condition 1 for the subsequent missing depth pixel(s). For the example in Fig. 1(a), if the first two neighboring true depth pixels, $D^{HR}(m-2, n+1)$ and $D^{HR}(m, n+1)$, are not equal, not only do we stop the remaining checking of Condition 1 for the missing depth pixel $D^{HR}(m, n)$, but we also discard the checking of condition 1 for the next missing depth pixel $D^{HR}(m, n+2)$. In the same way, for the example in Fig. 1(b), if the first two neighboring true depth pixels, $D^{HR}(m-1, n+2)$ and $D^{HR}(m+1, n+2)$, are not equal, we can discard the checking of Condition 1 for the next four missing depth pixels, $D^{HR}(m, n+1)$, $D^{HR}(m, n+2)$, $D^{HR}(m, n+3)$ and $D^{HR}(m, n+4)$. For the example in Fig. 1(c), if the first two neighboring true depth pixels, $D^{HR}(m-1, n+1)$ and $D^{HR}(m+1, n+1)$, are not equal, we can discard the checking of condition 1 for the next three missing depth pixels, $D^{HR}(m, n+1)$, $D^{HR}(m, n+2)$, and $D^{HR}(m, n+3)$. Besides the first two neighboring true depth pixels disussed above, the other two consecutive neighboring true depth pixels can be treated in the same way to speed up the checking of Condition 1.

Based on the Mobile3DTV dataset, the average ratio of the number of the homogeneous missing depth pixels over that of all missing depth pixels in D^{HR} is 73%. As for this case, we apply the proposed simple depth copying (DC) approach to construct each identified homogeneous missing depth pixel quickly, which will be formally described in Subsection III.A. Due to only considering the neighboring true depth pixels of $D^{HR}(m, n)$ to identify the pixel class of $D^{HR}(m, n)$, we thus adopt the smallest window, i.e. the 5×5 window. On the contrary, a larger window decreases the ratio of the homogeneous missing depth pixels, leading to no quality improvement.

Table 1. D-NOSE allowable intervals for balloons video sequence

Depth value d as index	0	1	...	98	99	100	101	...	254	255
$low(d)$	0	0	...	91	91	91	101	...	250	250
$upp(d)$	5	3	...	100	100	100	111	...	255	255
Mean of D-NOSE interval	3	3	...	96	96	96	106	...	253	253

2.2 Fast identify the Semi-homogeneous missing depth Pixels

Because "whether the missing depth pixel $D^{HR}(m, n)$ is semi-homogeneous or not" is dependent on the D-NOSE condition of the neighboring true depth pixels of $D^{HR}(m, n)$, we first introduce the preliminary of the D-NOSE model.

Let $Z_l(m, n)$ denote the raw depth value at the location (m, n) and let Z_{near} and Z_{far} denote the nearest and farthest raw depth values in the depth map, respectively. The quantized depth value of $Z_l(m, n)$ [37] is expressed by

$$D(m,n) = \lfloor 255 \times \frac{Z_{near}}{Z_l(m,n)} \times \frac{Z_{far} - Z_l(m,n)}{Z_{far} - Z_{near}} + 0.5 \rfloor \tag{2.2}$$

where the floor function $\lfloor w \rfloor$ denotes the largest integer smaller than or equal to w. The idea behind the D-NOSE model is that one depth value $D(m, n)$ can be perturbed to the other depth value $D'(m, n)$ such that this perturbation causes no warping error provided that $D'(m, n)$ is in the

D-NOSE allowable interval $[low(D(m,n)), upp(D(m,n))]$ where

$$\begin{aligned} low(D(m,n)) &= \left\lceil d^{-1}\left(\frac{\lceil (d(D(m,n)) - \lambda) \times N \rceil - 1}{N} + \lambda\right)\right\rceil \\ upp(D(m,n)) &= \left\lfloor d^{-1}\left(\frac{\lceil (d(D(m,n)) - \lambda) \times N \rceil}{N} + \lambda\right)\right\rfloor \end{aligned} \tag{2.3}$$

with λ-rounding and the precision $1/N$. In the term "$\frac{1}{N}\lceil (d(D(m,n)) - \lambda) \times N \rceil$", the ceiling function $\lceil w \rceil$ denotes the smallest integer greater than or equal to w, λ denotes the λ-rounding horizontal disparity of $d(D(m,n))$, and d^{-1} denotes the inverse function of d. Here, the horizontal disparity function $d(D(m,n))$ is defined by

$$d(D(m,n)) = \frac{f \times l_b}{Q^{-1}(D(m,n))} \tag{2.4}$$

where f denotes the focal length of the real camera and l_b denotes the baseline length between the real camera and the virtual camera. The inverse function $Q^{-1}(D(m,n))$ is defined by

$$\begin{aligned} Q^{-1}(D(m,n)) &= Z_l(m,n) \\ &= \frac{1}{\frac{D(m,n)}{255}\left(\frac{1}{Z_{near}} - \frac{1}{Z_{far}}\right) + \frac{1}{Z_{far}}} \end{aligned} \tag{2.5}$$

Let us take one real example to explain how the D-NOSE model works. The example is taken from the Balloons sequence with $Z_l(m,n) = 1149.02$, $\lambda = 0.5$, $N = 1$, $f = 2241.25607$, $l_b = 5$, $Z_{near} = 448.251214$, and $Z_{far} = 11206.280350$. By Eq. (2.2), the quantized depth pixel of $Z_l(m,n)$ is $D(m,n) = 93$. By Eq. (2.4), the calculated horizontal disparity value is $d(D(m,n)) = 9.75294$. Further, for $D(m,n) = 93$, by Eq. (2.3), the D-NOSE allowable interval is $[low(93), upp(93)] = [91, 100]$.

The D-NOSE model suggests that the value of $D(m,n)$ can be perturbed from 93 to any value $D'(m,n) \in [91, 100]$, causing no warping error. On the other hand, for $D(m,n)$ and $D'(m,n)$, the warping function maps the left color image pixel $I(m,n)$ to the same location (m, n_0) in the right virtual-view, where $(n_0 - n)$ denotes the horizontal disparity. After explaining the D-NOSE model, the semi-homogeneous missing depth in the flat region from the D-NOSE sense is defined below.

