Depth Map Compression Using Color-Driven Isotropic Segmentation and Regularised Reconstruction

Citation

Year
2015

Version
Peer reviewed version (post-print)

Link to publication
TUTCRIS Portal (http://www.tut.fi/tutcris)

Published in
Data Compression Conference Proceedings

DOI
10.1109/DCC.2015.88

Copyright
© 2015 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Take down policy
If you believe that this document breaches copyright, please contact tutcris@tut.fi, and we will remove access to the work immediately and investigate your claim.
Depth map compression using color-driven isotropic segmentation and regularised reconstruction

Mihail Georgiev, Evgeny Belyaev, Atanas Gotchev

Tampere University of Technology, Finland

Abstract: “View-plus-depth” is a popular 3D image representation format, in which the color 2D image is augmented with a gray-scale image representing the scene depth map aligned with the color pixels. In this paper, we propose a novel depth map compression method aimed at finding an optimal spatial depth scale and down-sampling (sparsifying) the depth image over it. The down-sampled depth image is then compressed by a combination of a new arithmetic and a predictive coder. Our approach is motivated by the current achievements in multi-sensor 3D scene sensing where low-resolution depth map captured by time-of-flight sensors is successfully up-sampled and aligned with high-resolution RGB images. In our approach, color image segmentation in terms of super-pixels is used for finding the optimal depth scale and corresponding down-sampling. In contrast to other segmentation methods, it results in an isotropic and balanced low-resolution depth image, which is easily compressible. A bilateral regularizer is used for reconstructing the original-size depth map out of the low-resolution one and for splitting and predictive coding of segments with high reconstruction error. The scheme compares favorably with other methods for depth map compression.

1. Introduction

Depth map is an image-like representation of the geometry of a 3D visual scene observed from a particular viewpoint. It usually augments an aligned color 2D image, as seen from the same perspective, and represents the distance between each color pixel and its corresponding scene point in 3D space. The combined 2D color image and depth map are referred to as ‘view plus depth’ (V+D) or ‘RGB-Z’ representation format [1, 2]. The depth map is stored as a greyscale image having the same size as the color image, so each pixel of the 2D color image has a corresponding depth value on the map (c.f. Figure 1c, d). Such 3D scene representation is typically used to render perspective views on 3D displays (e.g. auto-stereoscopic 3D displays), or for 3D view generation by Depth-Image-Based Rendering (DIBR) methods [3]. It offers flexibility in decoupling scene capture by various capture setups from scene visualization on various displays and has been considered as part of the new 3D video compression (3DVC) standards [1]. It is also instrumental in computational photography for generating post-capture effects such as refocusing, vertigo or synthetic aperture.

Depth map data is created by techniques such as ‘structure-from-stereo’ or sensed by dedicated range (depth) sensors employing e.g. Time-of-Flight (ToF) sensing principles [4]. In both approaches, initial depth estimates come degraded (noisy) and with lower spatial resolution with respect to the resolution of the color camera. In passive ‘structure-from-stereo’ approaches degradations are caused by occluded, or textureless, or repetitive texture areas and/or disparity quantization [5]. In active ToF approaches, degradations are
cased by the non-confocal positioning of the RGB and ToF sensors and the presence of noise caused by ambient light, low illumination or short integration time of the sensor [6]. The range measurements depend heavily on the intensity of the reflected light, and therefore the sensor elements are made bigger (e.g. \( \sim 150\mu m \)) than those for the color sensing (e.g. \( \sim 2\mu m \)) [7]. As a result, the sensed depth map comes in very low-resolution (e.g. 120x160 pixels) [8]. Furthermore, its non-confocal position with respect to the color camera requires projective alignment bringing the depth map grid at irregular positions with respect to the color pixel grid. A great deal of research has been devoted to upsampling and refining estimated and sensed depth maps. Use has been made of the depth map nature as a piece-wise smooth function where smooth objects at different depths are delineated by sharp edges coinciding with edges of the same objects in the color image and related bi-lateral or multi-lateral joint filtering schemes have been proposed [9, 10]. Figure 1 illustrates a 3D scene sensed by a low-resolution depth sensor in noisy conditions, where the depth data has been denoised, upsampled and aligned with the color data, in order to be used for virtual view synthesis.

