DEPTH MAP SUPER-RESOLUTION USING SYNTHESIZED VIEW MATCHING FOR DEPTH-IMAGE-BASED RENDERING

Wei Hu[o], Gene Cheung[#], Xin Li[*], Oscar Au[o]

[o] Hong Kong University of Science and Technology, [#] National Institute of Informatics,
[*] West Virginia University

ABSTRACT

In texture-plus-depth format of 3D visual data, texture and depth maps of multiple viewpoints are coded and transmitted at sender. At receiver, decoded texture and depth maps of two neighboring viewpoints are used to synthesize a desired intermediate view via depth-image-based rendering (DIBR). In this paper, to enable transmission of depth maps at low resolution for bit saving, we propose a novel super-resolution (SR) algorithm to increase the resolution of the received depth map at decoder to match the corresponding received high-resolution texture map for DIBR. Unlike previous depth map SR techniques that only utilize the texture map of the same view 0 to interpolate missing depth pixels of view 0, we use texture maps of the same and neighboring viewpoints, 0 and 1, so that the error between the original texture map of view 1 and the synthesized image of view 1 (interpolated using texture and depth maps of view 0) can be used as a regularization term during depth map SR of view 0. Further, piecewise smoothness of the reconstructed depth map is enforced by computing only the lowest frequency coefficients in Graph-based Transform (GBT) domain for each interpolated block. Experimental results show that our SR scheme out-performed a previous scheme by up to 1.7dB in synthesized view quality in PSNR.

***Index Terms*—** Multiview imaging, depth-image-based rendering, super-resolution

1. INTRODUCTION

Although there are numerous representations of 3D visual data proposed in the literature, *texture-plus-depth* format [1] has attracted much attention recently, due partly to strong interest in the free viewpoint TV (FTV) working group in the MPEG standardization body. In short, texture-plus-depth format means that texture maps (RGB images) and depth maps (per-pixel physical distances between captured objects in the 3D scene and the capturing camera) of multiple closely spaced viewpoints are encoded and transmitted from sender to the observing receiver. Having received the compressed data, the receiver can then synthesize an image of any freely chosen viewpoint via a depth-image-based rendering (DIBR) technique like 3D warping [2], using decoded texture and depth maps of two or more neighboring viewpoints as anchors. Transmission of large texture and depth maps of multiple viewpoints, however, translates to a high network transmission cost. Thus, compression of texture-plus-depth format of 3D data is an important research problem.

Unlike the more well studied and understood texture maps, depth maps are a relatively new kind of visual data that possesses unique characteristics like smooth surfaces and sharp edges. See Fig. 1 for an illustration. Depth maps can be acquired either directly using depth-sensing cameras [3], typically at lower resolution (LR) than corresponding texture maps, or derived from texture maps using stereo-matching algorithms. In either case, it has been proposed [4] to encode depth maps at LR to save coding bits, and then at receiver, increase the resolution of the decoded LR depth maps (*super-resolution* (SR)) to match the high resolution (HR) of the corresponding texture maps for DIBR. In the first case, this means encoding the depth maps at camera-captured LR; in the second, it means down-sampling the stereo-matched depth maps to LR prior to encoding.

Fig. 1. The depth map and corresponding texture map of the left viewpoint for *Teddy*.

Compared to the more general image SR problem in computer vision [5], depth map SR can be performed more simply because of the aforementioned unique characteristics and availability of corresponding HR texture maps from the same viewpoints as helpful side information. As an example, [4] proposed to super-resolve depth map of a given view 0 using HR texture map of the same viewpoint 0 during depth pixel interpolation. In this paper, instead we use texture maps of the

same and neighboring viewpoints, 0 and 1, so that the error between the original texture map of view 1 and the synthesized image of view 1 (interpolated using texture and depth maps of view 0) can be used as a regularization term during depth map SR of view 0. Further, piecewise smoothness of the reconstructed depth map is enforced as a prior by computing only the lowest frequency coefficients in Graph-based Transform (GBT) domain [6] for each interpolated block. Experimental results show that our SR scheme out-performed [4] by up to 1.7dB in synthesized view quality in PSNR.

The outline of the paper is as follows. We first discuss related work in Section 2. We then overview GBT, previously proposed in [6], in Section 3. We discuss our multiview imaging system in Section 4, and present our depth map SR algorithm in Section 5. Finally, we present experimental results and conclusions in Section 6 and 7, respectively.

