Visual Comput (2005) 21: 1–8
DOI 10.1007/s00371-005-0337-8

ORIGINAL ARTICLE

Sashi Kumar Penta
P.J. Narayanan

Compression of multiple depth maps for IBR

Published online: ?? ?? 2005

S.K. Penta (✉) · P.J. Narayanan
Centre for Visual Information Technology
International Institute of Information Technology
Gachibowli, Hyderabad 500019. INDIA.
{sashikumar, pjn}@iiit.ac.in

Abstract Image-based rendering techniques include those with geometry and those without. Geometric information in the form of a depth map aligned with the image holds a lot of promise for IBR due to the several methods available to capture it. It can improve the quality of generated views using a limited number of views. Compression of light fields or multiple images has attracted a lot of attention in the past. Compression of multiple depth maps of the same scene has not been explored much in the literature. We propose a method for compressing multiple depth maps in this paper using geometric proxy. Different quality of rendering and compression ratio can be achieved by varying different parameters. Experiments show the effectiveness of the compression technique on several model data.

Keywords Image-based rendering · Depth map · Data compression

1 Introduction

Image-based modeling and rendering (IBMR) has attracted much attention in the past decade. The potential to produce new views of a real scene with a high degree of realism makes IBR very appealing. IBMR aims to capture a scene or environment using a number of (carefully placed) cameras. Any view of the virtual environment can be generated subsequently from the captured scene. Different internal representations are used by different IBR methods. These can be divided into two broad categories: representations without any geometric model and representations with geometric model of some kind.

Early IBR efforts produced new views of scenes given two or more images of it [4, 24]. Many later techniques followed this philosophy of generating new views purely from images [1, 8, 16, 21, 25]. Some of these methods require a large number of input views for capturing a scene satisfactorily. This makes them practically unusable other than for static scenes. The internal representation computed was bulky and needed sophisticated compression schemes. Lumigraph rendering showed that the availability of even approximate geometry can reduce the requirements on number of views drastically [8]. View-dependent texture mapping [6] used known geometry and selects textures relevant to the view being generated to model architectural monuments. Unstructured Lumigraph [3] extended this idea to rendering using an unstructured collection of views and approximate models.

The virtualized reality system captured and modeled dynamic scenes for subsequent rendering using a studio with a few dozens of cameras and multi-view stereo [22]. The combination of a depth map and an aligned texture map captured using cameras hold a lot of promise for modeling dynamic scenes. Cameras are non-intrusive and fast sensors that are inexpensive enough to be deployed in numbers. Similar approaches have been tried in recent years for modeling, immersion, videoconferencing, etc. [2, 29]. A layered representation with full geometry recovery for modeling and rendering dynamic scenes was reported by Zitnick et al. [31].

In this paper, we focus on image-based models that use depth maps and aligned texture maps. This combination is sometimes referred to as depth images [19] or as depth+texture representation [23]. Please see [23] study of this representation from the point of view of generation and rendering. In this paper, we present an approach to compress the depth maps of a multi-depth image representation. We describe the depth image representation in the next section and present our proxy-based compression scheme in Sect. 3. Experimental results are presented in Sect. 4 and conclusions in Sect. 5.

2 Depth images

A depth image consists of an image and a depth map aligned with it. Both are acquired from a point of view of a camera whose location is the reference point or origin of the depth image. The calibration parameters of the camera are also included in the representation so that the 2D sampling grid and the imaging rays of the depth image can be known in 3D space. The depth map is a two-dimensional array of real values. Location (i, j) of the depth map stores the depth or normal distance to the closest 3D point along the imaging ray corresponding to the pixel (i, j). Location (i, j) of the image stores the color returned by that ray. The combination of a depth map and an image describes the local structure from one viewpoint and can be used to reconstruct the views of the scene from other viewpoints.

Computer vision provides several methods to compute depth image structure, called the 2-$\frac{1}{2}$D structure, using different clues such as motion, shading, focus, stereo, interreflections, etc. Stereo holds a lot of promise in estimating distances using triangulation in multiple views.