![Figure 1](HD Format)

In general, depth compression methods have been dealing with depth maps, which had been refined and aligned with the color images before compression. Several approaches have considered the piece-wise structure of the maps and suggested related decompositions in order to capture the edges along with large areas of constant or slowly varying depth. In [11], the authors have modeled depth blocks as piecewise-linear functions (platelets) and performed quad-tree decomposition in a wavelet packet fashion. The approach has been generalized in [12] for the case of anisotropic partitioning. A fractal-like splitting mechanism has been proposed leading to an anisotropic three structure better adapted to depth edges. Each block in the decomposition is approximated by a plane described by the block corner pixels, which are then predicatively encoded. Pyramidal decomposition for depth compression has been proposed in [13, 14]. The authors have employed two pyramid structures, one for arc breakpoints and another for sub-band samples. Other works have considered block partitioning and wedgelet edge modeling of non-rectangular intra-block segments [1]. The same wedgelet modeling approach has been combined with inter-component prediction, where blocks from the color image have been used to predict edge directions in the depth image [2]. A number of works have concentrated on detecting and encoding segments of the depth map, independently or in connection with the accompanying color image. A lossless compression method dealing with quantized disparity maps has been proposed in [15]. Depth contours are detected and efficiently encoded by a predictive encoder. The method has been applied also for lossy compression, where segments are obtained by iterative splitting into vertical and horizontal segments of constant value utilizing a greedy pursuit.
approach [16]. The inter-relation between color and depth contours has been exploited in [17], where the authors use color segmentation to predict shapes of different depth surfaces. Each depth segment is then approximated by a parameterized plane and subsequently encoded by an H.264/AVC codec. The representation requires storing auxiliary information about the segmentation. Another idea of storing segmentation contours as image areas has been studied in [18] along with an in-painting based predictive coding. Still, the approach requires storing pixel positions of segment border pixels.

In this contribution, we propose a novel approach to depth map compression in the joint V+D representation. Our approach is motivated by the fact that depth is a piece-wise function with low variations with respect to the resolution imposed by the aligned color image. Correspondingly, the inherent scale of the depth image is different and lower. This assumption is supported by the results of depth sensing where low-resolution depth maps have been successively upsampled and aligned with the accompanying color modality. Therefore, we develop our method on the concept of optimal depth map spatial scale, which can be inferred from the joint presence of color texture plus depth. To this end, we are after an isotropic depth downsampling related with that scale which leads to a depth image-like representation of low resolution. We employ isotropic color image segmentation in terms of super-pixels to find the corresponding downsampled depth version. We employ a bilateral filtering as a regularized depth reconstruction technique, which has proven its superiority in depth sensing applications. While the so-obtained downsampled depth map is easy to compress, the compression quality can still be improved by a subsequent partitioning of problematic segments followed by a predictive and specific arithmetic coding. We demonstrate that the method has a competitive performance especially for low bit-rates, making it especially applicable for applications where one wants to embed the compressed depth data into the color image. The scheme is very flexible as it can accommodate any established method for image and data compression.

2. Image segmentation in super-pixels

Super-pixel based segmentation plays a central role in our approach [19]. Super-pixels are segments that have regular (isotropic) and compact representation with low-computational overhead. A typical super-pixel behaves as a raster pixel on a low-resolution near-regular grid. Perceptually, super-pixel areas are homogeneous in terms of color and texture. Two main approaches for generating super-pixels can be cited, namely: SLIC (Simple Linear Iterative Clustering) [19, 20], and SEEDS (Super-pixels Extracted via Energy-Driven Sampling) [21, 22]. Examples of super-pixel segmentation are given in Figure 2.