2. RELATED WORK

While compression of texture maps is well studied, compression of depth maps is relatively new, and has been the focus of many recent research efforts [7, 6, 8]. Many of the proposed methods exploit depth maps' unique characteristics of sharp edges and smooth surfaces for compression gain. Though the goal of compact representation of depth information is the same, in this paper we develop a novel SR algorithm to increase resolution of received depth maps at decoder to match HR texture maps for DIBR, so that depth maps can be encoded in LR at encoder, saving coding bits.

Previous depth map SR algorithms typically exploit structure similarity between depth maps and corresponding texture maps of the same viewpoints. [9] proposed to super-resolve depth maps with the linear MPEG up-sampling filter, where filtering across edges is not allowed, thus preserving edge sharpness. The edge information of the super-resolved depth map is extracted from the HR texture map of the same viewpoint, where HR edges due to textural changes, rather than foreground-background transitions, are eliminated by checking the local depth intensity gradients in the LR depth map. Also taking advantage of the structure similarity, to interpolate a pixel u, [4] computed a weight for each of u's four nearest neighbors, based on pixel distance in the depth map as well as color difference in the corresponding texture map. Then, the depth map value of the neighbor with the largest weight will be selected to interpolate pixel u.

Different from previous proposals where the single texture map from the same viewpoint as the LR depth map is used, in this paper, we utilize texture maps of the same and neighboring viewpoints, 0 and 1, so that the error between the original texture map of view 1 and the synthesized image of view 1 (interpolated using texture and depth maps of view 0) can be used as a regularization term during depth map SR of view 0. We will detail our approach in Section 5.

3. GRAPH-BASED TRANSFORM

In our depth map SR scheme, we use GBT to enforce the piecewise-smooth prior in a pixel block during depth pixel interpolation. To understand GBT, we briefly discuss the three-step GBT construction procedure [6] in this section. First, prominent edges in a $n \times n$ pixel block are identified. Then, a graph describing the pixel connectivity given the identified edges (two neighboring pixels are connected except when divided by an edge) is constructed. Finally, an adaptive transform is built based on the connectivity graph.

In the first step, we identify edges (large pixel value transition across neighboring pixels) in a block. Edge sharpness can be preserved even if filtering in subsequently constructed adaptive GBT domain is performed.

In the second step, we treat each pixel in the $n \times n$ block as a node in a graph $\mathcal{G}$, and connect it to its four or eight immediate neighbors in the block, resulting in a 4- or 8-connectivity graph. Then, if there is an edge between two neighboring pixels / nodes, we eliminate their connection. Given the connectivity graph, we can define an adjacency matrix $\mathbf{A}$, where $\mathbf{A}(i,j) = \mathbf{A}(j,i) = 1$ if pixel positions i and j are connected, and 0 otherwise. We can similarly compute the degree matrix $\mathbf{D}$, where $\mathbf{D}(i,i)$ is the number of connections for node i, and $\mathbf{D}(i,j) = 0$ for all $i \neq j$.

In the third step, using computed $\mathbf{A}$ and $\mathbf{D}$, we can compute the Laplacian matrix $\mathbf{L} = \mathbf{D} - \mathbf{A}$ [10]. If we now project a signal $\mathbf{x}$ in the graph $\mathcal{G}$ onto the eigenvectors of the Laplacian $\mathbf{L}$, it becomes the spectral decomposition of the signal; i.e., it provides a "frequency domain" interpretation of signal $\mathbf{x}$ given graph support $\mathcal{G}$. Hence, we can construct GBT transform using eigenvectors of $\mathbf{L}$. In particular, we can stack pixels in the $n \times n$ block into a length-n^2 vector and compute $\mathbf{y} = \mathbf{E}^t \cdot \mathbf{x}$, where $\mathbf{E}$ is a matrix with eigenvectors of $\mathbf{L}$ as columns.

Using constructed GBT $\mathbf{E}$, we can interpret piecewise-smoothness of a depth block $\mathbf{x}$ as follows: only the lowest L frequency components of $\mathbf{y} = \mathbf{E}^t \cdot \mathbf{x}$ are non-zero; $L \geq 1$ depends on how "smooth" the depth block is assumed to be *a priori*. We will use this interpretation of piecewise-smoothness in our depth map SR algorithm discussed later.

