Preprocessing of Depth and Color Information for Layered Depth Image Coding

Seung-Uk Yoon, Sung-Yeol Kim, and Yo-Sung Ho

Gwangju Institute of Science and Technology (GIST)
1 Oryong-dong, Buk-gu, Gwangju, 500-712, Korea
{suyoon,sykim75,hoyo}@gist.ac.kr

Abstract. The layered depth image (LDI) is a popular approach to represent three-dimensional objects with complex geometry for image-based rendering (IBR). LDI contains several attribute values together with multiple layers at each pixel location. In this paper, we propose an efficient preprocessing algorithm to compress depth and color information of LDI. Considering each depth value as a point in the two-dimensional space, we compute the minimum distance between a straight line passing through the previous two values and the current depth value. Finally, the current attribute value is replaced by the minimum distance. The proposed algorithm reduces the variance of the depth information; therefore, it improves the transform and coding efficiency.

Keywords: Layered depth image, coding, image-based rendering

1 Introduction

Since there have been researches on geometry-based rendering methods, lots of useful modeling and rendering techniques have been developed. However, geometry-based rendering requires elaborate modeling and long processing time. As an attractive alternative to overcome these problems, image-based rendering (IBR) techniques have received much attention. They use two-dimensional (2-D) images as primitives to generate an arbitrary view of the three-dimensional (3-D) scene. IBR techniques require proper computational resources and do not bother from the complexity of 3-D objects in the scene. In addition, it is much easier to acquire a photo or a picture than complex 3-D models of the scene. In spite of these benefits, the amount of data generated from IBR is very huge. Therefore, coding of IBR data is one of the main requirements of IBR techniques.

Various IBR techniques can be classified into three categories based on how much geometry information is used [1], [2]: rendering with no geometry; rendering with implicit geometry; and rendering with explicit geometry. Among the variety of methods, a layered depth image (LDI) [3] is one of the efficient rendering methods for 3-D objects with complex geometries. LDI is contained in rendering with explicit geometry. It represents the current scene using an array of pixels viewed from a single camera location. However, LDI pixel contains not

K. Aizawa, Y. Nakamura, and S. Satoh (Eds.): PCM 2004, LNCS 3333, pp. 622–629, 2004.

just color values, but also several other attribute values. It consists of color information, depth between the camera and the pixel, and other attributes that support rendering of LDI. Three key characteristics of LDI are: (1)it contains multiple layers at each pixel location, (2)the distribution of pixels in the back layer is sparse, and (3)each pixel has multiple attribute values. Because of these special features, LDI enables us to render of arbitrary views of the scene at new camera positions. Moreover, the rendering operation can be performed quickly with the list-priority algorithm proposed by McMillan [4].

Despite of these benefits, a high resolution LDI contains a huge amount of data [5]. Fig. 1 shows an example. The Cathedral scene occupies 14.1 megabytes (MB), and Stream contains 10.86 MB of data. This means that a single LDI contains a large amount of data, unlike normal 2-D images. If we want to render or represent complex natural scenes with LDI, it requires even higher amount of data. Therefore, it is necessary to compress the LDI data efficiently for the real-time rendering and transmission under a limited bandwidth.

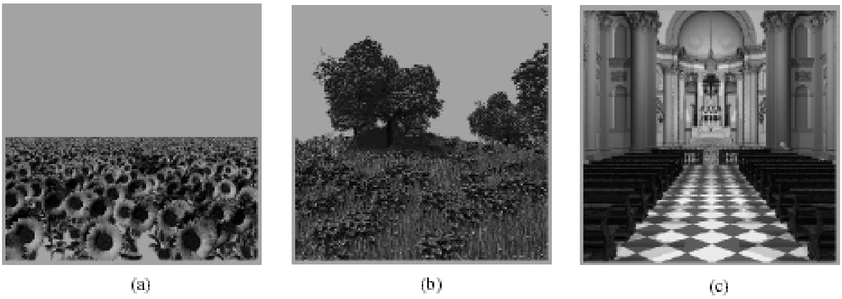

(a) (b) (c)

Fig. 1. LDI dataset: (a) Sunflowers, (b) Stream, and (c) Cathedral.