An elegant feature of the super-pixel segmentation is that it takes the desired number of super-pixels as an input parameter and that for this number it is reproducible in terms of super-pixel areas and indexing [20]. It also allows for iterative refinements. Super-pixels follow the edge shape between color textures [23]. This is instrumental for finding also edge shapes between objects in the 3D scene, as illustrated in Figure 2. An image partitioned in super-pixels has a clear isotropic form in terms of rows and columns. Therefore, such an image can be stored and indexed efficiently as an isotropic map.
mimicking a 2D image. To ensure reproducing the same super-pixels, the segmentation is applied on the already compressed color image in the encoder in a closed-loop.

In our approach we use the super-pixel segmentation in order to find an optimal depth downsampling. One can create a number of successive super-pixel splitting patterns and use them for depth map downsampling, that is, replacing the depth values within each segment by a constant. The optimal scale and down-sampling is selected subject to the given bit budget and targeted quality. The obtained depth image resembles a sensed low-resolution depth map and the original-resolution depth can be reconstructed by a regularized filtering, involving the color modality, using the super-pixels as adaptive support. Note, that the main benefit of super-pixels in contrast to other segmentation approaches is that the partitioning information (e.g. contours, edges, coordinates, indexing) is not required since the segmentation is fully reproducible.

![Figure 2: Super-pixel segmentation. From left to right: initial V+D; color and depth segmented by SEEDS; and by SLIC](image)

### 3. Depth map representation

In this section we describe the conversion of the initial depth map into an isotropic map and refinement partitioning, both driven by an optimal super-pixel segmentation of the color image.

#### 3.1. Finding an optimal down-sampled depth map and building an isotropic map

Consider a color image in YUV or RGB color space $y(n) = [y^Y(n), y^U(n), y^V(n)]$ or $y(n) = [y^R(n), y^G(n), y^B(n)]$ and the associated per-pixel depth $z(n)$, where $n = [n_1, n_2]$ is a spatial variable, $n \in \mathbb{N}$, $\mathbb{N}$ being the image domain. Consider several successive color image segmentation stages $l = 0, 1, \ldots, L - 1$ resulting in increasing (refined) number of super-pixels $N_0 < N_1 < \ldots < N_l < \cdots < N_{L-1}$. Denote the area of a super-pixel at stage $l$ and indexed by $i$, $i = 0, 1, \ldots, N_l - 1$, by $\Omega_i, \bigcup_{i=0}^{N_l-1} \Omega_i = N$. Under the assumption for a piece-wise constant depth within each super-pixel segment, the depth values within the segment are replaced by a constant value $z(n) = z_{l,\text{opt}}, n \in \Omega_i$, which minimizes the mean squared error (MSE) between the original depth and the depth reconstructed by a cross-bilateral filter:

$$z_{l,\text{opt}} = z(k),$$
\[ k = \arg\min_{n \in \Omega_i^l} \left( z(n) - \frac{\sum_{m \in R} e^{-\alpha \frac{\|n-m\|}{\|n-m\|}} e^{-\frac{\|y(n)-y(m)\|}{\|n-m\|}}}{\sum_{m \in R} e^{-\alpha \frac{\|n-m\|}{\|n-m\|}} e^{-\frac{\|y(n)-y(m)\|}{\|n-m\|}}} z(k) \right)^2; \quad k \in \Omega_i^l. \quad (1) \]

For each segmentation stage \( l = 0, 1, \ldots, L - 1 \), the parameters of the filter \( \alpha, \beta \) and the filter size \( R \) are also optimized to give a minimum MSE [9]. Figure 3 shows the effect of the cross-bilateral filtering on the quality of the reconstructed depth. The first column represents the scene in V+D format while the second column presents four zoomed-in patches. The effect of super-pixel driven down-sampling is illustrated by the images in column 3, while the depth reconstructed by the bilateral filter is shown in the right most column.