4. MULTIVIEW IMAGING SYSTEM

We first overview the multiview imaging system we envision for our depth map SR problem. At sender, texture maps of left and right views at HR are encoded along with depth maps of the same viewpoints at LR. As discussed in the Introduction, texture maps are captured in HR, while depth maps can be either captured directly in LR, or derived in HR via stereo-matching and subsequently down-sampled to LR. To preserve edge sharpness, instead of traditional DCT, each LR depth block is transform-coded using GBT as done in [6], so that filtering across detected edges is avoided and detected LR

edges are losslessly encoded and transmitted to receiver for inverse transform. Both compressed texture and depth maps are transmitted to receiver in texture-plus-depth format to represent the 3D scene.

At receiver, after decoding the received texture and depth maps, LR depth maps must be super-resolved to the same resolution of the HR texture maps. Receiver can then synthesize texture map of any intermediate virtual view between left and right captured views using the decoded texture maps and super-resolved depth maps of left and right views as anchors via DIBR. Essentially, a synthesized pixel in the virtual view is a weighted average of the corresponding pixels in the left and right texture maps, where the weights are inversely proportional to the distance between the virtual viewpoint and the left and right views. A synthesized pixel that has no corresponding pixels in both the left and right views is filled using an inpainting technique. We discuss next how SR of depth maps at decoder can be optimized for best possible DIBR-synthesized virtual view quality.

5. DEPTH SUPER RESOLUTION WITH SYNTHESIZED VIEW MATCHING

While existing depth SR techniques use a single HR texture map as side information to help super-resolve a LR depth map of the same viewpoint, we exploit disparity information[1] in both left and right HR texture maps for SR of a single depth map (left or right). Specifically, for each candidate interpolated pixel in the left HR depth map, one can map a left texture pixel to the corresponding pixel in the right HR texture map via DIBR, resulting in a *synthesized view matching error*—the difference in intensity between the two pixels. The idea is then to super-resolve a depth map so that it is consistent with the LR depth map *and* minimizes the sum of synthesized view matching errors of all interpolated depth pixels.

5.1. Synthesized View Matching Error Approximation

Given the disparity information provided by the left depth map, a pixel in the left texture map can be mapped to a pixel in the right texture map. The reverse procedure is the well-known *stereo matching* problem, which finds corresponding pixel patches between the left and right texture maps in order to estimate a disparity map.

Assuming that a left pixel that is mapped geometry-correctly to the right should have similar intensity value as the right corresponding pixel[2], we define a per-pixel error function as the difference in intensity between corresponding pixels for a given disparity. This error function, which we refer to as *synthesized view matching error*, is defined as follows for rectified texture maps:

$$E_l(m,n) = |I_l(m,n) - I_r(m, n - X_l(m,n) * r)| \quad (1)$$

where $I_l(m,n)$ is the intensity of the pixel in row m and column n in the left texture map, and it is mapped to the pixel $I_r(m, n - X_l(m,n) * r)$ in the right texture map with disparity $X_l(m,n)$ and scaling factor r.

In cases when no correspondence can be found in the right texture map due to camera shift, occlusion or unknown disparity in the original disparity map, the above definition of synthesized view matching error is not meaningful. In these relatively rare cases, we set the synthesized view matching error to zero.

5.1.1. *Approximation of Matching Error*

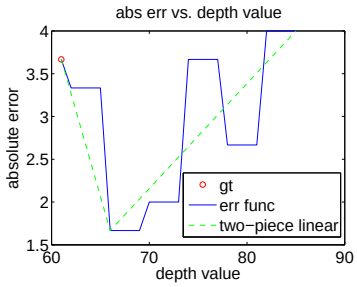


Fig. 2. Synthesized view matching error function and corresponding two-piece linear approximation.

For ease of later optimization, we approximate the synthesized view matching error function with a two-piece linear function:

$$g_i(x_i) = \max_j \{a_i(j)x_i + b_i(j)\} \quad j \in \{1,2\} \quad (2)$$

where x_i is the disparity value of pixel i, and $a_i(j)$'s and $b_i(j)$'s are respectively the slopes and y-intercepts of the two-piece linear function, $j \in \{1,2\}$.