As mentioned in the previous work [5], generic lossless coding tools, such as WinZip, cannot provide the high compression ratio. On the other hand, lossy coding tools, like JPEG-2000 and MPEG-4, guarantee higher coding efficiency, but they cannot be applied directly to compress LDI. Since the density of LDI pixels becomes lower in the back layer, a new algorithm is required to code the LDI data. In order to effectively deal with these features of LDI, a kind of divide and conquer methodologies was proposed by J. Duan *et al.* [5]. In their work, they divide the LDI data into eight components and compress each component image with different techniques. Although their approach provides a high compression ratio with moderate image quality, it does not consider coherency within the component image.

In this paper, we propose a new preprocessing algorithm to improve the transform efficiency. We separate LDI into several component images similarly to the previous work, but our algorithm exploits coherency among pixels within each component image. Considering each pixel as a point in the 2-D space, we compute the minimum distance between a straight line passing through the previous two values and the current depth value. Finally, the current attribute value is replaced by the minimum distance. The proposed algorithm reduces the variance of depth and color information; therefore, it improves the transform and coding efficiency.

The paper is organized as follows. The data structure and previous coding methods of LDI are briefly reviewed in Section 2 and Section 3, respectively. In Section 4, we explain details of our preprocessing algorithm. After experimental results are presented in Section 5, we draw conclusions in Section 6.

2 Layered Depth Image (LDI)

LDI pixels contain depth values along with their colors. In addition, LDI contains potentially multiple depth pixels per pixel location. The farther depth pixels, which are occluded from the LDI center, will act to fill in the disocclusions that occur as the viewpoint moves away from the center. Fig. 2 shows the generation process of LDI. The LDI scene viewed from C_1 is constructed by warping pixels in other camera locations, such as C_2 and C_3.

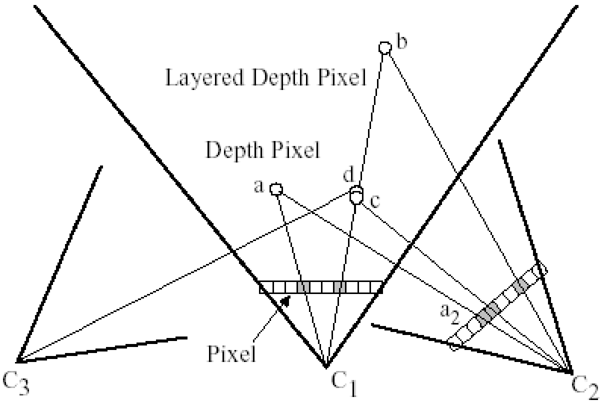
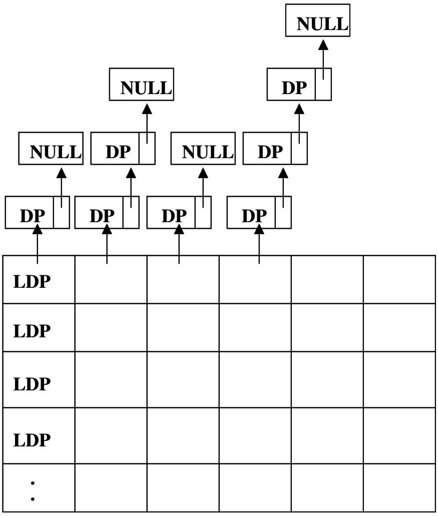


Fig. 2. The generation of the layered depth image.

Unlike the ordinary image consisting of the luminance and chrominance values, each LDI pixel contains 63 bit information [5]: 8 bits each for the R, G and B components, 8 bits for the alpha channel, 20 bits for the depth of the object, and 11 bits for the index into a splat table. The splat table is in turn divided into 5 bits for the distance, 3 bits for the x norm, and 3 bits for the y norm. It is used to support various pixel sizes in rendering of LDI. The overall data structure of the single LDI is shown in Fig. 3.



```
DepthPixel =
{ ColorRGBA: 32 bit integer
  Z: 20 bit integer
  SplatIndex: 11 bit integer
}

LayeredDepthPixel =
{ NumLayers : integer
  Layers[0..numlayers-1]:
  array of DepthPixel
}

LayeredDepthImage =
{ Camera: camera
  Pixels[0..xres-1, 0..yres-1]:
  array of Layered DepthPixel
}
```

Fig. 3. The structure of LDI.