![Figure 3: The effect of depth-downsampling and cross-bilateral filtering on the reconstructed depth quality. From left to right: V+D; zoomed-in patches in the original depth; result after down-sampling; result after cross-bilateral reconstruction.](image)

Given a target bit-budget and a targeted quality, an optimal super-pixel segmentation \( L_{opt} \) can be selected. The corresponding down-sampled depth is then converted to an isotropic map in the form of a regular image \( d(m) \), where each pixel takes the value of a rectangular tile, approximately covering one super-pixel of constant depth. In other words, the irregular super-pixel areas get replaced by equally sized rectangular tiles. In practice, this works like finding the optimal size of the individual tile, which would have the best aspect ratio determined by the irregular super-pixels and covering relatively the same area. Figure 4 illustrates the procedure. In the figure, the first column represents the V+D format where the color image is segmented in super-pixels and a candidate regular grid is super-imposed. The two remaining depth images at the bottom row represent down-sampled depth for different super-pixel segmentations. The corresponding upper-row images show the obtained regular (isotropic) maps. The upsampling from the low-resolution isotropic map back to the original size of the depth map is straightforward.

### 3.2. Refinement partitioning

Partitioning can further refine the depth downsampling structure fixed by the optimal super-pixel segmentation. Super-pixels that cause high MSE in the reconstructed depth are processed. The refinement makes use of the super-pixel segmentation stages \( L_{opt} + 1, L_{opt} + 2, \ldots \). Segmentation elements at these stages maintain the isotropy while the covered area of related super-pixels in two consecutive stages do not coincide in general.
For that reason, we implement a refinement, which also updates the neighboring segments of the segment being refined.

![Segmentation Diagram](image)

**Figure 4:** Conversion from down-sampled depth to isotropic (regular) map for different super-pixel segmentation: a) 128, b) 512, and c) 2048 elements

Consider a segment $i$ at stage $L_{opt}$ covering an area $\Omega_i^{L_{opt}}$ and causing highest MSE, hence to be refined. A set of indices of neighboring segments is denoted by $K_i$ and their areas are denoted by $\Omega_k^{L_{opt}}, k \in K_i$. The subsequent super-pixel segmentation $L_{opt} + 1$ leads to finding super-pixels fully overlapping the area $\Omega_i^{L_{opt}}$ and partially overlapping (some of) areas $\Omega_k^{L_{opt}}, k \in K_i$. The latter areas are shrunk by excluding the overlapping areas and then the associated depth values are updated by re-running the bilateral optimization in Eq. (1). The procedure is illustrated by Figure 5. Note that the update does not take place if the neighboring sub-pixels have been already refined. The procedure continues with the segment causing the next highest MSE. The refinements can also go deeper into further states $L_{opt} + 2, L_{opt} + 3, \ldots$ etc. creating a tree structure to be predictively encoded.

![Refinement Diagram](image)

**Figure 5:** Proposed refinement by super-pixel inter-stage partitioning. From left to right: super-pixel $C$ at level $L$ to be split and neighboring super-pixels $A$ and $B$; super-pixels of upper level $L+1$ partially overlapping $A$ and $B$; updated super-pixels $A$ and $B$.

The final data structure contains the following components: isotropic map (a 2D grayscale image) $d(m)$ corresponding to the segmentation stage $L_{opt}$, a tree $B$, containing the subsequent stages and list of refined segments per stage; and corresponding refinement values in a sequence $P$. 
4. Encoding depth map and related structure

4.1. Predictive encoding specific to super-pixel partitioning

The isotropic map $d(m)$ is predictively encoded similarly to JPEG-LS standard [24]. With reference to Figure 6a, consider the pixel element $X^0$ to be predicted by neighboring elements $A^0$, $B^0$ and $C^0$. The predictor $Pr^0$ is defined as:

$$Pr^0 \begin{cases} 
\min(A^0, B^0) & \text{IF } C^0 \geq \max(A^0, B^0) \\
\max(A^0, B^0) & \text{IF } C^0 < \min(A^0, B^0) \\
A^0 + B^0 - C^0, & \text{otherwise}
\end{cases}$$

(2)

Figure 6: Proposed prediction for super pixels encoding
a) isotropic map for initial isotropic level, and b) for remaining levels

Refinement super-pixels, e.g. $X^1_0, X^1_1, X^2_2, X^3_3$ as in Figure 6b are predicted by their parents and the prediction differences are stored in the sequence $P$ and subsequently encoded by an adaptive multi-alphabet range coder [25]. For achieving higher efficiency, a uniform scalar quantization of prediction errors prior to their arithmetic encoding is applied.