For a given depth pixel i, the approximation process is as follows. First, we find the minimum and maximum disparity, $d_{\min}$ and $d_{\max}$, of the four immediate neighbors of pixel i in LR depth map, and define the domain of x_i as $[(1-\delta)d_{\min}, (1+\delta)d_{\max}]$, where δ, $0 \leq \delta \leq 1$, is a predefined constant. This is based on the assumption that the depth value of a pixel is within or close to the depth range of its neighbors.

Next, we calculate the synthesized view matching error for disparity values in the defined domain using (1). Then, we find the minimum synthesized view matching error in the domain, and construct line segments from it to the corresponding errors at $d_{\min}$ and $d_{\max}$ to get a two-piece linear function. Fig. 2 shows an example of the actual synthesized view matching error function and corresponding two-piece linear

[1]Though there is a one-to-one correspondence between depth and disparity, we assume disparity maps are encoded instead of depth maps.

[2]In the case where the 3D scene is Lambertian, the difference is zero.

1. Set a search range for pixel i from the minimum and maximum disparity of its four nearest neighbors. The lower and upper bound are set as the two endpoints of the two-piece linear function.
2. Compute the synthesized view matching error of each disparity value within the search range.
3. Find the disparity with the minimum synthesized view matching error and set it as the turning point of the two-piece linear function.

Fig. 3. Construction of the two-piece linear function for synthesized view matching error approximation.

approximation. The left and right piece line segments are defined by slopes $a_i(1)$ and $a_i(2)$ and y-intercepts $b_i(1)$ and $b_i(2)$, respectively.

Fig. 3 summarizes how we construct the two-piece linear function. Note that since the constructed two-piece linear function is the point-wise maximum of two affine functions, it is a convex function.

5.2. Problem Formulation

As illustrated in Fig. 1, depth maps are piecewise-smooth 2D functions—smooth interior surfaces within sharp edges. An elegant way of interpreting the piecewise-smooth prior, as discussed in Section 3, is through GBT: only GBT low-frequency components are non-zero. We hence pose our SR problem in GBT domain, where a depth pixel block $\mathbf{X}$ is represented by inverse GBT Φ^{-1} and transform coefficients α:

$$\mathbf{X} = \Phi(\mathbf{e})^{-1}\alpha, \tag{3}$$

where $\mathbf{e} = \{e_1, e_2, \ldots\}$ denotes the set of defined edges in a depth pixel block, from which the adaptive GBT Φ is constructed as described in Section 3.

Our objective function is a weighted sum of: i) the square difference between the received LR depth block $\mathbf{Y}$ and the low-pass filtered and sub-sampled version of the reconstructed HR block $\mathbf{X} = \Phi(\mathbf{e})^{-1}\alpha$ in pixel domain, and ii) the total synthesized view matching error of all depth pixels in the HR block:

$$\begin{aligned} \min_{\mathbf{e},\alpha} \quad & \|DH\Phi(\mathbf{e})^{-1}\alpha - \mathbf{Y}\|_2 + \lambda \sum_i g_i(\phi_i(\mathbf{e})^{-1}\alpha) \\ \text{s.t.} \quad & \alpha_k = 0, k \in \{L+1, L+2, \ldots\} \end{aligned} \tag{4}$$

where λ is a weight parameter specifying the desired tradeoff between the SR term (first term) and synthesized view matching error term, H is a low-pass filter prior to down-sampling, and D is the down-sampling operator. $\phi_i(\mathbf{e})^{-1}$ is the i-th row in the inverse GBT matrix $\Phi(\mathbf{e})^{-1}$. To enforce piecewise smoothness, we only allow L (e.g., $L = 2$) lowest GBT frequency coefficients to be non-zero.

5.3. Optimization Algorithm

Problem (4) is difficult with edge set $\mathbf{e}$ as discrete variables and transform coefficients α as continuous variables. We propose to solve it in two separate steps: (i) given LR depth block $\mathbf{Y}$ and LR edges, we guess possible HR edges $\mathbf{e}$; (ii) Given discovered HR edges $\mathbf{e}$, we can find optimal GBT frequency coefficients α.

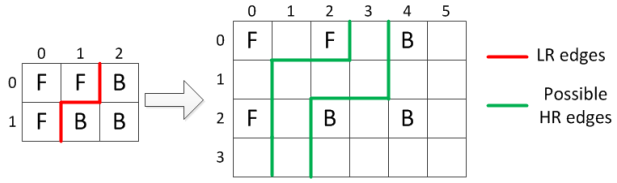


Fig. 4. An example of depth map super-resolution (from a 2×3 pixel block with pixel intensities F and B to a 4×6 pixel block). Possible HR edges can be inferred from the LR edges.