Definition 2. *For one missing depth pixel $D^{HR}(m,n)$, when all of its neighboring true depth pixel values covered by the 5×5 window W have the same D-NOSE allowable interval, $D^{HR}(m,n)$ is in a flat region from the D-NOSE sense and $D^{HR}(m,n)$ is identified as a semi-homogeneous missing depth pixel.*

In order to quickly identify whether one missing depth pixel is semi-homogeneous or not, Table 1 is built up in advance to record the 256 D-NOSE allowable intervals for depth values over the integer interval [0, 255], where "*d*" denotes the quantized depth value. For d = 93, by Table 1, its D-NOSE allowable interval, [91, 100], can be accessed in O(1) time.

To further reduce the identification time, one extra row is added in Table 1 to record the mean of each D-NOSE allowable interval. For example, suppose the depth block type of $D^{HR}(m,n)$ is type C and the four neighboring true depth pixel values are $D^{HR}(m-1, n-1) = 93$, $D^{HR}(m-1, n+1) = 98$, $D^{HR}(m+1, n-1) = 96$, and $D^{HR}(m+1, n+1) = 100$. From Table 1, all the mean values of the four corresponding D-NOSE allowable intervals are 96, implying that $D^{HR}(m,n)$ is a semi-homogeneous depth pixel. Based on the test depth maps, the average ratio of the number of the semi-homogeneous missing depth pixels over that of all missing depth pixels in D^{HR} is 12%.

Using the early jumping technique and the look-ahead pruning technique described in Subsection II.A, we also can speed up the checking of Definition 2. In detail, based on the early jumping technique, for one missing depth pixel, on average, it takes only 2.7 (= (3 + 3 + 2)/3) D-NODE value comparisons to check whether Condition 2 is held or not. Using the look-ahead pruning technique, we may discard the checking of condition 2 for the subsequent missing depth pixel(s).

2.3 Identify the Non-homogeneous missing depth Pixels

Let S_m, S_h, and S_n denote the set of all missing, all homogeneous missing, all semi-homogeneous missing, and all non-homogeneous missing depth pixels in D^{HR}, respectively. We thus have $S_n = S_m \backslash S_h \backslash S_s$ where the operator "\" denotes difference operation in set theory. The non-homogeneous missing depth pixel is defined below.

Definition 3. *For one missing depth pixel $D^{HR}(m,n)$, when Conditions 1-2 are violated, it is identified as a non-homogeneous missing depth pixel.*

On the other hand, for one missing depth pixel, if Conditions 1-2 are false, it should be a non-homogeneous missing depth pixel. Based on the test depth maps, the average ratio of $|S_n|$ over $|S_m|$ is 15%.

3 The Proposed Region-Based Depth Map Upsampling Method

We first depict the sketch of our region-based depth map upsampling method with one example. Fig. 2(a) illustrates one given low-resolution depth map example. After constructing all homogeneous missing depth pixels by our depth copying (DC) approach, the reconstructed depth pixels are depicted in the flat regions of Fig. 2(b). Further, after reconstructing all semi-homogeneous depth pixels by our mean value (MV) approach, Fig. 2(c) illustrates the reconstructed depth pixels appearing in the semi-flat regions. Using BIC, the reconstructed depth pixels appear in the non-flat regions of Fig. 2(d). Fig. 2(e) illustrates the final upsampled depth map of Fig. 2(a) by our DC+MV+BIC method.

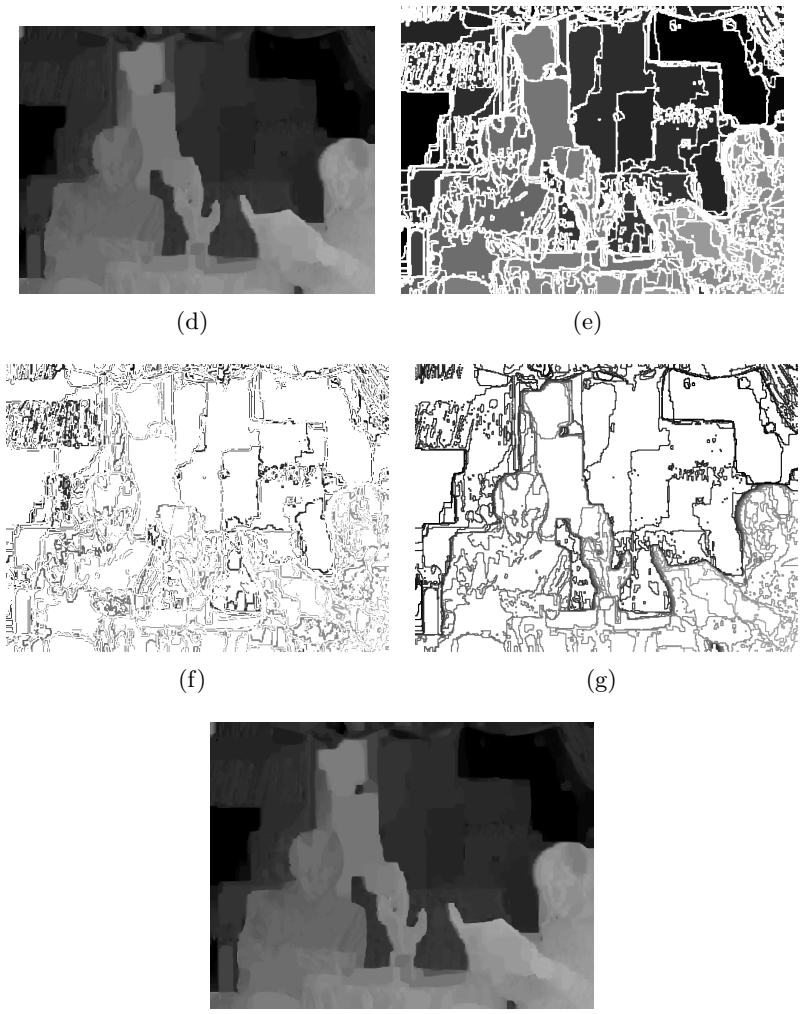
In the next three subsections, our region-based depth map upsampling method is described in detail.

3.1 Constructing homogeneous missing depth Pixels by the depth copying (DC) approach

For one homogeneous missing depth pixel $D^{HR}(m,n)$ in S_h, by our DC approach, it is constructed by

$$D^{HR}(m,n) = \begin{cases} D^{HR}(m,n-1) & \text{for type A block} \\ D^{HR}(m-1,n) & \text{for type B block} \\ D^{HR}(m-1,n-1) & \text{for type C block} \end{cases} \tag{3.1}$$

It takes only one assigment operation in Eq. (3.1). Because the average ratio of $|S_h|$ over $|S_m|$ is 73%, the proposed simple DC approach achieves low computational cost and high reconstruction accuracy merits.