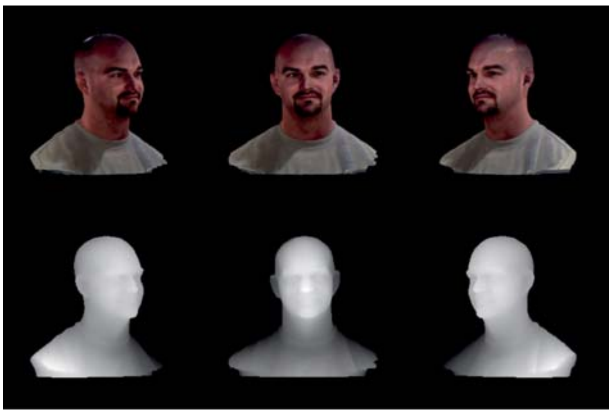

Fig. 1. Image and depth map from three viewpoints for the male model. Depth map is shown immediately below the corresponding image

Several scanning devices are also available today that can capture depth images directly. Figure 1 shows some depth images for a male model.

Depth images have been used for image-based modeling and rendering in the past [9, 19, 23]. Since a single depth image represents only a partial structure, holes will appear in reconstructed views if its location is far from the reference point of the depth image. Multiple depth images with suitably distributed origins can provide hole-free view of the scene. Virtualized reality captured these depth images by placing a few dozens of cameras around the scene [14]. A discussion and analysis of the multiple depth image representation, ways to acquire them, rendering techniques for them, etc., can be found in [23].

Depth maps can be represented and stored essentially as images. The range of the depth map depends on the extent of scene of interest and the resolution of the structure recovery algorithm. A 16-bit, fixed-point representation of depth values will suffice in most cases as it can cover a scene extent of 65 meters with a resolution of 1 millimeter. Thus, each depth image can be represented using a color image in a suitable format, 16-bit per pixel image for the depths and a 3×4 calibration matrix for the corresponding camera.

The representation of a scene, therefore, consists of a set $\mathcal{S}$ of depth images. Each depth image $\mathcal{D}_i$ consists of a depth map $\boldsymbol{D}_i$, an image $\boldsymbol{I}_i$, and the calibration matrix $\boldsymbol{C}_i$. $\boldsymbol{D}, \boldsymbol{I}$ values are represented against a 2D grid as images. $\boldsymbol{C}$ values consist of 12 real numbers.

The topic of compressing multiple images of a scene, called light field usually, has attracted a lot of attention. Several methods have been reported for this. Early light field compression techniques such as vector quantization [16], disparity compensation techniques [7, 28], wavelet transform-based techniques [7, 12], and incremental representation exploiting spatial coherence [30] are used to compress the images alone without using any geometry. Enhancement in prediction of accuracy is shown in [18] by using geometry such as depth maps and 3D models. Spatially encoded video [10] uses spatial coherence to encode sample images using model-based depth information. All these techniques look for compression of light field or multi-view images. Compression of multiple depth maps of the common scene has not been worked on until now. We present a method to compress multiple depth maps in this paper.

Performance of rendering of novel views can be increased by using geometry proxies [3, 8, 13, 20, 27]. Geometry proxies are used to increase appearance prediction by depth correction [26]. Geometry proxy is an approximated geometric model. All these techniques used geometry proxies for increasing the quality of rendering views. In this paper, we use geometry proxy for compressing multiple depth maps.

3 Compressing multiple depth maps

Depth maps contain distances from a point usually organized on a regular, 2D sampling grid. Each depth map is, therefore, like an image. Adjacent grid-points are highly correlated as depth values change slowly except at occlusion boundaries. The spatial redundancy suggests the use of standard image compression techniques for depth maps. The lossy methods like JPEG are perceptually motivated and try to minimize the visual appearance between the original image and the compressed image. These may not work well for depth maps as the programs that use depth maps have different characteristics from the human visual system. Krishnamurthy et al. [15] used ROI coding and reshaping of dynamic range where the accuracy of depth is crucial for compressing depth maps.