5.3.1. Finding HR Edges

First, we define a feasible search space for HR edges using the LR depth map and LR edges. As shown in Fig. 4, each LR edge maps to a set of possible HR edges. For example, an edge between pixels $(0,1)$ and $(0,2)$ in the first row of the LR block means that there should be an edge either between pixels $(0,2)$ and $(0,3)$, or between pixels $(0,3)$ and pixel $(0,4)$ in the corresponding HR block. Hence, we can get sets of possible HR edges in a pixel block using LR edges. In order to compute synthesized view matching error of a depth block, for a given selection of HR edges within the search space, we can fill each missing HR depth pixel (e.g., pixel $(0,3)$ in Fig. 4) using the nearest filled HR depth pixel on the same side of the edge (e.g. pixel $(0,4)$ if there is edge between pixels $(0,2)$ and $(0,3)$).

Further, since the contour of a physical object tends to be smooth, we consider an *edge-consistent penalty* in addition to the synthesized view matching error during HR edge search, i.e., edges that go back and forth within a block will be penalized more. We can now solve for the HR edges by minimizing the weighted sum of the synthesized view matching error and edge-consistent penalty as follows:

$$\min_{\mathbf{e}} \sum_i g_i(\mathbf{X}_i(\mathbf{e})) + \mu \sum_j |\Delta(e_j, e_{j-1}) - \Delta(e_{j+1}, e_j)| \tag{5}$$

where $\Delta(e_j, e_{j-1})$ is the direction from edge e_{j-1} to e_j, and μ is a parameter to control the relative importance of the synthesized view matching error and edge-consistent penalty.

To solve the combinatorial problem (5) is still difficult. We employ a greedy search strategy to efficiently find a set of good HR edges. Specifically, starting from an initial guess of HR edges, we compare each edge to its closest alternative and select the one with the smaller penalty. We then repeat until the overall penalty does not decrease further.

5.3.2. Interpolating depth pixels

After finding appropriate HR edges $\mathbf{e}$, (4) becomes a convex optimization problem and can be easily solved using a convex optimization tool *CVX* [3]. With the optimal non-zero GBT low frequency coefficients α solved, we can get the final super-resolved depth map $\mathbf{X}$ via inverse GBT $\Phi(\mathbf{e})^{-1}$.

6. EXPERIMENTATION

6.1. Experimental Setup

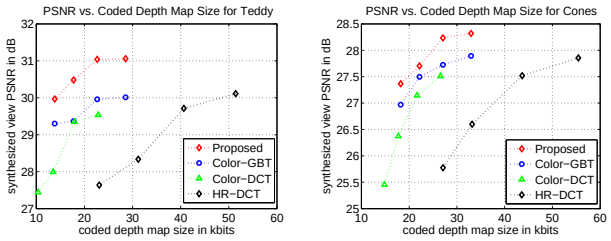
We tested our proposed SR algorithm based on synthesized view matching and GBT (`SVM-GBT`) on two Middlebury multiview image sets `Teddy` and `Cones`[4]. Experiments were conducted in the multiview imaging system described in Section 4, where texture maps of left and right views are encoded and decoded using H.264/AVC Reference Software JM 17.1 [5], while depth maps are low-pass-filtered and down-sampled at encoder and then super-resolved at decoder. GBT is used to encode the LR depth map and fixed quantization parameter (QP) values of 24, 28, 32 and 36 are adopted during compression. The super-resolved depth maps are then used for DIBR with a simple implementation of 3D warping [2]. Our coding scheme is compared against three other schemes: (1) H.264 intra with DCT encoding original HR depth maps (`HR-DCT`); (2) H.264 intra with DCT encoding reduced resolution LR depth maps, which are super-resolved at decoder with the `color-based` depth SR algorithm proposed in [4]—the state-of-the-art depth map SR approach to our best knowledge (`Color-DCT`); (3) GBT encoding reduced resolution LR depth maps followed by the `color-based` depth SR (`Color-GBT`).

The weight parameters, λ in (4) and μ in (5), are experimentally set to 0.01 and 10, respectively. The low-pass filter H in (4) was assumed to be an identity matrix for simplicity. Note, however, that our formulation is sufficiently general that it can easily adapt to most filters.