(d) (e)

(f) (g)

(h)

Fig. 2. The sketch of our region-based depth map upsampling method. (a) The given low-resolution depth map. (b) The reconstructed homogeneous depth pixels, which occupy 73% of all reconstructed pixels, by our DC copying approach. (c) The reconstructed semi-homogeneous depth pixels, which occupy 12% of all reconstructed pixels, by our MV approach. (d) The reconstructed non-homogeneous depth pixels, which occupy 15% of all reconstructed pixels, by using BIC. (e) The constructed high-resolution depth map by our "DC+MV+BIC" method

3.2 Constructing semi-homogeneous missing depth Pixels by the mean value (MV) approach

For one semi-homogeneous missing depth pixel $D^{HR}(m,n)$ in S_s, by our MV approach, it is constructed by

$$D^{HR}(m,n) = \begin{cases} \sum_{k\in\{-2,0,2\},l\in\{-1,1\}} D^{HR}(m+k,n+l)/6 & \text{for type A block} \\ \sum_{k\in\{-1,1\},l\in\{-2,0,2\}} D^{HR}(m+k,n+l)/6 & \text{for type B block} \\ \sum_{k\in\{-1,1\},l\in\{-1,1\}} D^{HR}(m+k,n+l)/4 & \text{for type C block} \end{cases} \tag{3.2}$$

On average, it only takes 5.3 (= (6 + 6 + 4) / 3) arithmetic operations by Eq. (3.2) to reconstruct the value of $D^{HR}(m,n)$. By our MV approach, the reconstructed depth value of $D^{HR}(m,n)$ in S_s has the same D-NOSE allowable interval as that of any neighboring true depth pixel covered by the 5×5 window, causing no warping error.

3.3 Constructing non-homogeneous missing depth Pixels by a Bicubic interpolation (BIC) related approach

In this subsection, we apply a bicubic interpolation (BIC) to reconstruct the non-homogeneous missing depth pixels. In addition, we also investigate the BIC- and luma guided-based (BLG-based) fusion approach to reconstruct the non-homogeneous missing depth pixels.

When using BIC to construct the non-homogeneous depth pixel $D^{HR}(m,n)$, we perform the 9×9 BIC on the true depth pixels covered by a 9×9 window W centered at the location (m,n) to construct the value of $D^{HR}(m,n)$, denoted by $D^{HR}_{BIC}(m,n)$.

We now investigate an alternate BIC- and LG-based (BLG-based) fusion approach to reconstruct the non-homogeneous missing depth pixels of $D^{HR}(m,n)$. Suppose the block type of $D^{HR}(m,n)$ is type C, as shown in Fig. 1(c). The four neighboring true depth pixel values, $D^{HR}(m-1,n-1)$, $D^{HR}(m-1,n+1)$, $D^{HR}(m+1,n-1)$, and $D^{HR}(m+1,n+1)$, and the four corresponding co-located luma pixel values, $L(m-1,n-1)$, $L(m-1,n+1)$, $L(m+1,n-1)$, and $L(m+1,n+1)$, constitute the following four linear relations:

$$\begin{bmatrix} L(m-1,n-1) & 1 \\ L(m-1,n+1) & 1 \\ L(m+1,n-1) & 1 \\ L(m+1,n+1) & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} D^{HR}(m-1,n-1) \\ D^{HR}(m-1,n+1) \\ D^{HR}(m+1,n-1) \\ D^{HR}(m+1,n+1) \end{bmatrix} \tag{3.3}$$

where in Eq. (3.3), we have only two parameters, a and b, to be solved, but there are four equations. Eq. (3.3) is also called an over-determined system.

To solve the over-determined system, we apply the linear least squares fitting technique to find the best fitting straight line in terms of the slope "a" and the intercept "b" through the four known points, $(D^{HR}(m-1,n-1), L(m-1,n-1))$, $(D^{HR}(m-1,n+1), L(m-1,n+1))$, $(D^{HR}(m+1,n-1), L(m+1,n-1))$, and $(D^{HR}(m+1,n+1), L(m+1,n+1))$. By the widely used normal equation

formula [38], the parameter-pair (a, b) is solved by

$$\begin{bmatrix} a \\ b \end{bmatrix} = \left(\begin{bmatrix} L(m-1,n-1) & 1 \\ L(m-1,n+1) & 1 \\ L(m+1,n-1) & 1 \\ L(m+1,n+1) & 1 \end{bmatrix}^T \begin{bmatrix} L(m-1,n-1) & 1 \\ L(m-1,n+1) & 1 \\ L(m+1,n-1) & 1 \\ L(m+1,n+1) & 1 \end{bmatrix} \right)^{-1} \begin{bmatrix} L(m-1,n-1) & 1 \\ L(m-1,n+1) & 1 \\ L(m+1,n-1) & 1 \\ L(m+1,n+1) & 1 \end{bmatrix}^T \begin{bmatrix} D^{HR}(m-1,n-1) \\ D^{HR}(m-1,n+1) \\ D^{HR}(m+1,n-1) \\ D^{HR}(m+1,n+1) \end{bmatrix} \tag{3.4}$$

By Eq. (3.4), the estimated value of $D^{HR}(m,n)$ is given by

$$D^{HR}_{LG}(m,n) = a \times L(m,n) + b \tag{3.5}$$

Further, we fuse the two estimated values, $D^{HR}_{BIC}(m,n)$ and $D^{HR}_{LG}(m,n)$ by computing

$$\begin{aligned} D^{HR}_{BLG}(m,n) = {} & W_1(m,n) \times D^{HR}_{BIC}(m,n) \\ & + W_2(m,n) \times D^{HR}_{LG}(m,n) \end{aligned} \tag{3.6}$$

where the two weights $W_1(m,n)$ and $W_2(m,n)$ can be determined by the error-prediction based technique, as described in what follows. Before constructing the error maps of $D^{HR}_{BIC}(m,n)$ and D^{HR}_{LG}, we first perform the 4×4 BIC on $D^{HR}_{BIC}(m,n)$ and D^{HR}_{LG} to obtain the two downsampled results $D^{LR}_{BIC}(m,n)$ and D^{LR}_{LG}, respectively. Then, the two error maps are calculated by

$$\begin{aligned} err(D^{LR}_{BIC}) &= D^{LR} - D^{LR}_{BIC} \\ err(D^{LR}_{LG}) &= D^{LR} - D^{LR}_{LG} \end{aligned} \tag{3.7}$$