3 Coding of LDI Data

In the previous work [5], they investigate the compression of the sparse and nonrectangular supported data of LDI. They first record the number of layers (NOL) at each pixel location. The LDI data is then reorganized into a more suitable layout by dividing LDI into layers, each of which contains a mask indicating the existence of pixel in the layer. Each LDI layer is then separated into individual components, such as Y, Cr, Cb, alpha, and depth. Fig. 4 shows eight components of LDI.

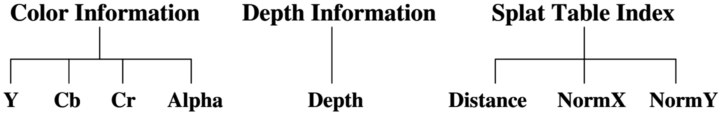


Fig. 4. Layered depth image: component separation.

The component images of each layer are compressed separately. They aggregate the data on the same layer so that the data is more compactly distributed. An arbitrary shape wavelet transform and coding algorithm is used to compress the aggregated data. Finally, the compressed bitstreams of the different layers and components are concatenated to form the compressed LDI bitstream. A practical rate-distortion model is used to optimally allocate bits among all the components.

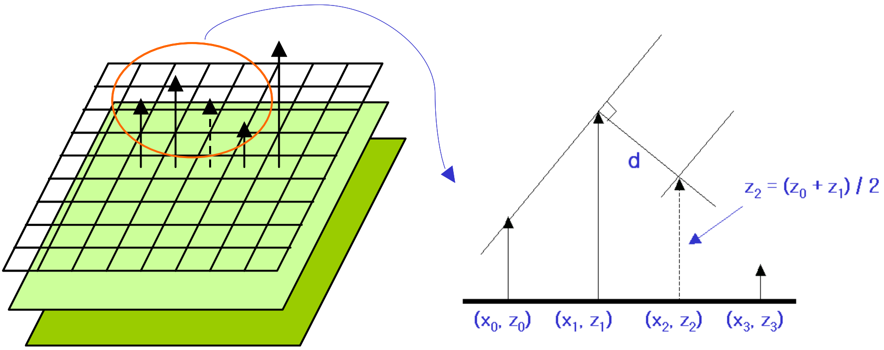


Fig. 5. Calculation of the minimum distance.

4 Preprocessing for LDI Coding

Because of the special data structure of LDI, existing still image compression methods, such as JPEG, cannot be applied directly or are not very efficient. There are three key characteristics of the LDI data. It contains multiple layers at each pixel location; the distribution of pixels in the back layer is sparse; and each pixel has multiple attribute values, including color, depth, and splat table index. In the previous work [5], data aggregation is performed to use these key features of LDI. After aggregating the LDI data, an arbitrary shape wavelet transform and coding algorithm is applied.

In this paper, we propose a new preprocessing algorithm to improve the efficiency of the wavelet transform, which directly affects on the compression ratio of each component image. Thus, our algorithm is performed prior to the wavelet transform. The depth and color information is processed in the same way. Since we observe (x, z) values are changing for the fixed Y-axis, we can consider the one-dimensional (1-D) depth value as the 2-D point. Along the increasing direction of the X-axis, we draw a line passing through two points, and then calculate the Euclidean distance between the line and the current depth value, as illustrated in Fig. 5.

In Fig. 5, the left planes shows the spatial relationship among layers of LDI. The proposed preprocessing method uses correlation among attribute values in the same layer for each component image. In the right figure, the dotted arrow represents the padded depth value at the empty pixel location. We insert the average value of the previous two points into the vacant position. After calculating the minimum distance, we replace the current depth value by the minimum distance. Finally, the inserted average values are removed before the data aggregation. The distance between the line, passing through $A(x_0, z_0)$ and $B(x_1, z_1)$, and the point $C(x_2, z_2)$ is computed by

$$d = \frac{\left|(A - B)^{\perp} \cdot (C - A)\right|}{|A - B|}, \tag{1}$$

where $A^{\perp}$ is the counterclockwise perpendicular vector to the given A; it means that $(x_0, y_0)^{\perp}$ is $(-y_0, x_0)$.

These procedures are similar to the differential coding method. Instead of calculating the direct difference between two values, we use the distance from the line through the previous two points. Since the direct difference becomes greater in the back layer, the differential methodology is not properly applied to the data structure of LDI. We compare the standard deviation using the differential scheme with the proposed method in our experimental results.

