CODING DEPTH IMAGES WITH PIECEWISE LINEAR FUNCTIONS FOR MULTI-VIEW SYNTHESIS

Yannick Morvan[1]*, Dirk Farin*[1] *and Peter H. N. de With*[1,2]

[1] Univ. of Technol. Eindhoven, P.O. Box 513
5600 MB Eindhoven, Netherlands
Email:{y.morvan,d.s.farin}@tue.nl

[2] LogicaCMG, P.O. Box 7089
5605 JB Eindhoven, Netherlands
Email:P.H.N.de.With@tue.nl

ABSTRACT

An efficient way to transmit multi-view images is to send the texture image together with a corresponding depth-map. The depth-map specifies the distance between each pixel and the camera. With this information, arbitrary views can be generated at the decoder. This technique requires a compression technique for the depth-maps. Ordinary image compression algorithms like JPEG provide low quality, since the ringing artifacts along edges generate clouds of pixels in 3D space. For this reason, we propose a new algorithm for the coding of depth maps that uses piecewise linear functions to approximate the depth information. This algorithm shows no degradation along discontinuities and it provides a high compression factor with bit-rates as low as 0.05 bit/pixel.

1. INTRODUCTION

The upcoming 3-D display technology allows to present images with the illusion of depth. Such display technology is based on showing several views of the same scene, viewed from different positions at the same time. An independent transmission of these views has several disadvantages. First, it is inefficient since there is a strong correlation between pairs of images. Second, different displays will support a different number of views, which makes it impractical to prepare the displayed views at the encoder. Instead, the views should be synthesized at the decoder that knows the connected display characteristics. These disadvantages can be eliminated by transmitting the texture data independent from the geometry data. One approach is to send the texture data for the central view and a depth-map [1] that specifies the depth of each pixel in the texture image. This depth map can be represented by a gray-scale image where dark and bright pixels correspond to far and near pixels, respectively. Based on this depth-map, arbitrary views can be synthesized at the decoder.

Previous work on depth image coding has used block-based and wireframe modelling techniques [2]. The former method codes the depth image using a modified MPEG coder, while the latter technique divides the image into triangular patches. In each patch, the image is approximated by a linear function. If the data cannot be represented with a single linear function, smaller patches are used for that area. However, the placement of the patches is not adapted to the image content, such that a large number of small patches are generated along edges.

Using a transform coder for depth-maps imposes two difficulties. First, the coding is inefficient, since depth-maps usually comprise large smooth areas with little detail. Furthermore, transform coders generate ringing artifacts along edges that lead to errors in the pixels' depth, which can be perceived as a noisy cloud of pixels along the object borders.

We have observed that large parts of typical depth-maps usually show walls and other approximately planar surfaces. As a result, the input contains both sharp edges and regions of linear depth changes. This has brought us to the concept of modelling the signal with piecewise linear functions $f(x,y) = ax + by + c$, where a, b, c are parameters which are adjusted to the image content. The image is subdivided using a quadtree decomposition and an independent model is used for each node. Furthermore, we provide a mode that describes the data of a node with two linear models which are separated by a straight line. This last mode enables to code discontinuities without introducing many small nodes along edges. Our results show that, prior to entropy coding, our proposal gives a bit rate as low as 0.05 bit/pixel for synthetic and 0.19 bit/pixel for natural images.

The sequel of this paper is structured as follows. Section 2 briefly illustrates the generation of arbitrary views. Section 3 describes the framework of our depth-map coding algorithm, while Section 4 provides further details about the data approximation for each coding mode. In Section 5, we describe a rate-distortion optimization of the quadtree decomposition. Results are presented in Section 6 and the paper concludes with Section 7.

2. ARBITRARY VIEW SYNTHESIS

A single texture image and a corresponding depth-map $D(x,y)$ which indicates the distance of each pixel to the camera, are sufficient to synthesize views from arbitrary camera positions. To synthesize a new view, we first convert the texture image into a set of 3-D points. We assume that the focal-length of the recording camera was a fixed parameter F and the principal point of the image is at (o_x, o_y). Furthermore, we assume that the depth-map modulates the distance between the pixels and the camera along the viewing ray. This lets us calculate the *homogeneous* 3-D position $\boldsymbol{p}$ of each pixel (x,y) as

$$\boldsymbol{p} = ((x-o_x)z/F),\ (y-o_y)z/F,\ z,\ 1)^\top,$$

where $z = F + D(x,y)$ is the distance of the pixel to the recording camera.