Spatial redundancy along the 2D grid provides one source of compression for an individual depth map. What additional redundancy can be exploited when multiple depth maps of the same scene are taken together? There will be considerable overlap and commonality in them. Since they represent the 2-$\frac{1}{2}$ structure of the common scene, the redundancy lies in the fact that they describe the same scene. The common scene structure can be modeled separately and can provide a basis for a compressed representation of multiple depth maps.

3.1 Geometry proxy

We use a *geometry proxy* – an approximate description of the scene – to model the common, position-independent, scene structure. The geometry proxy $\boldsymbol{P}$ can be a parametric model or an approximate triangulated model. The proxy is assumed to represent the geometric structure of the scene adequately. The depth value of each grid point is replaced by the difference of the input depth map from the distance along the same direction to the proxy. This is similar to a predictive compression framework with the (projection of the) geometry proxy providing the prediction in each depth image. The difference at each grid point between the predicted and the actual depth values is stored as residues. The residues are expected to be small in range everywhere if the proxy is a good approximation of the scene geometry. Naturally, the size/quality of the proxy can be traded off against the size of the residues or the net compression ratio.

The geometry proxy can be a parametric object like a bounding box, a best-fit ellipsoid, or an approximate polygon-based model of the scene. Such proxies can be created from the input depth maps themselves. These data-dependent proxies can result in very small residues.

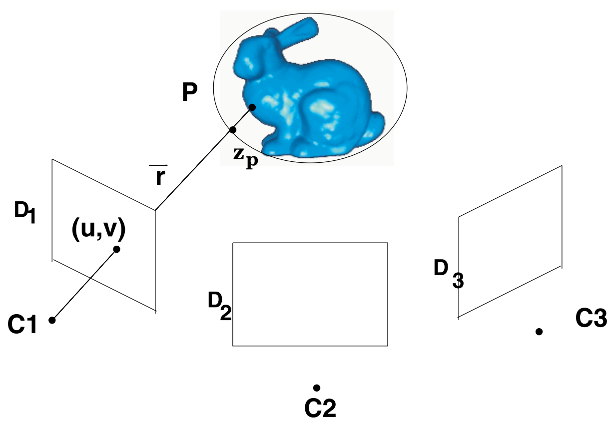

Fig. 2. The geometry proxy $\boldsymbol{P}$ (an ellipsoid in this case) represents common structure. It is projected to every depth image. The difference in depth between the proxy and the scene is encoded as the residue at each grid point of each depth map

Algorithm 1 RepresentProxy($\mathcal{S}$, $\boldsymbol{P}$)

1. For each $\boldsymbol{D}_i$ in collection $\mathcal{S}$ do,
2. For each grid point (u, v) of the depth map do,
3. Compute the ray $\vec{r}$ for pixel (u, v) using $\boldsymbol{C}_i$,
4. Intersect $\vec{r}$ with proxy $\boldsymbol{P}$. Let z_p be the z-coordinate of the intersection point
5. Set $\boldsymbol{R}_i(u, v) = z_p - \boldsymbol{D}_i(u, v)$.

Figure 2 shows how the residue images $\boldsymbol{R}_i$ are computed by projecting the proxy to each depth image. A simple ellipsoid proxy is shown in the figure. The residue image is also defined over the same 2D grid as the depth image. Algorithm 1 outlines the steps involved in constructing a proxy-based representation from a collection of depth images using a given proxy.

The proxy-based representation of the collection $\mathcal{S}$ replaces each depth map $\boldsymbol{D}_i$ by a residue image $\boldsymbol{R}_i$ and adds a single proxy $\boldsymbol{P}$ that encodes the common structure. The proxy uses a representation suitable for its type.

3.2 Computing geometry proxy

Different types of geometry proxy could be used to represent multiple depth maps. Parametric proxies include a box, ellipsoid, polynomial surfaces, etc. The model parameters can be estimated from the depth maps once the parametric form is fixed. Each depth map samples the scene at a few points. All depth maps can be brought to a common 3D space giving a large 3D point cloud. The bounding box of this point cloud can serve as a box proxy. The best fitting ellipsoid of the point cloud can make a reasonable proxy for the collection of depth maps and can be computed using principal component analysis as given in Algorithm 2.