Performing the 9×9 BIC on $err(D^{LR}_{BIC})$ and $err(D^{LR}_{LG})$, the error maps of D^{HR}_{BIC} and D^{HR}_{LG}, $err(D^{HR}_{BIC})$ and $err(D^{HR}_{LG})$ are obtained. The two weights $W_1(m,n)$ and $W_2(m,n)$ in Eq. (3.6) are calculated by

$$\begin{aligned} W_1(m,n) &= \frac{(e^{-err(D^{HR}_{BIC}(m,n))})^{-1}}{(e^{-err(D^{HR}_{BIC}(m,n))})^{-1} + (e^{-err(D^{HR}_{LG}(m,n))})^{-1}} \\ W_2(m,n) &= \frac{(e^{-err(D^{HR}_{LG}(m,n))})^{-1}}{(e^{-err(D^{HR}_{BIC}(m,n))})^{-1} + (e^{-err(D^{HR}_{LG}(m,n))})^{-1}} \end{aligned} \tag{3.8}$$

Then, by Eq. (3.6), the fused result of $D^{HR}(m,n)$ can be obtained. According to our experimental results, for upsampling one depth map, on average, the PSNR gain of the DC+MV+BLG variant over our DC+MV+BIC method is about 0.02 dB, but the execution-time cost of DC+MV+BLG is more than two times of our DC+MV+BIC method. Therefore, in the experiment section, we only compare our DC+MV+BIC method with the other comparative methods.

4 The Proposed Joint Upsampling and Location Map-free Reversible Data Hiding Method: JUR

From Definitions 1-3 for missing depth pixels, we observe that for each missing depth pixel, its neighboring ground truth depth pixels, i.e. those true depth pixels around the missing depth pixel, provides an opportunity to develop a joint depth map upsampling and location map-free RDH method, called the JUR method. Note that the locations of these ground truth depth pixels are prohibited from embedding hidden data; all the other missing depth pixels can be upsampled and

embedded by hidden data simultaneously. As mentioned in the first paragraph of Subsection 1.3, each ground truth depth pixel is located at the position $(2i, 2j)$ and each marked depth pixel is located at the other position, so we don't need a location map to record where the hidden data are embedded into the upsampled depth map. Besides inheriting the quality merit of our region-based depth map upsampling method, in particular, JUR does not need any location map, achieving higher embedding capacity of the marked depth maps.

4.1 Upsampling and embedding process

Following the definition of the D-NOSE allowable interval $[low(d), upp(d)]$ in Eq. (2.3), to guarantee no warping error in the right warped virtual-view, if we want to embed the hidden data h into the upsampled depth pixel $D^{HR}(m,n)$, $D^{HR}(m,n)$ can be perturbed by

$$D^{HR}(m,n) = low(D^{HR}(m,n)) + h \tag{4.1}$$

subject to the constraint: $low(D^{HR}(m,n)) + h \leqslant upp(D^{HR}(m,n))$, leading to no warping error. On the contrary, if the constraint is violated, it leads to $low(D^{HR}(m,n)) + h \geqslant upp(D^{HR}(m,n))$, producing the warping error. Accordingly, the maximal bit-string length of h is equal to

$$L(h) = \lfloor \log(upp(D^{HR}(m,n)) - low(D^{HR}(m,n)) + 1) \rfloor \text{ bits} \tag{4.2}$$

By Eq. (4.2), the hidden data h is thus taken from the first $L(h)$ bits from the current hidden string $\mathbf{H}$, achieving the maximal data embedding for each reconstructed depth pixel $D^{HR}(m,n)$.

The proposed upsampling and embedding process in JUR is realized by the procedure ***"Procedure: Upsampling and Embedding Process"***.

For example, in Fig. 1(a), by Step 1, the current missing depth pixel $D^{HR}(m,n)$ is identified to be in S_s. After performing our MV approach on $D^{HR}(m,n)$, the upsampled depth pixel value of $D^{HR}(m,n)$ is 96 by Eq. (3.2). By Steps 2-3, the bit-length of h is equal to 3 ($= L(96) = \lfloor \log(upp(96) - low(96) + 1) \rfloor = \lfloor \log(100 - 91 + 1) \rfloor$). We thus take the first three bits from $\mathbf{H} = (01100011)_2$, and the hidden data to be embedded is equal to $h = (011)_2$. Consequently, the value of the marked depth pixel $D^{HR}(m,n)$ is set to 94 ($= low(96) + h = 91 + 3$). In addition, the hidden binary string becomes $\mathbf{H} = (00011)_2$.

Procedure: Upsampling and Embedding Process

Input: Current missing depth pixel $D^{HR}(m,n)$ and the current hidden binary string $\mathbf{H}$ to be embedded.
Output: Marked depth pixel $D^{HR}(m,n)$.
Step 1: For $D^{HR}(m,n)$, we perform our depth map upsampling method to obtain its upsampled depth pixel, still denoted by $D^{HR}(m,n)$.
Step 2: By Table 1, the D-NOSE allowable interval of $D^{HR}(m,n)$, $[low(D^{HR}(m,n)), upp(D^{HR}(m,n))]$, is accessed.
Step 3: By Eq. (4.1), $D^{HR}(m,n)$ is perturbed to $D^{HR}(m,n) = low(D^{HR}(m,n)) + h$ in which the hidden data h with length $L(h)$ (see Eq. (4.2)) is taken from the first $L(h)$ bits of $\mathbf{H}$.
Step 4: We report $D^{HR}(m,n)$ as the marked depth pixel. In addition, we cutoff the first $L(h)$ bits from $\mathbf{H}$.

4.2 Extracting and recovering process

Basically, the extracting and recovering process in JUR is the reverse of the ***Procedure: upsampling and embedding process***. Although the upsampled depth pixels are modified due to data embedding, the neighboring ground truth depth pixels of each marked depth pixel are always not modified and retain their original values. Therefore, in the proposed extracting and recovering process, for each marked depth pixel at location (m, n), based on its ground truth depth pixels covered by a 5×5 window centered at location (m, n), we repeat the depth upsampling process, as described in Section 3, to obtain the constructed value of $D^{HR}(m, n)$.

Furthermore, based on Table 1, we take the constructed depth value $D^{HR}(m, n)$ as the key to query the values of $low(D^{HR}(m, n))$ and $upper(D^{HR}(m, n))$. By Eq. (4.2), the maximal bit-string length of the hidden data h can be calculated, and then the hidden data h with bit-length $L(h)$ can be correctly extracted by subtracting the value $low(D^{HR}(m, n))$ from the marked depth pixel value $D^{HR}(m, n)$. Note that the upsampled depth pixel at location (m, n) can be recovered correctly by performing the depth map upsampling on its neighboring ground truth depth pixels again, and this is the reason why our JUR method can recover the original upsampled depth map, achieving the RDH goal. The proposed extracting and recovering process for each marked depth pixel $D^{HR}(m, n)$ is realized by the procedure ***"Procedure: Extracting and Recovering Process"***.