To generate a *new* camera view for synthesis from a camera position $\boldsymbol{t}$, a 3×3 camera rotation matrix $\boldsymbol{R}$, and a *new* corresponding focal-length f, we get the projection of each pixel as

$$\boldsymbol{p'} = \begin{pmatrix} f & 0 & o_x \\ 0 & f & o_y \\ 0 & 0 & 1 \end{pmatrix} (\boldsymbol{R}|\boldsymbol{Rt})\, \boldsymbol{p}.$$

(a) (b) (c) (d) (e)

Figure 1: Various views of a scene. All views were synthesized using only a single texture image and a corresponding depth-map.

For the case that $\boldsymbol{t} = \boldsymbol{0}$, $f = F$, and $\boldsymbol{R}$ is equal to the identity matrix, we obtain exactly the input image. Example views that were synthesized from a single texture image and the depth information are depicted in Figure 1.

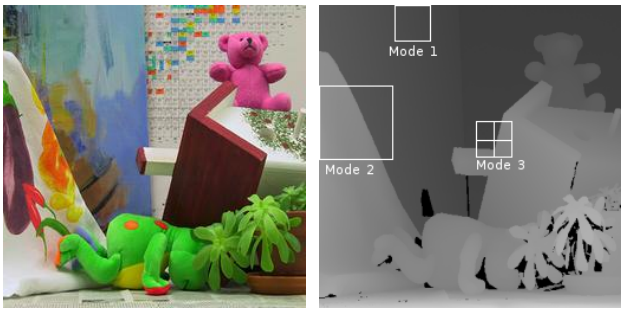
3. DEPTH-MAP CODING ALGORITHM

In this section, we present a novel approach for depth coding using the piecewise linear functions mentioned in Section 1. With a single linear function, we can represent one planar surface of the scene, like the ground plane, walls, or small objects. Hence, a single function can cover areas of variable size in the image. To identify the location of these planar surfaces in the image, we employ a quadtree decomposition which recursively divides the image into rectangles, i.e. nodes of different size. In some cases, the depth-map within one rectangle can be approximated with a single linear approximation. If no suitable approximation can be determined for the rectangle, it is subdivided into four smaller rectangles. Additionally to this standard quadtree subdivision, we apply a special coding mode if there are discontinuities in the depth map. To prevent that many small rectangles are required along a discontinuity, we separate the rectangle along a straight line into two regions. Each of these two regions is coded with a separate linear function. Consequently, the coding algorithm chooses between three possible modes for each node in the quadtree:

- *Mode 1*: Approximate the rectangle content with a single linear function;
- *Mode 2*: Subdivide the rectangle along a straight line into two regions and approximate each region with an independent linear function;
- *Mode 3*: Subdivide the rectangle into smaller ones and recursively process these rectangles.

The decision for each coding mode is based on a rate-distortion optimization described in Section 5.

To illustrate our approach, let us examine the natural depth image "Teddy" [3] as an example (see Figure 2). Three rectangles are marked that will be coded with the three described modes. The top rectangle only comprises a wall and can thus be coded with a single model. The left rectangle shows part of a planar object and the background, separated by an edge. Here, we can use *Mode 2* with one model for the first object and another one for the background. The rectangle at the center of the image shows complex content and has to be subdivided. After this subdivision, each resulting node shows one or two regions, so that the node can be coded using either *Mode 1* or *Mode 2*. For an approximation of higher quality, a node can be subdivided into smaller rectangles by employing an additional recursive iteration.



(a) (b)

Figure 2: Example input image "Teddy" (Figure 2(a)) and the corresponding depth image (Figure 2(b)). Three rectangles are marked for which different coding modes are used.

