In A. Kuijper, K. Bredies, T. Pock, H. Bischof (Eds.):
Scale-Space and Variational Methods in Computer Vision.
Lecture Notes in Computer Science, Vol. 7893, 319–330, Springer, Berlin, 2013.
The final publication is available at link.springer.com.

Compression of Depth Maps with Segment-based Homogeneous Diffusion

Sebastian Hoffmann, Markus Mainberger, Joachim Weickert, and Michael Puhl

Mathematical Image Analysis Group
Faculty of Mathematics and Computer Science, Campus E1.7
Saarland University, 66041 Saarbrücken, Germany
{hoffmann,mainberger,weickert,puhl}@mia.uni-saarland.de

Abstract. The efficient compression of depth maps is becoming more and more important. We present a novel codec specifically suited for this task. In the encoding step we segment the image and extract between-pixel contours. Subsequently we optimise the grey values at carefully selected mask points, including both hexagonal grid locations as well as freely chosen points. We use a chain code to store the contours. For the decoding we apply a segment-based homogeneous diffusion inpainting. The segmentation allows parallel processing of the individual segments. Experiments show that our compression algorithm outperforms comparable methods such as JPEG or JPEG2000, while being competitive with HEVC (High Efficiency Video Coding).

Keywords: depth map, image compression, segmentation, homogeneous diffusion inpainting, partial differential equations (PDEs)

1 Introduction

In recent years, 3D cinema technology has become increasingly popular. In the corresponding so called multi-view video + depth (MVD) format, multiple images are captured from different perspectives along with their respective depth images. To cope with the huge amount of data, an efficient compression is indispensable. Combined compression methods have been presented (see e.g. [1]), where the correlated information between the colour images and the corresponding depth maps is exploited. It is also possible to incorporate a temporal component into the compression framework. In this paper, however, we focus exclusively on the problem of depth map compression.

Besides well-established methods like JPEG or JPEG2000, compression algorithms based on partial differential equations (PDEs) recently gained attention. While in the encoding step, only a small subset of all pixels is selected and stored, the missing information is reconstructed by means of PDE-based interpolation when decoding. This idea was introduced by Galić et al. in 2005 [2] and extended in 2008 [3]. A further developed version of Schmaltz et al. [4] was able to beat JPEG2000. In the special case of cartoon-like images, which are in their nature very similar to depth maps, a much simpler and computationally favourable PDE-based codec has been proposed: In [5] the authors encode the grey or colour values on both sides of image edges. In contrast to the nonlinear anisotropic diffusion processes used in the aforementioned methods, a basic homogeneous diffusion inpainting is then sufficient to reconstruct missing pixels. For cartoon-like images the method could outperform JPEG2000. Indeed, for depth map compression, extracting and storing edges is actually a natural idea as they are crucial to obtain a good visual perception of the geometry. However, due to the fact that the above codec cannot handle homogeneous variations, it turns out that its application to depth maps leads to unsatisfactory results.

In [6, 7] modified versions of this edge-based approach have been suggested. The main changes consist of adding grey values at regular mask points, exploiting different edge detectors, and encoding parts of the extracted data with other methods. Similarly homogeneous diffusion can be incorporated in existing block-based approaches where additional edge information is used to attain sharp edges [8]. However, all these methods are either data intensive or lead to a fairly complex overall codec.

Other approaches try to split the depth image recursively into smaller parts, resulting in a tree structure, and recover the depth map on the lowest tree level by means of linear interpolation [9, 10]. In [11], the depth map is approximated by linear functions within segments. A similar method, also working on segments, has been introduced in [12] where mainly bilinear interpolation of data on a regular grid has been used to reconstruct the depth information. All these methods have the drawback that a lot of information has to be stored to be sufficiently flexible.

The goal of the present paper is to address the aforementioned problems. We present a conceptually simple codec for depth maps. While it is also based on homogeneous diffusion inpainting, it differs from [5–7] by the fact that it replaces edges by closed contours that result from a segmentation. This creates a decoupling into sub-problems and allows to benefit from parallel implementations. More importantly, by assuming homogeneous Neumann boundary conditions between segments, we show that it is unnecessary to store grey values at contours. Instead, we select hexagonal grid points as well as points at some specific locations. The corresponding grey values to be stored are optimised. In the end we do not only achieve a codec for depth maps that outperforms JPEG [13] and JPEG2000 [14], but even has the potential to compete with the substantially more complex HEVC (High Efficiency Video Coding), which is one of the most favorable methods to encode this type of images [15].

Our paper is organised as follows. First we introduce segment-based homogeneous inpainting in Section 2. Based on this concept we describe our encoding process in Section 3 and discuss the corresponding decoding steps in Section 4. Experimental results will be presented in Section 5, and a summary in Section 6 concludes the paper.