6.2. Experimental Results

First, we tried different values for the number of non-zero low frequency GBT coefficients L in (4). It turns out that $L = 2$ is a good enough value for `Teddy` and `Cones`.

Fig. 5 demonstrates RD curves of our proposed method and the aforementioned three competing schemes, with the synthesized view PSNR calculated with respect to the ground truth middle texture map. We can see while `Color-GBT` performs better than `Color-DCT` and `HR-DCT`, our proposed `SVM-GBT` achieves synthesized view PSNR gain of up to 1dB and 0.4dB for `Teddy` and `Cones` respectively when compared to `Color-GBT`, and up to 1.7dB and 0.8dB gain when compared to `Color-DCT`. Further, 3.4dB and 2.1dB gain for `Teddy` and `Cones` respectively is achieved by our method in comparison with the baseline scheme `HR-DCT`.



(a) Teddy (b) Cones

Fig. 5. RD curves of different coding schemes for `Teddy` and `Cones` respectively.

Besides objective quality, Fig. 6 and Fig. 7 show super-resolved depth maps and corresponding synthesized views of `Teddy` and `Cones` respectively with different coding schemes at comparable bit rate. We see that edges produced by `SVM-GBT` are cleaner and sharper, and the interior surfaces are much less contaminated by blocking artifacts than that of `Color-DCT` and `HR-DCT` respectively. More importantly, the synthesized view facilitated by our generated depth map with `SVM-GBT` has much less ringing artifacts along object boundary compared to that of the other two methods.

7. CONCLUSION

Compact representation of depth maps is important for texture-plus-depth format of 3D visual data, which is commonly used for view synthesis at decoder via depth-image-based rendering (DIBR). In this paper, we propose a super-resolution (SR) algorithm to increase the resolution of the received depth maps to match high resolution (HR) texture maps, so that depth maps can be transmitted at encoder at lower resolution (LR), saving coding bits. The key novelty is to use HR texture map of the same and neighboring views to compute a synthesized view matching error, which is used as a regularization term during SR. Piecewise smoothness is enforced by searching only low frequency components in the Graph-based Transform (GBT) of the pixel block during interpolation. Experimental results show that our SR algorithm outperformed previous ones by up to 1.7dB in synthesized view quality in PSNR.

8. REFERENCES

[1] P. Merkle, A. Smolic, K. Mueller, and T. Wiegand, "Multi-view video plus depth representation and coding," in *IEEE International Conference on Image Processing*, San Antonio, TX, October 2007.

[2] W. Mark, L. McMillan, and G. Bishop, "Post-rendering

[3] http://cvxr.com/cvx/

[4] http://cat.middlebury.edu/stereo/newdata.html

[5] http://iphome.hhi.de/suehring/tml/

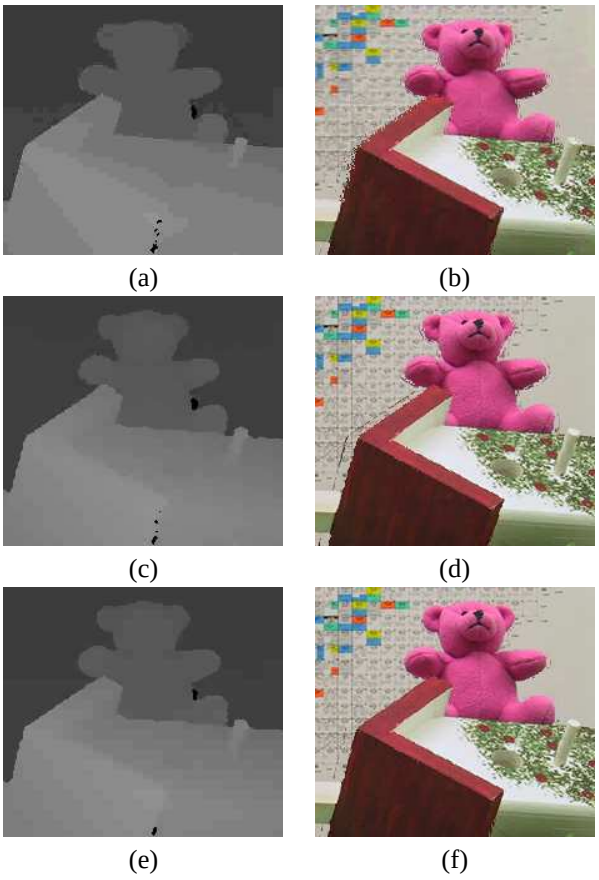


Fig. 6. The super-resolved right depth maps and corresponding synthesized views of `Teddy` by different coding schemes at comparable bit rate: (a)-(b) `HR-DCT`; (c)-(d) `Color-DCT`; (e)-(f) `SVM-GBT`.