4. PARAMETER ESTIMATION FOR CODING MODES 1 AND 2

This section explains the computations of the parameters that are used in the two coding modes. For *Mode 1*, a single linear function is used for which the three parameters a, b, c must be calculated. For *Mode 2*, the location of the separation line and the two sets of parameters for the two regions should be calculated.

4.1 Parameter estimation for *Mode 1*

In order to determine the three parameters a, b, c of the linear function $f(x,y) = ax + by + c$, a least-squares fitting is used. This optimization minimizes the sum of squared differences between depth data and the proposed linear model. Accordingly, parameters a, b, c are determined so that the error

$$E(a,b,c) = \sum_{x=1}^{n} \sum_{y=1}^{n} (ax + by + c - D(x,y))^2$$

is minimized, where $D(x,y)$ denotes depth at position (x,y) and n denotes the node size (assuming square image nodes for simplicity). This error has a minimum when the gradient satisfies $||\nabla E|| = 0$. When taking the partial derivatives with respect to a, b and c of this equation, we find a set of linear equations specified by

$$\begin{pmatrix} t & u & v \\ u & t & v \\ v & v & n^2 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} \sum_{x=1}^{n} \sum_{y=1}^{n} xD(x,y) \\ \sum_{x=1}^{n} \sum_{y=1}^{n} yD(x,y) \\ \sum_{x=1}^{n} \sum_{y=1}^{n} D(x,y) \end{pmatrix},$$

with

$$t = \frac{n^2(n+1)(2n+1)}{6}, \quad u = \frac{n^2(n+1)^2}{4}, \text{ and } v = \frac{n^2(n+1)}{2}.$$

Since the matrix on the left side of our equation system is constant, it is possible to pre-compute and store the inverse for each size of the node. This enables to compute the parameters a, b, c with a simple matrix multiplication.

4.2 Parameter estimation for *Mode 2*

For this second model, we try to approximate the depth value $D(x,y)$ with two linear functions, $f_1(x,y)$ and $f_2(x,y)$ in the supporting areas S_1 and S_2. Each area is delineated by a subdividing line p_1p_2, defined by points p_1 and p_2 (see Figure 3). To evaluate the model fit, we consider the L_2 distance between the model and the image data. The best set of parameters $\{f_1(x,y), f_2(x,y), p_1, p_2\}$ for *Mode 2* are those parameter values that minimize the approximation error

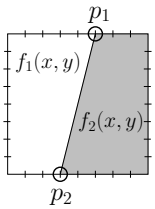


Figure 3: The second model is defined by two linear functions $f_1(x,y)$ and $f_2(x,y)$ separated by a subdividing line p_1p_2.

$$E = \sum_{(x,y)\in S_1} (f_1(x,y) - D(x,y))^2 + \sum_{(x,y)\in S_2} (f_2(x,y) - D(x,y))^2.$$

This optimization problem cannot be solved with a least-squares-minimization because of the following reason. Without knowing the subdivision boundary between the two regions, it is not possible to determine their model parameters. On the other hand, it is not possible to identify the subdivision line as long as the region parameters are unknown. For this reason, the parameter estimation is performed in testing every possible line that divides the rectangle into two areas. This step provides a candidate subdivision line $p_1^c p_2^c$ and two candidate regions S_1^c and S_2^c. Subsequently, parameters of candidate functions $f_1^c(x,y)$ and $f_2^c(x,y)$ are computed using a least-squares-minimization over the candidate regions S_1^c and S_2^c, respectively. At the end of the exhaustive search, the best fitting model is selected. The major advantage of such a technique is that it provides the optimal set of parameters minimizing the L_2 distance between the model and the image data. However, such a full search is computationally expensive. For this reason, we also investigated a fast sub-optimal parameter estimation algorithm (see [4]).

5. RATE DISTORTION ANALYSIS

This section describes the mode decision algorithm that we apply to adjust the coding bit-rate. We will only outline the algorithmic principle and omit detailed discussions about optimality. Furthermore, we have omitted the use of entropy coding, so that the bit-rate calculation is relatively simple.

