Coding of depth-maps using piecewise linear functions

CODING OF DEPTH-MAPS USING PIECEWISE LINEAR FUNCTIONS

Yannick Morvan[a], Dirk Farin[a] and Peter H. N. de With[a,b]
[a]Eindhoven University of Technology, 5600 MB Eindhoven, The Netherlands
[b]LogicaCMG, TSE-2, 5605 JB Eindhoven, The Netherlands
y.morvan@tue.nl, d.s.farin@tue.nl and p.h.n.de.with@tue.nl

An efficient way to transmit multi-view images is to send a single texture image together with a corresponding depth-map. The depth-map specifies the distance between each pixel and the camera. With this information, arbitrary 3-D views can be generated at the decoder. In this paper, we propose a new algorithm for the coding of depth-maps that provides an efficient representation of smooth regions as well as geometric features such as object contours. Our algorithm uses a segmentation procedure based on a quadtree decomposition and models the depth-map content with piecewise linear functions. We achieved a bit-rate as low as 0.33 bit/pixel, without any entropy coding. The attractivity of the coding algorithm is that, by exploiting specific properties of depth-maps, no degradations are shown along discontinuities, which is important for depth perception.

1. INTRODUCTION

The upcoming 3-D displays show 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 the set of images covering the same scene. Second, different displays will support a varying number of views, which makes it impractical to prepare the displayed views at the encoder. Instead, the views should be synthesized at the decoder, where display geometry is known. The independent transmission of several views can be avoided 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 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. For efficient transmission, the coding of depth-maps needs to be addressed.

In previous work, one of the approaches is to use a wireframe modeling technique [2] to code depth-maps. This technique divides the image into triangular patches, which are filled by linear functions. If the data cannot be represented with a single linear function, smaller patches are used for such that area. However, the patch structure is not adapted to the image content, such that a large number of small patches is generated along edges. An alternative technique uses a transform-based algorithm derived from MPEG-4 coders. Key advantage of using a standardized video coding algorithm to compress depth-maps is the backward compatibility with the existing technology. However, DCT transform coders generate ringing artifacts along edges that yields a blurry cloud of pixels along the object borders in 3-D reconstruction.

The characteristics of depth-maps differ from normal video signals. For example, it can be seen in Figure 1 that large parts of typical depth-maps usually contain smooth objects and other approximately planar surfaces. As a result, the input depth-map contains various areas of linear depth changes, corresponding to surfaces of objects. Furthermore, at the objects boundarie, the depth-map shows step functions, i.e. sharp edges. Therefore, we are interested in a compression algorithm that can efficiently decompose a typical depth-map into regions of linear depth changes, which are bounded by sharp edges.

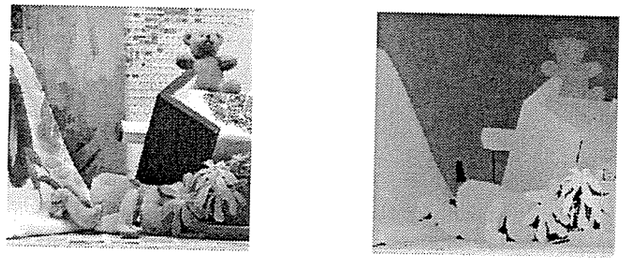


Figure 1: Example image "Teddy"[1] (left) and the corresponding depth-map (right).

This has brought us to the concept of modeling the signal using 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 with a quadtree decomposition and an independent linear function is used for each leaf of the tree. Additionally, we provide a mode that describes object contours, i.e. depth discontinuities, with a straight line separating two linear functions. This last mode

enables to code depth discontinuities, without introducing many small leafs along edges. Experimental results show that prior to entropy coding, the new proposal gives attractive low bit-rates for natural images, while simultaneously preserving the quality of the edges.

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

2. DEPTH-MAP CODING ALGORITHM

In this section, we present a novel approach for depth coding using the above-mentioned piecewise linear functions. 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 squares of different size. In some cases, the depth-map within one square can be approximated with a single linear function. If no suitable approximation can be determined for the square, it is subdivided into four smaller squares. Additionally to this standard quadtree subdivision, we apply a special coding mode when there are discontinuities in the depth-map. To prevent that many small squares are required along a discontinuity, we separate the square along a straight edge 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 square content with a single linear function;
- *Mode 2*: Subdivide the square along a straight line into two regions and approximate each region with an independent linear function;
- *Mode 3*: If the previous two modes fail, then subdivide the square into four smaller squares and recursively process these squares.

The decision for each coding mode is based on a rate-distortion optimization described in Section 4.

3. PARAMETER ESTIMATION FOR CODING MODES 1 AND 2

This section explains the computation of the parameters that are used in the two coding modes.