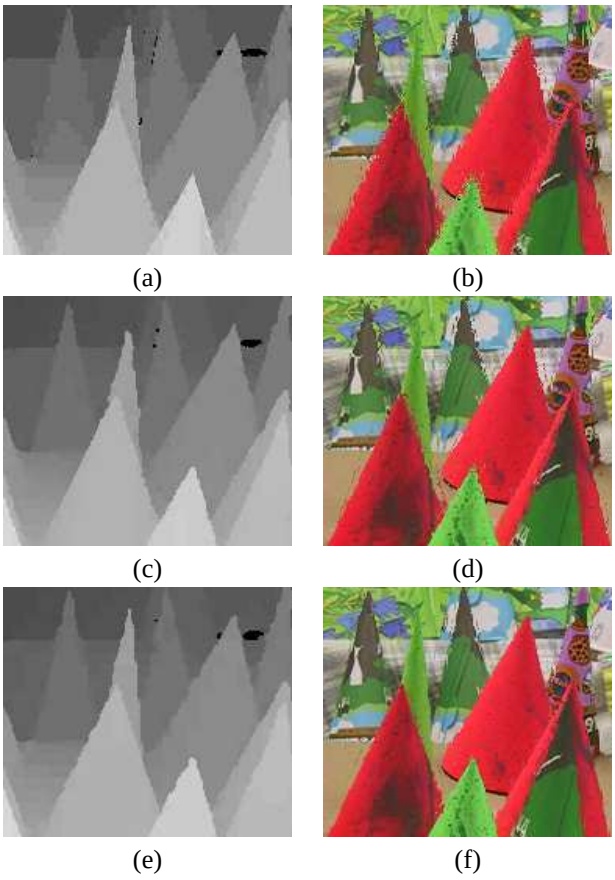


Fig. 7. The super-resolved left depth maps and corresponding synthesized views of `Cones` by different coding schemes at comparable bit rate: (a)-(b) `HR-DCT`; (c)-(d) `Color-DCT`; (e)-(f) `SVM-GBT`.

3D warping," in *Symposium on Interactive 3D Graphics*, New York, NY, April 1997.

[3] S. Gokturk, H. Yalcin, and C. Bamji, "A time-of-flight depth sensor—system description, issues and solutions," in *Conference on Computer Vision and Pattern Recognition Workshop (CVPRW)*, Washington, DC, June 2004.

[4] M. Wildeboer, T. Yendo, M. Tehrani, T. Fujii, and M. Tanimoto, "Depth up-sampling for depth coding using view information," in *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON)*, Antalya, Turkey, May 2011.

[5] X. Li and M. Ochard, "New edge-directed image interpolation," in *IEEE Transactions Image Processing*, vol. 10, no.10, October 2001, pp. 1521–1527.

[6] G. Shen, W.-S. Kim, S. Narang, A. Ortega, J. Lee, and H. Wey, "Edge-adaptive transforms for efficient depth map coding," in *IEEE Picture Coding Symposium*, Nagoya, Japan, December 2010.

[7] M. Maitre, Y. Shinagawa, and M. Do, "Wavelet-based joint estimation and encoding of depth-image-based representations for free-viewpoint rendering," in *IEEE Transactions on Image Processing*, vol. 17, no.6, June 2008, pp. 946–957.

[8] G. Cheung, J. Ishida, A. Kubota, and A. Ortega, "Transform domain sparsification of depth maps using iterative quadratic programming," in *IEEE International Conference on Image Processing*, Brussels, Belgium, September 2011.

[9] E. Ekmekcioglu, M. Mrak, S. Worrall, and A. Kondoz, "Utilisation of edge adaptive upsampling in compression of depth map videos for enhanced free-viewpoint rendering," in *IEEE International Conference on Image Processing*, Cairo, Egypt, Nov. 2009.

[10] D. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," in *Elsevier: Appplied and Computational Harmonic Analysis*, vol. 30, April 2010, pp. 129–150.