The guiding principle is the application of a Lagrangian cost function [5] $D + \lambda R$ where D denotes the distortion resulting from the quadtree decomposition in the pixel domain and R stands for the bit-rate required to code the sub-image with the corresponding parameters. For each coding mode, a corresponding cost (distortion and rate) can be defined. If the node is subdivided in smaller rectangles, the cost of this node includes the sum of the children node costs. To make an optimal decision for one node, we select the coding mode that gives the lowest Lagrangian cost. Because the cost depends on the cost of children nodes, an optimal solution can be found by computing the coding modes by a bottom-up traversal of the quadtree. However, this optimal solution is too computationally expensive, since it requires the computation of the completely expanded quadtree. Instead, we propose an approximation that can be computed by a top-down traversal of the tree. The algorithm can be summarized as follows.

- **Step 1:** Compute the Lagrangian cost for coding *Mode 1* and *2*.
- **Step 2:** Compute the Lagrangian costs for the children nodes and select temporarily the modes of the children with the minimum cost.
- **Step 3:** For the current node, choose the best coding mode based on the cost of *Mode 1*, *Mode 2* and the sum of the children costs.

This top-down traversal of the tree is more computationally efficient since we can stop the subdivision as soon as *Mode 1* or *Mode 2* is selected.

6. EXPERIMENTAL RESULTS

For evaluating the performance of the algorithm, experiments were carried out using the synthetic depth images "Corridor" [1] and the natural depth image "Teddy".

Experiments have revealed that the proposed algorithm can code large areas of the image with a single node. Examples are the top center node in Figure 4(c) and Figure 4(f). Furthermore, it can be seen that sharp discontinuities, e.g. the vertical left edge in the "Teddy" image, are accurately approximated.

With respect to the obtained bit rate, the following can be concluded. We have made a conservative estimate of the bit rate, assuming that the coding of the quadtree and the choice of coding modes 1-3, requires two bits per node. Moreover, we assume that 24 bits are used for *Mode 1* and 64 bits for *Mode 2*. The result of this conservative bit-rate estimation is summarized in Figure 5. The improvements that can be obtained by entropy coding are under study.

7. CONCLUSION

We have presented a new algorithm for coding depth maps that exploits the smooth properties of depth signals. Regions are modeled by piecewise linear functions and they are separated with straight lines along their boundaries. The algorithm employs a quadtree decomposition to enable the coding of small details as well as large regions with a single node. The bit rate (i.e. between 0.05 bit/pixel and 0.19 bit/pixel for synthetic and natural images, respectively) is controlled by a rate-distortion algorithm that selects the most appropriate

[1] From the Computer Vision and Pattern Recognition Group, University of Bonn, Germany.