Procedure: Extracting and Recovering Process

Input: Marked depth pixel $D^{HR}(m, n)$.
Output: Recovered upsampled depth pixel $D^{HR}(m, n)$ and the extracted hidden data h.
Step 1: According to the neighboring ground truth depth pixels of the marked depth pixel $D^{HR}(m, n)$, by Definitions 1-3, we can identify the region class of the missing depth pixel at the location (m, n) to be in S_h, S_s, or S_n. Then, based on the neighboring ground truth depth pixels and the identified region class at the location (m, n), we apply the corresponding depth pixel upsampling approach, DC, MV, or BIC, to reconstruct the depth value at the location (m, n), outputting as the recovered upsampled depth pixel $D^{HR}(m, n)$.
Step 2: Using the value of the recovered upsampled depth pixel $D^{HR}(m, n)$ as a key, by Table 1, The value of $low(D^{HR}(m, n))$ can be accessed. By Eqs. (4.1)-(4.2), the extracted hidden data h with bit-length $L(h)$ is reported by subtracting the value $low(D^{HR}(m, n))$ from the input marked depth pixel value $D^{HR}(m, n)$.

Returning to the data hiding example introduced in the last paragraph of Subsection 4.1, we konw that the value of the marked depth pixel $D^{HR}(m, n)$ is 94. We now take it as the input to explain why the proposed extracting and recovering procedure can extract the hidden data h correctly and recover the originally upsampled depth pixel completely. After performing Step 1 of the above procedure extracting and recovering procedure on the location (m, n), the recovered upsampled depth pixel value at location (m, n) is 96. In Step 2, by Table 1, we have $low(96) = 91$; by Eq. (4.2), the value of the bit-length $L(96)$ is equal to 3. From Eq. (4.1), we know that the hiding data h is equal to 3 (=94-91). Furthermore, the value of $L(96)$ (= 3) indicates that the hidden data h is of 3-bit length. Because the value of h is 3 and the bit-length of h is 3, the hidden data h can be correctly extracted as $(011)_2$. Consequently, the hidden data h and the originally upsampled depth pixel can be correctly extracted and recovered, respectively.

Table 2. PSNR, SSIM and execution-time comparison among the concerned depth map upsampling methods

	PSNR			SSIM			Execution-time (seconds)		
	2x	4x	8x	2x	4x	8x	2x	4x	8x
BIC [16]	46.36	40.73	36.55	0.9922	0.9791	0.9662	0.21	0.27	0.28
GF [13]	45.03	41.29	37.87	0.9873	0.9763	0.9662	0.097	0.119	0.123
MM [12]	45.8	40.65	36.43	0.9814	0.9679	0.9527	97.96	95.62	99.43
EGU [30]	47.25	42.84	38.1	0.9928	0.9837	0.964	1001.43	2453.69	4036.33
SRI [18]	47.51	41.33	37.06	0.9935	0.9811	0.9687	8.33	10.44	10.91
FSRCNN [5] (GPU)	37.13	34.45	-	0.9425	0.9342	-	71.9	19.07	-
MSG-Net [14] (GPU)	55.35	49.76	45.03	0.9979	0.9932	0.9848	0.37	0.54	0.62
CGI [20]	42.63	38.99	36.28	0.9796	0.9674	0.9562	0.81	0.99	1.05
DC+MV+BI	48.66	43.86	40.2	0.9943	0.9859	0.9755	0.052	0.069	0.074

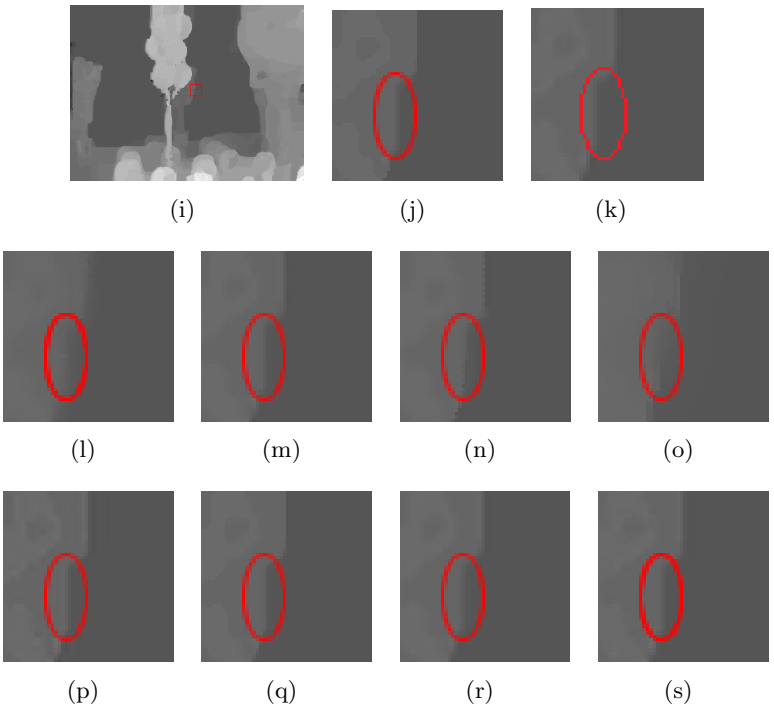


Fig. 3. The visual effect comparison for the upsampled depth map "Balloons". (a) The ground-truth high-resolution depth map. (b) The magnified depth sub-map cutoff from Fig. 3(a). (c) BIC [31]. (d) GF [13]. (e) MM [11]. (f) EGU [18]. (g) SRI [21]. (h) FSRCNN [28]. (i) MSG-Net [27]. (j) CGI [15]. (k) Our method

5 Experimental Results

Following the same performance evaluation way used in the previous depth upsampling methods, the first set of experiments is used to compare the PSNR, SSIM, and the visual effect of the upsampled depth maps by the concerned methods. In addition, the actual time cost for each concerned depth map upsampling method is reported, indicating the lowest computational cost merit of our DC+MV+BIC method. The second set of experiments is used to justify the quality

and embedding capacity merits of the marked depth maps by our JUR method. For fairness, we also apply the comparative RDH methods, D-RDH and MD-RDH, to the upsampled depth maps generated by some concerned depth map upsampling methods.