Algorithm 2 EllipsoidProxy($\mathcal{S}$)

1. Construct the complete point cloud by computing the 3D points for every grid point (u, v) of all $\mathcal{D}_i$.

2. Find the principal components of the point cloud.
3. Apply Whitening transform $\boldsymbol{W}$ to convert to a sphere.
4. Find optimal diameter of the sphere. For example, using least-squared distances from the sphere.
5. Apply $\boldsymbol{W}^{-1}$ to the sphere to get an oriented ellipsoid. Use it as proxy.

Volumetric merging techniques can be used to merge the depth maps into a polygon model [5]. A polygonal model of the surface can be generated from the volumetric model using the Marching cubes algorithm [17]. Different approximations of the model can be obtained by controlling the resolution of the voxel space. Alternately, a high resolution model can be constructed using a fine voxel space and reduced using standard mesh simplification techniques [11] to a proxy of suitable detail. Algorithm 3 outlines the procedure to generate a polygon proxy from the collection of depth images.

Algorithm 3 PolygonalProxy($\mathcal{S}$, r)
1. Merge each depth map to a volumetric space of resolution r using Curless and Levoy's algorithm [5].
2. Create a polygonal model using the Marching cubes algorithm in the voxel space.
3. Decimate the polygon mesh using a mesh simplification algorithm to the desired size.

3.3 Compression of residue images

The residue images are defined over the same pixel grids of the original depth maps. The residue values will be small and correlated if the proxy used approximates the underlying geometry well. The residue images should compress better as a result. Standard image compression techniques like JPEG or JPEG2000 can be used to compress the residue images. They may smooth discontinuities or edges in residue images, resulting in a distortion of the geometric structure. Lossy image compression techniques provide a range of compression ratios by varying the amount of loss that can be tolerated. Lossless compression of the residue images, such as an LZW-based algorithm can reduce the residue data considerably if the residues are mostly correlated, small values.

3.4 Progressive representation and compression

If the residue values are small, the proxy is a good approximation of the scene structure. As more of the residues are added, the scene structure improves. When all residues are added, the input depth map is exactly recovered. We can get a progressively finer representation using more and more of the residue values. Progressive representations can use a range of compression levels; addition of relatively small amounts of data to such a representation can improve the model quality in a near-continuous fashion.

Bit-plane ordering. Progressive decompression of residue values can be achieved by separating the bits of the residues and introducing them one bit at a time, starting with the most significant bit. Each bit brings the depth map value closer to the original value. Each bit can be separated into a binary image and represented separately. An LZW compression for these bit-images effect a lossless entropy compression. The base depth map is obtained by mapping the proxy model to the sampling grid of the depth image. No residue is used in this case. Successively more accurate versions of the depth map can be obtained by adding the bits of the residue values.

3.5 Evaluation of compression

Proxy-based representation can be evaluated using two measures: the compression ratio and the compression quality. The compression ratio is measured by comparing the size of representing the depth maps $\boldsymbol{D}_i$ of the original collection and the size of representing the proxy model and the residue maps $\boldsymbol{R}_i$. Progressive representations do not yield high compression ratios ordinarily but can provide continuously varying detail to the model.

The quality of the representation can be compared using the peak signal to noise ratio (PSNR) obtained for new view generation, since the depth images were meant for IBR. The decompressed depth map is used for generating new views from a few viewpoints. The rendering algorithm with blending described in [23] was used to render all views. The generated image is compared with the corresponding view generated using the original depth images. The PSNR is calculated in dB as

$$PSNR = 10 \log_{10} \frac{\left(2^{\text{bitlength}} - 1\right)^2}{\frac{1}{N} \sum_{u,v} [f(u,v) - g(u,v)]^2} ,$$

where $f(u,v)$ is the image generated using original depth map, $g(u,v)$ is the image generated using the compressed depth maps, N is the number of pixels in $f(u,v)$, and bitlength is the number of bits in the original image.