3.1 Parameter estimation for *Mode 1*

For *Mode 1*, a single linear function is used for which the three parameters a, b, c should be calculated. In order to determine the three parameters a, b, c of the linear function $f(x, y) = ax + by + c$, we employ a least-squares minimization. This algorithm minimizes the sum of squared differences between the depth-map $I(x, y)$ and the proposed linear model. Accordingly, parameters a, b, c are determined such that the error

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

is minimized, where n denotes the node size in terms of pixels. This error function $E(a, b, c)$ is minimized when the gradient satisfies $||\nabla E|| = 0$. When taking the partial derivatives of this equation with respect to a, b and c, we find a set of linear equations specified by

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

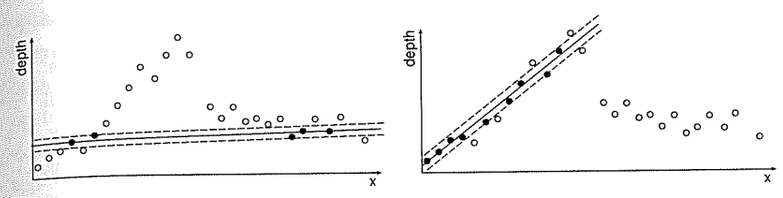
with

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

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

3.2 Parameter estimation for *Mode 2*

For *Mode 2*, the location of the separation line and the two sets of parameters for the two regions should be computed. This cannot be solved with a least-squares minimization for the following reason. Without knowing the subdivision bound-



(a) (b)

Figure 2: Two samples are randomly selected to compute a candidate model indicated by the line and input data close to the line candidate is considered as inliers (black dots). Fig. 2(a) and Fig. 2(b) show two candidate models with a small and a high number of inliers, respectively. The model with the larger number of inliers Fig. 2(b) is selected.

ary between the two regions, it is not possible to determine uniquely their model parameters. Similarly, it is not possible to identify the subdivision line as long as the region parameters are unknown.

Plane parameters estimation

For this reason, we apply a robust estimation using the RANdom SAmple Consensus (RANSAC [3]) algorithm. This algorithm can estimate parameters even if the pixel data are corrupted by a large number of outliers. In our case, we assume that the data comprise pixels from two regions with different models. We consider the data corresponding to the larger region as the inlier data, while the data in the smaller region are considered as outliers. Our algorithm starts with estimating the parameters for the larger region in a first run of the RANSAC algorithm. To this end, the algorithm randomly selects three pixels, which are used to compute candidate parameters a, b, c. The algorithm then counts the number of pixels that fit to this model (inliers). This procedure is repeated several times (see the 1-D example in Figure 2) and finally, the parameters with the largest support are selected as the parameters for region A (see Figure 3). Using the obtained parameters for region A, we remove all inlier pixels from the data. On this reduced data set, we carry out a second RANSAC parameter estimation. As a result, we obtain the parameters for the smaller region B from the second RANSAC estimation.

Subdivision line parameters estimation

The next phase is the determination of the parameters of the subdivision line. To obtain a robust estimation of line parameters, we build a dictionary of all possible subdivision lines that divide the square into two areas. Subsequently, an exhaustive search through the dictionary is performed and the best fitting model is selected. To evaluate the model fit, we compute a quality metric of the proposed model, i.e. the L_2 distance between the model and the image data. More formally, the best edge model minimizes the approximation error by

$$\min_{(A.B)\in D}\left(\sum_{(x,y)\in A}(f_A(x,y)-I(x,y))^2+\sum_{(x,y)\in B}(f_B(x,y)-I(x,y))^2\right),$$

where the parameters for f_A and f_B are computed such that the previously computed parameters resulting from the RANSAC procedure are reused.

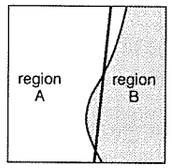


Figure 3: Regions A and B are identified with a classification map of inliers and outliers; the boundary between both regions is approximated with a straight line.

4. RATE DISTORTION ANALYSIS

This section describes the subdivision criterion of a node in the quadtree that we apply to optimize the rate-distortion behavior of the quadtree. 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 [4] $D+\lambda R$, 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 squares, 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 tree. However, this optimal solution is too computationally expensive. 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 than a bottom-up traversal, since we can stop the subdivision in smaller nodes as soon as *Mode 1* or *Mode 2* is selected.

5. EXPERIMENTAL RESULTS

For evaluating the performance of the algorithm, experiments were carried out using the depth-map of the "Teddy" scene. Experiments have revealed that the proposed algorithm can code large areas of the image with a single node (examples are the top left nodes in Figure 4(c)).

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 model and line parameters require 8 bits each. In total, we get 24 bits for *Mode 1* and 64 bits for *Mode 2*. This conservative bit-rate estimation results in, prior to possible entropy coding, a bit-rate of 0.33 bit/pixel at a PSNR of 32.6 dB for the image "Teddy".

6. CONCLUSION

We have presented an algorithm for coding depth-maps in 3D video that exploits the smoothness properties of such signals. Regions are approximated using piecewise linear functions and they are separated by straight lines along their boundaries. The length of a line segment (and thus the distortion) depends on the level of the quadtree decomposition in the actual area of the depth-map.

(a) (b) (c)

Figure 4: Example image "Teddy" (left) and the corresponding reconstructed depth-map (center) at a bit rate, prior to entropy coding, of 0.33 bit/pixel for a PSNR of 32.6 dB. The superimposed nodes of the quadtree are portrayed by Fig. 4(c).

The algorithm allows the coding of small details as well as large regions within a single node. The previous consideration results without any entropy coding in a bit-rate of 0.33 bit/pixel at a PSNR of 32.6 dB for the "Teddy" image. The performance of the algorithm is controlled by a computationally efficient top-down approach in which Lagrangian cost functions for individual coding modes are evaluated and coding modes giving minimum costs are selected at each step of the decomposition.

REFERENCES

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

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

[3] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

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