The execution codes of our DC+MV+BIC and JUR methods can be accessed from the website in [39]. BIC [31] and GF [13] are implemented in Visual C++ 2017. The available codes for the six concerned methods, MM [11], EGU [18], SRI [21], FSRCNN [28], MSG-Net [27], and CGI [15], are implemented in Matlab R2013a. Under the deep learning supporting environment, MSG-Net and FSRCNN are implemented on the platform: GeForce GTX 1080 Ti GPU with 24 GB of RAM based on the Caffe framework. All the related experiments have been performed under the Windows 10 operating system.

5.1 Performance comparison among the concerned depth map upsampling methods

In this subsection, the three upsampling ratios, "2×", "4×", and "8×", are considered in the experiments. Here, "2×", "4×", and "8×" mean that the size of the upsampled depth map is four times, 16 times, and 64 times, respectively, as large as the low-resolution depth map. Note that there are only two upsampling ratios "2×" and "4×" considered in the available code of FSRCNN. Differing from the GF and our DC+MV+BIC methods, the other concerned upsampling methods perform the upsampling ratio "4×" and "8×" directly instead of completing the upsampling ratio "2×" and "4×", respectively, in advance. In our experiment, the dataset Mobile3DTV with 6 videos are used and for each video, we take the first ten frames in the test.

5.1.1 PSNR and SSIM comparison

To evaluate the quality of the upsampled depth maps by the concerned methods, the PSNR and SSIM metrics are used. SSIM is measured by the joint effects of the luminance, contrast, and structure similarity preserving effect between the high-resolution ground-truth depth map and the upsampled depth map. We suggest that the readers refer to [40] for the detailed definition of SSIM. From Table 2, we observe that in terms of PSNR and SSIM, MSG-Net is in first place in red; our DC+MV+BIC method is in second place in green. In terms of PSNR, EGU is in third place in blue and SRI is in fourth place.

5.1.2 Visual effect comparison

Given the ground-truth high-resolution depth map shown in Fig. 3(a), the magnified depth submap cut off from the ground-truth upsampled depth map in Fig. 3(a) is shown in Fig. 3(b). After performing BIC, GF, MM, EGU, SRI, FSRCNN, MSG-Net, CGI, and our DC+MV+BIC method on the downsampled depth map of Fig. 3(b), Figs. 3(c)-(k) demonstrate the eight upsampled depth maps. As shown in the regions marked by red ellipses, MSG-Net and our method have the best visual effect relative to Fig. 3(b). However, due to the training effect, the boundary part in Fig. 3(i) by MSG-Net is too sharp relative to Fig. 3(b). For detailed visual comparison by the concerned methods, we suggest that readers refer to the website [41].

5.1.3 Execution time comparison

Because the two concerned CNN-based methods, FSRCNN and MSG, are realized by GPU in a parallel way and the other seven concerned methods are realized by CPU in a sequential way, it is unfair to represent their complexity in terms of big-O notation [36]. Accordingly, in terms of the execution-time in seconds per map required by each concerned method, on average, our DC+MV+BIC method only takes 0.0651 (= (0.0524 + 0.069 + 0.074)/3) seconds and is in first

place; GF takes 0.113 seconds and is in second place. BIC takes 0.26 seconds and is in third place; MSG-Net takes 0.51 seconds and is in fourth place; CGI and SRI are in fifth place and sixth place, respectively.

5.2 Embedding capacity and quality merits of our JUR method

To compare the performance of the concerned RDH methods for the upsampled depth maps, first, the maximal embedding capacity (ME-capacity), which is expressed as the maximal number of the average hidden bits saved in one depth pixel, is reported. For fairness, we perform each comparative RDH method on the upsampled depth maps generated by all the concerned upsampling methods. Note that the location maps used in D-RDH and MD-RDH have been compressed by the arithmetic codec.

Table 3. Maximal embedding capacity comparison among the concerned rdh combinations

Combination		ME-capacity (bpp)
BIC	D-RDH	0.798
	MD-RDH	0.988
GF	D-RDH	0.698
	MD-RDH	0.898
MM	D-RDH	0.696
	MD-RDH	0.864
EGU	D-RDH	0.731
	MD-RDH	0.933
SRI	D-RDH	0.718
	MD-RDH	0.893
FSRCNN	D-RDH	0.581
	MD-RDH	0.771
MSG-Net	D-RDH	0.717
	MD-RDH	0.866
CGI	D-RDH	0.730
	MD-RDH	0.900
BC+MV+BI	D-RDH	0.731
	MD-RDH	0.882
JUR		**1.634**

Table 4. Maximal embedding capacity improvement and psnr decay for "JUR + Histogram shifting"

	JUR	JUR + Histogram Shifting
ME-capacity (bpp)	1.63	1.88
PSNR	45.44	43.78

5.2.1 Maximal embedding capacity comparison

Based on the Mobile3DTV dataset, Table 3 tabulates the ME-capacity comparison of all the concerned RDH combinations. Because of the location map-free merit in JUR, our JUR method has the highest ME-capacity in boldface. MD-RDH is in second place and outperforms D-RDH. In addition, based on the same test depth maps, we apply the histogram shifting-based RDH technique

[3] on the marked depth maps generated by JUR to further enhance the ME-capacity of our JUR method. For convenience, this enhancement way is called the "JUR + Histogram Shifting" method. As shown in Table 4, the experimental data demonstrated that the ME-capacity gain of "JUR + Histogram Shifting" over JUR is 0.25 (= 1.88 − 1.63) bpp and the PSNR loss is 1.66 (= 45.44 − 43.78) dB, leading to quality decay. Since the histogram shifting method is also a RDH method, the marked depth map by performing our JUR method can be correctly recovered from the marked map by performing the "JUR + Histogram Shifting" method; therefore, the problem of overriding depth copying pixels will not happen. Since the histogram shifting method is also a RDH method, the marked depth map by performing our JUR method can be correctly recovered from the marked map by performing the "JUR + Histogram Shifting" method; therefore, the problem of overriding depth copying pixels will not happen.