4 Results

In this section, we demonstrate the effect of proxy-based compression of depth maps on a number of models. Compression ratio and the corresponding PSNR are shown for a number of situations. For the experiments reported in this section, 10 depth images of the male model were used.

Table 1. The compression ratio and PSNR for different quality factors of JPEG compression of residue images. Results for the male model using a polygonal proxy, with 294 faces

Quality Factor	Compression Ratio	PSNR (dB)
10	20.1	23.6
20	19.2	24.9
30	18.2	25.7
40	17.4	26.6
50	16.5	26.9
60	15.7	27.1
70	14.4	28.1
80	12.6	29.3
90	9.7	30.3
100	4.1	32.4

These were distributed uniformly in space. The depth and residues were represented using 16-bit numbers.

4.1 Polygonal proxy

4.1.1 Lossy compression

For this experiment, a male model is compressed using a polygonal proxy with 294 faces, which is computed using a standard mesh simplification algorithm. The residue images were compressed using standard JPEG algorithm using different quality factors. The residue values were rescaled to the range [0 . . . 255] to facilitate easy use of JPEG. The uncompressed size is the number of bytes to represent 10 original depth maps each compressed using `gzip`. The compressed size is the sum of the proxy model and 10 JPEG compressed residue images. The PSNR was computed for a random new viewpoint from a rendering using the original depth maps and another using the compressed and decompressed depth maps.

Table 1 shows the results of applying JPEG compression to the male model. The PSNR values are low indicating poor quality for most compression ranges. This is due to the distortion brought by scaling the residue values and the error introduced by JPEG algorithm. Figure 3 shows the novel views rendered with various qualities of JPEG for the data shown in the Table 1.

4.1.2 Progressive representation

For this experiment, we encode the residue values one bit at a time. A range of decompressed depth maps are created by adding one bit at a time to the projection of the proxy model. The uncompressed size is the number of bytes to represent 10 original depth maps each compressed using `gzip` as before. The compressed size includes the size of the proxy and the `gzip` compressed versions of all bit-planes used. The PSNR compares the views rendered using the uncompressed depth map and the progressively decompressed depth map.

Fig. 3. [left – right, top – bottom] Novel views of the scene rendered with different quality factors [10, 20, 70, and 100] of JPEG compression. Edginess around the face due to JPEG compression decreases with increase in the quality factor

Table 2. Compression ratio and PSNR for progressive addition of bits from residues to the base depth map computed from the proxy. Addition starts with the most significant bit. Results for the male model using polygonal proxy, with 294 faces

Number of bits	Compression Ratio	PSNR (dB)
0	113.2	26.6
1	35.8	26.6
2	21.2	26.6
3	14.8	26.6
4	10.5	27.0
5	6.9	27.6
6	4.4	28.7
7	2.8	30.0
8	1.9	31.9
9	1.4	33.5
10	1.1	34.7
11	0.9	36.4
12	0.8	38.6

The progressive improvement in the quality and compression ratios are shown for male model using polygonal proxies of different sizes.

Table 2 tabulates the results of introducing the 12 bits of the residue values to the polygonal proxy with 294 faces in the significance order. The results show that even a modest polygonal proxy can serve as a good approximation if local texture and blending are used. A few bits of residue bring the rendering quality to acceptable levels. Figure 4 shows the novel views rendered

Fig. 4. [left – right, top – bottom] Novel views of the scene rendered when 0, 4, 7, and 12 bits from residues are added to the base depth map computed from polygonal proxy with 294 faces. Addition starts with the most significant bit. Rendering quality improves as the number of bits increases

Fig. 5. [left – right, top – bottom] Novel views of the scene rendered when 0, 3, 6, and 12 bits from residues are added to the base depth map computed from polygonal proxy with 36 faces. Addition starts with the most significant bit. The neck region shows quick improvements with addition of more bits

Table 3. Compression ratio and PSNR for progressive addition of bits from residues to the base depth map computed from the proxy. Addition starts with the most significant bit. Results for the male model using polygonal proxies, with (i) 36 faces and (ii) 2366 faces