4.2. Encoding of partitioning index tree

Figure 7 shows the structure of the tree, which contains the indexes of refined super-pixels. First, the isotropic map is indexed for the refined and non-refined elements, where indexes of non-refined elements are encoded by zero and indexes of refined elements are encoded by 1. This forms a binary map, which is scanned column-wise to form the first tree level. Next level is for elements of segmentation stage $L_{opt} + 1$. At this stage, elements being further refined are encoded by 1 and those, which are not – by zero. The subsequent level performs the same for stage $L_{opt} + 2$ and so on. Note that we do not need to store information about the number of children of refined super-pixels, as this is automatically found when the super-pixel segmentation for a particular stage is run.

The binary isotropic map is encoded separately from the rest of the tree. We apply an own realization of Context-Adaptive Binary Range Coder (CABRC) [26]. For context modeling we assume that “split/no split” of current super-pixel depends on “split/no split” of its neighbors. Using this assumption, to compress partition map for current super-pixel $X^0$, we use four binary contexts:

$$If \begin{cases} 
\text{all neighbors} : A^0, B^0, C^0 \text{ are not split} \rightarrow 0 \\
\text{one neighbor in } A^0, B^0, C^0 \text{ is split} \rightarrow 1 \\
\text{two neighbors in } A^0, B^0, C^0 \text{ are split} \rightarrow 2 \\
\text{all neighbors} : A^0, B^0, C^0 \text{ are split} \rightarrow 3
\end{cases}.$$
The remaining tree levels are compressed using one binary context.

5. Experiments and results

We present experiments for two V+D images: *Breakdancer (View 0, Frame 0)* and *Ballet (View 4, Frame 0)* [27]. We compare the PSNR between the original and decompressed depth vs the bitrate measured in bits per pixel (bpp). For our method, we present results without and with bilateral regularization and without and with predictive refinement. We compare our results with those in Milani et al. [17], P80[13, 14], GSOs+CCLV[16], H.264/AVC Intra, and JPEG 2000 standard. The results are summarized in Figure 9. Our method is quite competitive, especially for moderate (0.05÷0.2 bpp) and very low (< 0.05 bpp) bitrates. Even for the basic implementation and with no refinement, it reaches PSNR above 40 dB. At such quality, the reconstructed depth maps are considered very good for virtual view rendering. The refinement steps contribute further improvement, which in some cases might go above 50 dB, which is near-lossless.

6. Conclusions

We have presented a depth compression method based on depth down-sampling driven by super-pixel color segmentation. Among other state-of-the-art methods, our method excels with its simplicity and elegant encoding paradigm based on the assumption that depth scale is lower than the color image one. The method offers scalability, e.g. refinement steps can be omitted in some less-demanding or real-time applications.
Varying the starting number of super-pixel elements easily controls the performance. The optimization procedure facilitates finding such super-pixel segmentation, which agrees with the depth boundaries. The superior results for very low and moderate bitrates position the method as a candidate in applications targeting embedding the encoded depth as auxiliary data into the image coded stream. The depth down-sampling mechanism is simplified to great extent and can further benefit from powerful depth restoration methods originally developed in the context of active or passive depth sensing. The method has also a side effect of aligning color and depth edges which is beneficial for virtual view synthesis.

Figure 9: PSNR between original and decoded depth map vs bpp. Breakdancer (View 0, Frame 0 (up)) and Ballet (View 4, Frame 0 (bottom)); Encoding modes (from left to right): isotropic map only with regularization; predictive refinement and quantization.

7. References

Methods for depth superpixels ex.

Highly scalable coding of depth representation