Table 5. PSNR comparison for different values of embedding capacity among the concerned rdh combinations

Combination		E-capacity (bpp)			
		0.1	0.3	0.5	0.7
BIC	D-RDH	48.95	44.31	42.55	41.48
	MD-RDH	48.86	44.44	42.87	42.67
GF	D-RDH	48.99	44.31	42.59	41.55
	MD-RDH	48.85	44.43	42.89	42.68
MM	D-RDH	48.95	44.30	42.57	41.56
	MD-RDH	49.76	45.33	43.55	42.73
EGU	D-RDH	48.97	44.31	42.58	41.54
	MD-RDH	48.87	44.44	42.88	42.65
SRI	D-RDH	48.95	44.30	42.55	41.48
	MD-RDH	49.62	45.23	43.46	42.68
FSRCNN	D-RDH	48.84	44.11	42.39	41.38
	MD-RDH	49.84	45.34	43.58	42.97
MSG-Net	D-RDH	48.94	44.31	42.56	41.51
	MD-RDH	48.88	44.49	42.92	42.74
CGI	D-RDH	48.99	44.27	42.61	41.67
	MD-RDH	49.56	45.10	43.45	42.73
BC+MV+BIC	D-RDH	48.94	44.29	42.51	41.44
	MD-RDH	48.77	44.35	42.79	42.60
JUR		**55.60**	**50.77**	**48.56**	**47.04**

5.2.2 PSNR comparison for different values of embedding capacity

Since the minimum of the ME-capacity values for all the concerned RDH combinations in Table 3 is about 0.7, we thus select the four embedding capacity (E-capacity), 0.1, 0.3, 0.5, and 0.7, as the base to compare PSNR of all the concerned RDH combinations. From Table 5, we observe that for each fixed E-capacity value, our JUR method has the highest PSNR value in boldface among all the concerned RDH combinations; the MD-RDH method is in second place and always outperforms D-RDH. Fig. 4 depicts the quality-E-capacity tradeoff comparison among ten considered combinations, each depth map upsampling method associated with the MD-RDH method. From the highest curve in Fig. 4, we observe that the proposed JUR method has the best quality-E-capacity tradeoff among the considered combinations. From the middle fat curve, MM-MD-RDH, SRI-MD-RDH, FSRCNN-MD-RDH, and CGI-MD-RDH have the similar quality-E-capacity tradeoff. From the lowest fat curve, BIC-MD-RDH, GF-MD-RDH, EGU-MD-RDH, MSG-Net-MD-RDH, and BC+MV+BIC-MD-RDH have the worst quality-E-capacity tradeoff. In summary, when setting the bpp values to 0.1, 0.1, 0.5, and 0.7, the PSNR improvement ratios of our JUR method are at least 11.6%, 12%, 11.4%, and 9.5%, respectively, relative to the nine comparative combinations.

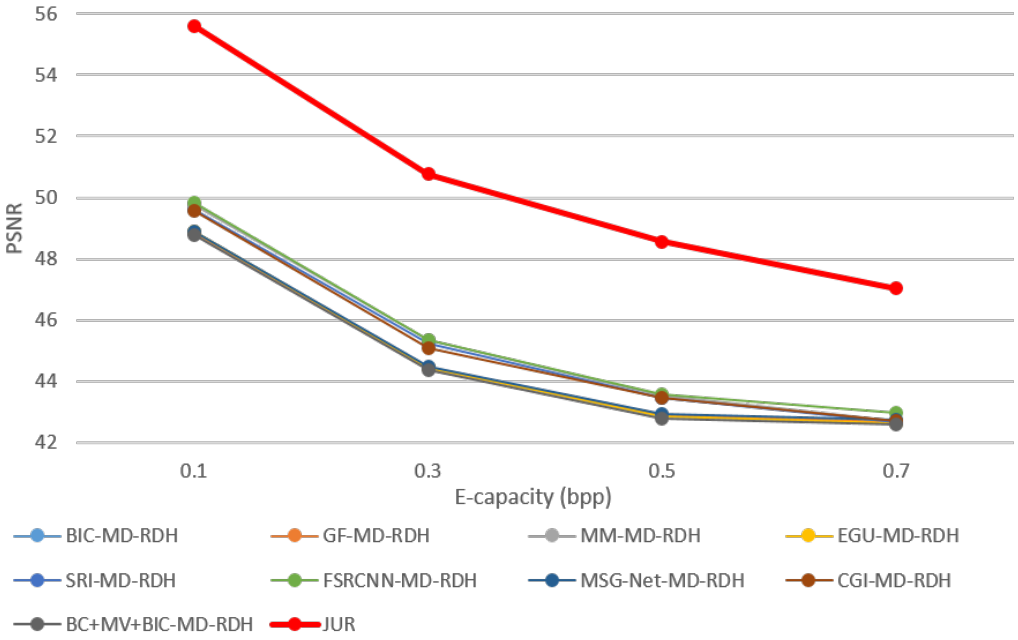


Fig. 4. The PSNR vs. E-capacity tradeoff comparison among the considered combinations

6 Conclusion

We have presented our fast method to partition all missing depth pixels into three disjoint regions, and then have presented our fast and effective region-based depth map upsampling method. Based on the test depth maps in Mobile3DTV, among our region-based upsampling method and the eight comparative methods, the comprehensive experimental results have demonstrated the quality, execution-time, visual effect, and non-deep learning supporting environment merits of our method. In addition, for the comparison among our joint upsampling and location map-free RDH method, i.e. JUR, D-RDH, and MD-RDH, the experimental results have justified the maximal embedding capacity and the PSNR (for different embedding capacity values) merits of our JUR method. One future research work is to combine the low-resolution depth guided joint trilateral upsampling [42] and our depth map upsampling method to handle upsampling and noise removal together.

Acknowledgment

This work was supported by Grant MOST-108-2221-E-011-077-MY3. The authors appreciate the valuable comments of the three anonymous reviewers and the proofreading help of Ms. C. Harrington to improve the manuscript.

Competing Interests

Authors have declared that no competing interests exist.

References

[1] Shao F, Jiang G, Yu M, Chen K, Ho YS. Asymmetric coding of multi-view video plus depth based 3D video for view rendering. IEEE Transactions on Multimedia. 2012;14(1):157-167.

[2] Fehn C. Depth-image-based rendering (DIBR), compression and transmission for a new approach on 3D-TV. Proc. SPIE. 2004;5291:93-104.

[3] Ni Z, Shi QY, Ansari N, Su W. Reversible data hiding. IEEE Trans. Circuits Syst. Video Technol. 2006;16(3):354-362.

[4] Durand F, Dorsey J. Fast bilateral filtering for the display of high-dynamic-range images. ACM Transactions on Graphics. 2002;21(3).

[5] Kopf J, Cohen MF, Lischinski D, Uyttendaele M. Joint bilateral upsampling. ACM Transactions on Graphics. 2007;26(3). Art. ID 96

[6] Kim J, Lee J, Han S, Kim D, Min J, Kim C. Trilateral filter construction for depth map upsampling. IEEE IVMSP Workshop. 2013;1-4.