	36 Faces		2366 Faces	
# Bits	CR	PSNR (dB)	CR	PSNR (dB)
0	871.9	25.5	12.9	28.1
3	12.8	25.6	7.4	28.1
6	3.2	29.1	4.5	29.0
9	1.4	34.1	1.8	33.0
12	0.9	38.7	0.9	39.2

Fig. 6. [left – right, top – bottom] Novel views of the scene rendered when 0, 2, 3, 4, 5, and 12 bits from residues are added to the base depth map computed from ellipsoid proxy. Addition starts with the most significant bit. Ears are shown multiple times when 0 and 2 bits are used, as ellipsoid proxy is not very accurate representation of the model. Improvement is shown when more bits are added

with bits added from residues for the data shown in the Table 2.

Table 3 tabulates the results of introducing the 12 bits of the residue values to the polygonal proxies with 36 faces and 2366 faces in the steps of 3 bits added. Figure 5 shows the novel views rendered when polygon with 36 faces used as a proxy.

As we can see from the Tables 2 and 3, using the polygonal proxy with large number of faces (a high detailed one) the novel views rendered using proxy alone result in a very good quality images. In the case of polygonal proxy with size 2366 faces achieves a 28 dB using proxy alone, where as it is 25.5 dB in the case of polygonal proxy with 36 faces. At a compression ratio of 4, all the polygonal proxies achieves approximately a constant quality of novel views rendered around 28 dB.

4.2 Ellipsoid proxy

In this section we use an ellipsoid proxy computed using the Algorithm 2. Only a few numbers are required to store the direction cosines and lengths of major, minor and subminor axes of ellipsoid. These numbers will occupy a size of 60 bytes.

Table 4 tabulates the results of introducing the 12 bits of the residue values to the proxy in the significance order. The results show that a simple parametric proxy like el-

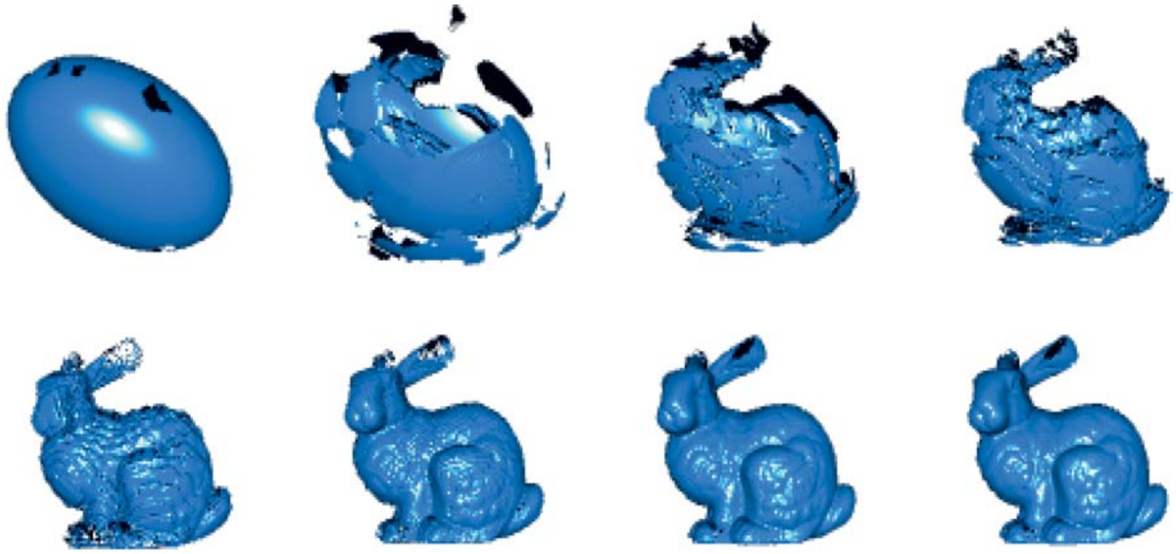

Fig. 7. [left – right, top – bottom] Models constructed when 0, 2, 3, 4, 5, 6, 7, and 8 bits are used to represent residues computed using ellipsoid proxy for the bunny model