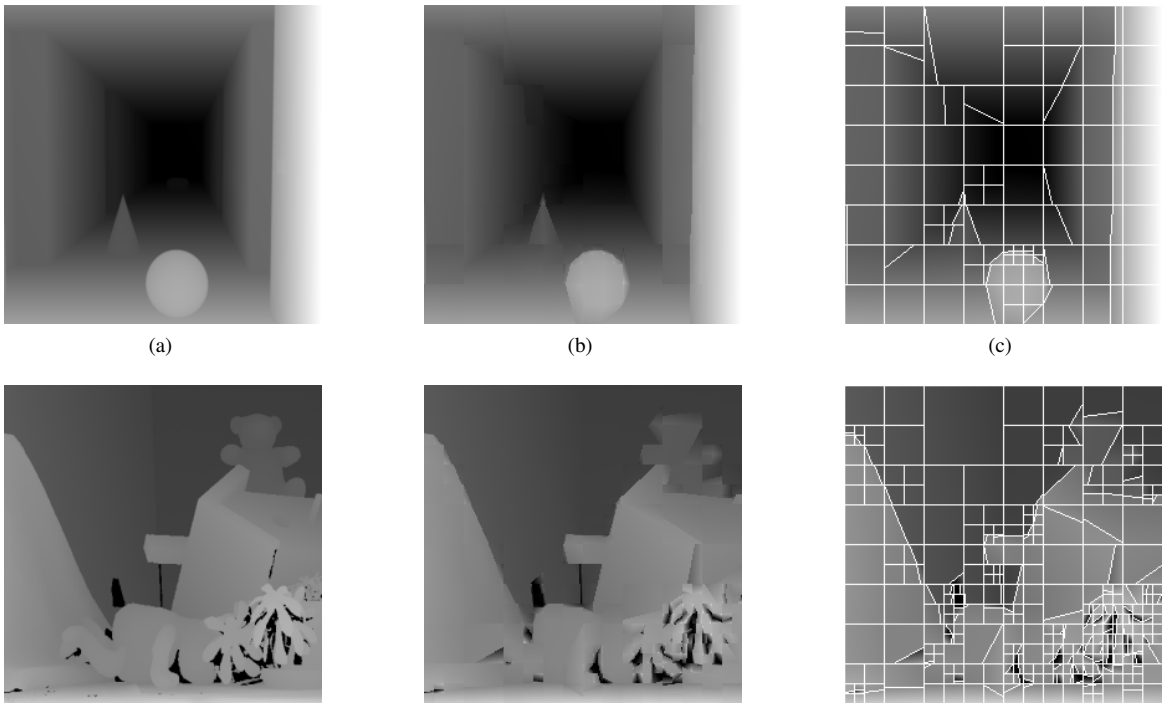
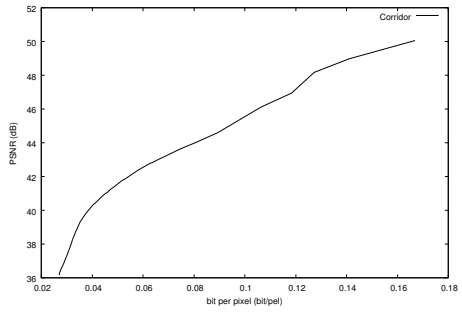
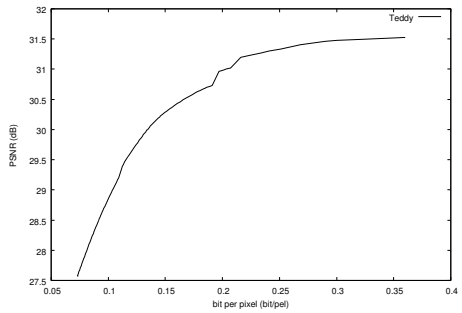


Figure 4: Figure 4(a) and Figure 4(d) show the original depth images "Corridor" and "Teddy", the corresponding reconstructed depth images are Figure 4(b) and Figure 4(e), respectively. The superimposed nodes of the quadtree are portrayed by Figure 4(c) and Figure 4(f). Coding for the depth images "Corridor" and"Teddy" is achieved at a bit rate of 0.05 bit/pixel and 0.19 bit/pixel for a PSNR of 41.9 dB and 30.7 dB, respectively.



(a)



(b)

Figure 5: Rate-distortion curve for the "Corridor" 5(a) and "Teddy" 5(b) depth image, employing the top-down analysis of Section 5.

coding mode and the level of quadtree decomposition. We have found an algorithm that features a top-down analysis of the coding modes which is sub-optimal, but much more computationally efficient than the optimal bottom-up analysis. Current research is exploring the benefits of entropy coding when it is added to the proposed algorithm.

REFERENCES

[1] S. Roy and I. J. Cox, "A maximum-flow formulation of the n-camera stereo correspondence problem," in *ICCV*, 1998, pp. 492–502.

[2] D. Tzovaras, N. Grammalidis, and M.G. Strintzis, "Disparity field and depth map coding for multiview image sequence," in *Proceedings on Int. Conf. Image Processing*, 1996, vol. 2, pp. 887–890.

[3] D. Scharstein and R. Szeliski., "High-accuracy stereo depth maps using structured light.," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2003, vol. 1, pp. 195–202.

[4] Y. Morvan, D. Farin, and P. H. N. de With, "Novel coding technique for depth images using quadtree decomposition and plane approximation," in *Visual Communications and Image Processing*, July 2005.

[5] A. Ortego and K. Ramchandran, "Rate-distortion methods for image and video compression," *IEEE Signal Processing Magazine*, vol. 15, pp. 23–50, 1998.