[7] Zhang B, Allebach JP. Adaptive bilateral filter for sharpness enhancement and noise removal. IEEE Transactions on Image Processing. 2008;17(5):664-678.

[8] Jung SW. Enhancement of image and depth map using adaptive joint trilateral filter. IEEE Transactions on Circuits and Systems for Video Technology. 2013;23(2):258-269.

[9] Dhara BC, Chanda B. Color image compression based on block truncation coding using pattern fitting principle. Pattern Recognition. 2007;40(9):2408-2417.

[10] Yang J, Wright J, Huang T, Ma Y. Image super-resolution via sparse representation. IEEE Transactions on Image Processing. 2010;19(11):2861-2873.

[11] Ham B, Cho M, Ponce J. Robust image filtering using joint static and dynamic guidance. IEEE conference on computer vision and pattern Recognition. 2015;48234831.

[12] The Middlebury Stereo Datasets. [Online].
Available:http://vision.middlebury.edu/stereo/data/

[13] He K, Sun J, Tang X. Guided image filtering. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2013;35(6):1397-1409.

[14] Ferstl D, Reinbacher C, Ranftl R, Rther M, Bischof H. Image guided depth upsampling using anisotropic total generalized variation. IEEE International Conference on Computer Vision. 2013;993-1000.

[15] Li Y, Min D, Do MN, Lu J. Fast guided global interpolation for depth and motion. Proc. European Conference on Computer Vision, Speech and Signal Processing. 2016;717-733.

[16] Tallon M. Upsampling and denoising of depth maps via joint segmentation," EUSIPCO 2012 (20th European Signal Processing Conference 2012). 2012;27-31.

[17] Choi O, Jung S. A consensus-driven approach for structure and texture aware depth map upsampling. IEEE Transactions on Image Processing. 2014;23(8):3321-3335.

[18] Xie J, Feris RS, Sun MT. Edge-guided single depth image super resolution. IEEE Transactions on Image Processing. 2016;25(1):428-438.

[19] Zeyde R, Elad M, Protter M. On single image scale-up using sparse-representations. Proc. International Conference on Curves and Surfaces. 2010;711-730.

[20] Xie J, Chou CC, Feris R, Sun MT. Single depth image super resolution and denoising via coupled dictionary learning with local constraints and shock filtering. IEEE International Conference on Multimedia and Expo. 2014;1-6.

[21] Konno Y, Tanaka M, Okutomi M, Yanagawa Y, Kinoshita K, Kawade M. Depth map upsampling by self-guided residual interpolation. International Conference on Pattern Recognition. 2016;1394-1399.

[22] Wang L, Wu H, Pan C. Fast image upsampling via the displacement field. IEEE Transactions on Image Processing. 2014;23(12):5123-5135.

[23] Aodha OM, Campbell ND. Nair A. Brostow GJ. Patch based synthesis for single depth image super-resolution. IEEE European Conference on Computer Vision. 2012;71-84.

[24] Chang Y, Kim S, Ho Y. Depth upsampling methods for high resolution depth map. 2018 International Conference on Electronics, Information, and Communication (ICEIC). 2018;1-4.

[25] Chan D, Buisman H, Theobalt C, Thrum S. A noise-aware filter for real-time depth upsampling. Workshop on Multi-camera and Multi-modal Sensor Fusion Algorithms and Applications. 2008;1-12.

[26] Lee SB, Kwon S, Ho YS. Discontinuity adaptive depth upsampling for 3D video acquisition. Electronic Letters. 2013;49(25):1612-1614.

[27] Hui TW, Loy CC, Tang X. Depth map super-resolution by deep multi-scale guidance. Proc. European Conference on Computer Vision. 2016;353-369.

[28] Dong C, Loy CC, Tang X. Accelerating the super-resolution convolutional neural network. Proc. European Conference on Computer Vision. 2016;391-407.

[29] Kim B, Ponce J, Ham B. Deformable kernel networks for guided depth map upsampling; 2019. arXiv:1903.11286

[30] Yi Guo, Ji Liu, Depth edge guided CNNs for sparse depth upsampling; 2020. arXiv:2003.10138

[31] Keys RG. Cubic convolution interpolation for digital image processing. IEEE Transactions on Acoustics, Speech, Signal Processing. 1981;29:1153-1160.

[32] Chung KL, Yang WJ, Yang WN. Reversible data hiding for depth maps using the depth no-synthesis-error model. Information Sciences. 2014;269(6):159-175.

[33] Zhao Y, Zhu C, Chen Z, Yu L. Depth no-synthesis-error model for view synthesis in 3-D video. IEEE Transactions on Image Processing. 2011;20(8):2221-2228.

[34] Witten IH, Neal RM, Cleary JG. Arithmetic coding for data compression. Communications of the ACM. 1987;30(6):520-540.

[35] Shi X, Ou B, Qin Z. Tailoring reversible data hiding for 3D synthetic images. Signal Processing: Image Communication. 2018;64:46-58.

[36] Cormen TH, Leiserson CE, Rivest RL, Stein C. Asymptotic notation in introduction to algorithms. 3rd ed. London, U.K.: MIT Press; 2009. sec. 3.1.

[37] Fehn C, Kauff P, Op de Beeck M, Ernst F, Ijsselsteijn W, Pollefeys M, Vangool L, Ofek E, Sexton I. An evolutionary and optimised approach on 3D-TV. International Broadcast Conference. 2002;357-365.

[38] Watkins DS. Geometric approach to the least-squares problem in Fundamentals of matrix computations. New York: Wiley; 1991. Subsection 3.5.

[39] Execution code.[Accessed: 26 Jan. 2019]. [Online].
Available: ftp://140.118.175.164/Upsample&JUR/Codes

[40] Wang Z, Bovik AC, Sheikh HR, Simoncelli EP. Image quality assessment: From error visibility to structural similarity. IEEE Transactions on Image Processing. 2004;13(4):600-612.

[41] Experimental results. [Accessed: 26 Jan. 2019] [Online].
Available: ftp://140.118.175.164/Upsample&JUR/Visual-effect-Exp

[42] Yuan L, Jin X, Li Y, Yuan C. Depth map super-resolution via low-resolution depth guided joint trilateral up-sampling. J. of Visual Communication and Image Representation. 2017;46(7):280291.

Peer-review history:
The peer review history for this paper can be accessed here (Please copy paste the total link in your browser address bar)
http://www.sdiarticle4.com/review-history/57935