Table 4. Compression ratio and PSNR for progressive addition of bits from residues to the base depth map computed from the proxy. Addition starts with the most significant bit. Results for the male model using ellipsoid proxy

Number of bits	Compression Ratio	PSNR (dB)
0	5071.8	16.1
1	51.4	16.1
2	19.1	16.4
3	10.0	18.0
4	6.1	19.4
5	3.9	20.7
6	2.6	22.3
7	1.8	23.4
8	1.4	24.6
9	1.1	25.2
10	0.9	25.7
11	0.8	25.8
12	0.7	25.9

Table 5. The compression ratio when different number of bits are used to represent the residues for the bunny model using ellipsoid proxy

Number of bits	Compression Ratio
0	7220.0
1	72.3
2	14.8
3	10.4
4	7.4
5	5.2
6	3.5
7	2.4
8	1.6

lipsoid can serve as a good approximation. A few bits of residue bring the rendering quality to close to acceptable levels (may not be as good as a triangle proxy). Figure 6 shows the novel views rendered with bits added from residues for the data shown in the Table 4.

We looked at how the overall shape changes with resolution of the residue representation. To explore this, we represent residues using 2, 3, ... 8 bits, decompress and reconstruct the depth map using the proxy. These decompressed depth maps are merged into a shape using the vrip package [5]. A Stanford bunny model, with an ellipsoid proxy is used in this experiment. The evolution of the shape is shown in the Fig. 7. Table 5 shows the compression ratios when different number of bits are used to represent the residues.

5 Conclusions

In this paper we have described a novel method for compressing multiple depth maps using geometry proxy. We showed good results on models using different proxies. We also described a progressive representation, where we encode the residue values one bit at a time. The progressive improvement in the novel views rendered when these bits are added is demonstrated.

Acknowledgement We thank cyberware for providing the male model which is used in experiments.