Since each pixel contains the depth and color information at the same location, we can easily compute the minimum distance for color values of Y, Cr, and Cb components; hence, Eq. 1 can be directly reused. In our algorithm, the Euclidean distance is used as the measure for representing the coherency among neighboring depth and color values. This is reasonable because the Euclidean distance is one of the widely used similarity measures for normal 2-D images in general. However, it cannot be applied directly to other attribute values, such as the distance or norms of the splat table index, because their correlations cannot be measured by the Euclidean distance.

5 Experimental Results and Analysis

Efficiency of the proposed preprocessing algorithm is demonstrated with the following experiments. Fig. 6 shows the test data set of LDI scenes. The resolution of Ball LDI is 246 x 246 and that of Flower is 690 x 476. Three layers are used in our experiment for each LDI.

(a) (b)

Fig. 6. Test LDI data set: (a) Ball, (b) Flower.

We calculate the standard deviations for test LDIs to evaluate the performance of the proposed algorithm before and after the preprocessing. As shown in Table 1, the standard deviation of each LDI decreases over 45% after applying the preprocessing algorithm. Especially, Table 1 shows that the standard deviation is reduced much more for Flower LDI. It means that the more pixels, the more reduction occurs because the replaced minimum distance lowers differences among depth values.

Table 1. Standard deviations of depth information

	Ball			Flower		
	Before preprocessing	After preprocessing	Reduction rates	Before preprocessing	After preprocessing	Reduction rates
Layer 1	51.98	27.06	47.94 %	1815.07	364.46	79.92 %
Layer 2	112.80	46.28	58.97 %	3076.95	613.61	80.06 %
Layer 3	154.26	65.66	57.43 %	3740.26	756.03	79.79 %

Table 2 shows the amount of depth information after the wavelet transform and variable length coding. The data size is decreased over 20% because the distribution of depth values is skewed.

Table 2. Amount of depth information of test LDIs [kBytes]

Ball			Flower		
Before preprocessing	After preprocessing	Reduction rates	Before preprocessing	After preprocessing	Reduction rates
96.00	75.70	21.15 %	510.25	405.33	20.56 %

Finally, we compare our algorithm with the differential coding method in terms of the standard deviation. Table 3 shows that the proposed scheme provides higher reduction ratio, because direct differences among depth values become greater than the minimum distance, especially in the back layer. Therefore, the proposed preprocessing algorithm further reduces the variance of depth and color information of LDI.

Table 3. Comparison between the differential technique and the proposed algorithm

	Ball			Flower		
	Original	Differential	Proposed	Original	Differential	Proposed
Layer 1	51.98	21.05	27.06	1815.07	399.82	364.46
Layer 2	112.30	47.24	46.28	3076.95	924.50	613.61
Layer 3	154.26	74.57	65.66	3740.26	1459.07	756.03

6 Conclusions

In this paper, we propose an efficient preprocessing algorithm to code depth and color information of layered depth images. We consider each depth value as a 2-D point. After the minimum Euclidean distance between a line and the current point is calculated, the current depth value is replaced by the minimum distance. Since the previous approach does not consider coherency among neighboring pixels, we focus on using correlations of depth and color values within each component image. Experimental results demonstrate that the proposed algorithm reduces the variance of depth and color information. Therefore, the transform efficiency was improved and the amount of data was reduced.

Acknowledgements. This work was supported in part by Gwangju Institute of Science and Technology (GIST), in part by the Ministry of Information and Communication (MIC) through the Realistic Broadcasting Research Center (RBRC) at GIST, and in part by the Ministry of Education (MOE) through the Brain Korea 21 (BK21) project.

References

1. Shum, H., Kang, S.: A Review of Image-based Rendering Techniques. IEEE/SPIE Visual Communications and Image Processing (VCIP), June (2000) 2–13
2. Shum, H., Kang, S., Chan, S.: Survey of Image-Based Representations and Compression Techniques. IEEE Transaction on Circuit and Systems for Video Technology, Vol. 13, No. 11, November (2003) 1020–1037
3. Shade, J., Gotler, S., Szeliski, R.: Layered Depth Images. ACM SIGGRAPH, July (1998) 291–298
4. McMillan, L.: A List-Priority Rendering Algorithm for Redisplaying Projected Surfaces. University of North Carolina, 95–005 (1995)
5. Duan, J., Li, J.: Compression of the LDI. IEEE Transaction on Image Processing, Vol. 12, No. 3, March (2003) 365–372