References

1. Adelson, E.H., Bergen, J.R.: The plenoptic function and the elements of early vision. In: Computational Models of Visual Processing. MIT Press, Cambridge, MA (1991)
2. Baker, H., Tanguay, D., Sobel, I., Dan Gelb, M.E.G., Culbertson, W.B., Malzbender, T.: The Coliseum Immersive Teleconferencing System. In: International Workshop on Immersive Telepresence (ITP2002) (2002)
3. Buehler, C., Bosse, M., McMillan, L., Gortler, S.J., Cohen, M.F.: Unstructured lumigraph rendering. In: SIGGRAPH (2001)
4. Chen, S., Williams, L.: View interpolation for image synthesis. In: SIGGRAPH (1993)
5. Curless, B., Levoy, M.: A volumetric method for building complex models from range images. In: SIGGRAPH (1996)
6. Debevec, P.E., Taylor, C.J., Malik, J.: Modeling and rendering architecture from photographs: a hybrid geometry and image-based approach. In: SIGGRAPH (1996)
7. Girod, B., Chang, C.L., Ramanathan, P., Zhu, X.: Light field compression using disparity-compensated lifting. In: ICASSP (2003)
8. Gortler, S.J., Grzeszczuk, R., Szeliski, R., Cohen, M.F.: The lumigraph. In: SIGGRAPH (1996)
9. Gortler, S.J., He, L.W., Cohen, M.F.: Rendering layered depth images. Tech. Rep. MSTR-TR-97-09, Microsoft Research (1997).
10. Gotz, D., Mayer-Patel, K., Manocha, D.: Irw: an incremental representation for image-based walkthroughs. In: MULTIMEDIA '02: Proceedings of the 10th ACM International Conference on Multimedia, pp. 67–76. ACM Press, New York (2002)
11. Heckbert, P.S., Garland, M.: Survey of polygonal surface simplification algorithms. Tech. Rep. CMU-CS-95-194 (1997)
12. Ihm, I., Park, S., Lee, R.K.: Rendering of spherical light fields. In: Pacific Graphics (1997)
13. Yu, J., McMillian, L., Gortler, S.: Scam light field rendering. In: Pacific Graphics (2002)
14. Kanade, T., Rander, P.W., Narayanan, P.J.: Virtualized reality: constructing virtual worlds from real scenes. IEEE Multimedia **4**(1), 34–47 (1997)
15. Krishnamurthy, R., Chai, B.B., Tao, H., Sethuraman, S.: Compression and transmission of depth maps for image-based rendering. In: International Conference on Image Processing (2001)
16. Levoy, M., Hanrahan, P.: Light field rendering. In: SIGGRAPH (1996)
17. Lorensen, W.E., Cline, H.E.: Marching cubes: a high Resolution 3D surface construction algorithm. In: SIGGRAPH (1987)
18. Magnor, M., Eisert, P., Girod, B.: Multi-view image coding with depth maps and 3D geometry for prediction. In: Proceedings SPIE Visual Communication and Image Processing (VCIP-2001), San Jose, CA, 2001, pp. 263–271
19. Mark, W.R., McMillan, L., Bishop, G.: Post-rendering 3D warping. In: SI3D '97: Proceedings of the 1997 symposium on Interactive 3D Graphics, pp. 7–16. ACM Press, New York (1997)
20. Matusik, W., Buehler, C., Raskar, R., McMillan, L., Gortler, S.: Image-based visual hulls. In: Proceedings of ACM SIGGRAPH, 2000, pp. 369–374
21. McMillan, L., Bishop, G.: Plenoptic modeling: an image-based rendering algorithm. In: SIGGRAPH (1995)
22. Narayanan, P.J., Rander, P.W., Kanade, T.: Constructing virtual worlds using dense stereo. In: Proceedings of the International Conference on Computer Vision (1998)
23. Narayanan, P.J., Penta, S.K., Sireesh Reddy, K.: Depth+texture representation for image-based rendering. In: ICVGIP (2004)
24. Seitz, S.M., Dyer, C.R.: View morphing. In: SIGGRAPH (1996)
25. Shum, H.Y., He, L.W.: Rendering with concentric mosaics. In: SIGGRAPH (1999)
26. Shum, H.Y., Kang, S.B., Chan, S.C.: Survey of image-based representations and compression techniques. IEEE Trans. Circuits Syst. Video Technol. 13(11), 1020–1037 (2003)
27. Shum, H.Y., Sun, J., Yamazaki, S., Li, Y., Tang, C.K.: Pop-up light field: an interactive image-based modeling and rendering system. ACM SIGGRAPH **23**(2), 143–162 (2004)
28. Tong, X., Gray, R.M.: Coding of multi-view images for immersive viewing. In: ICASSP, Istanbul, Turkey, 9 June 2000
29. Towles, H., Chen, W.C., Yang, R., Kam, S.U., Fuchs, H.: 3D Tele-Collaboration over internet 2. In: International Workshop on Immersive Telepresence, Juan Les Pins, France, 6 December 2002
30. Wilson, A., Mayer-Patel, K., Manocha, D.: Spatially-encoded far-field representations for interactive walkthroughs. In: Proceedings of the 9th ACM International Conference on Multimedia, Ottawa, Canada, 2001, pp. 348–357
31. Zitnick, C.L., Kang, S.B., Uyttendaele, M., Winder, S., Szeliski, R.: High-quality video view interpolation using a layered representation. ACM Trans. Graph. **23**(3), 600-608 (2004)

SASHI KUMAR PENTA received a B.S. (Honors) and an M.S. in Computer Science from the International Institute of Information Technology (IIIT), Hyderabad in 2003 and 2005, respectively. He is a member of the Centre for Visual Information Technology (CVIT), IIIT since 2001. His current research interests include computer graphics, computer vision, and machine learning.

P.J. NARAYANAN is an Associate professor at IIIT, Hyderabad and heads CVIT. He received a B.S from IIT, Kharagpur, in 1984 and a Ph.D. in Computer Science from University of Maryland in 1992. He head the Computer Vision and Virtual Reality group, Centre for Artificial Intelligence and Robotics (CAIR), Bangalore, from 1996 to 2000. He was a faculty member at the Robotics Institute of Carnegie Mellon from 1992 to 1996. His current research interests include computer graphics, computer vision, machine learning, and virtual reality.