2 Segment-based Homogeneous Diffusion

One key element of our compression codec is the *segment-based homogeneous diffusion (SBHD)* that is used to reconstruct an image from a small amount of stored data. Given a greyscale image $f(x, y)$, SBHD relies on a segmentation of the image domain into several sub-domains. For each of the segments, we assume the grey values at specific points - the so called mask points - to be given. This information is used to inpaint the rest of the respective segment.

The inpainting can be described as computing the steady state solution of the *homogeneous diffusion equation* [16]

$$\partial_t u = \Delta u \tag{1}$$

subject to the following *mixed boundary conditions*:

$$\begin{cases} u = f & \text{at mask points} \quad (\textit{Dirichlet boundary conditions}) \\ \partial_{\boldsymbol{n}} u = 0 & \text{at segment boundaries} \ (\textit{homogeneous Neumann boundary cond.}) \end{cases}$$

Thereby, $\boldsymbol{n}$ is the unit normal vector to the respective segment boundary, and $\partial_{\boldsymbol{n}} u$ denotes the partial derivative of u in normal direction. The discretisation of this partial differential equation can be done in a straightforward way by using finite differences [17]. Then, as long as we have at least one mask pixel in each segment, there exists an unique solution of the discrete problem (cf. [18]).

As a result of SBHD we obtain an image containing (i) sharp edges at segment boundaries and (ii) smooth transitions within segments steered by the values at the mask points. These transitions can also represent unsharp edges. SBHD is therefore well suited for the representation of depth images.

An important advantage of this SBHD is the fact that, by construction, each mask point does not have a global influence: Its impact is limited by the respective segment boundaries. This allows a segment-wise parallel processing as we will see later. In order to get a solution of the diffusion equation we make use of the *fast explicit diffusion (FED)* scheme [19] together with a CUDA implementation on the GPU. Compared to a CPU version we achieve a substantial speedup due to the parallelism. Another speedup is gained by reducing the number of required iterations to reach a steady state. To this end, we initialise all unknown values at non-mask points with the mean value of the known grey values within each individual segment.

3 Data Extraction and Encoding

Our encoding algorithm consists of several parts that are described next.

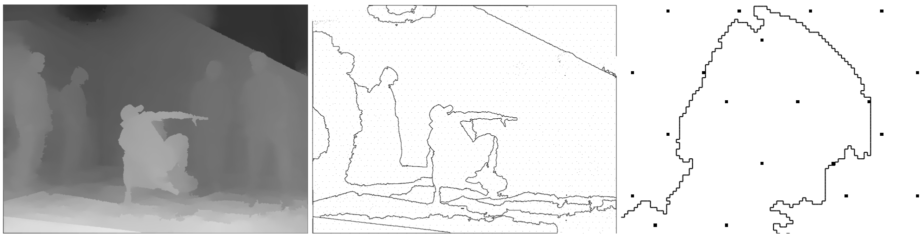

Fig. 1. Contours extraction example. **Left:** Original depth image *breakdancers* (1024×768). **Middle:** Extracted closed contours between pixels gained by the segmentation ($T_1 = 1$, $T_2 = 5$, $\sigma = 0.5$). **Right:** Zoom at the contour of one head. Gray values are given at specific mask points (black dots).

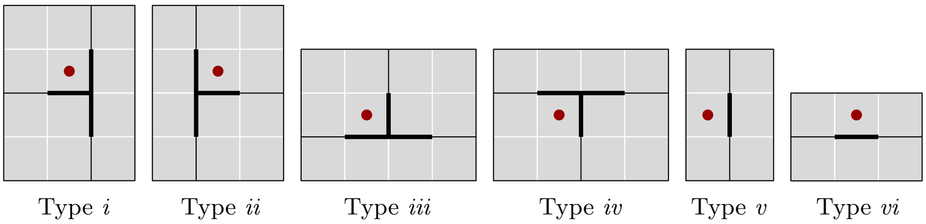

Fig. 2. Different types of edge crossings along with their respective reference point (red dot).

3.1 Part 1 - Segmentation

The first step is to find a segmentation fulfilling two properties. On the one hand there should be a large contrast at the boundary between adjacent segments, such that it pays off to save information at these locations. Since we store the contour information it is possible to precisely recover these sharp edges. On the other hand we want to have smooth transitions within each of the segments such that they can be reconstructed well by our SBHD inpainting from the existing information at mask points.

Our segmentation algorithm consists of two steps. First a region growing algorithm is applied to get four-connected segments. A threshold T_1 thereby determines whether or not a neighboring pixel belongs to the same segment. Small values of T_1 usually lead to a slight over-segmentation. Therefore, in a second step, we consider a Gaussian smoothed version of the original image with a standard deviation of σ. In this image we compute the mean contrast between adjacent pixels along a contour separating two neighboring segments. Successively we remove the boundary with the lowest average contrast. We repeat this as long as the lowest contrast is smaller than some threshold T_2. This method yields very precise contours between adjacent segments while allowing smooth transitions not to be split.

As already mentioned the segmentation immediately yields closed contours at between-pixel locations. An example depth image along with the extracted contours is depicted in Figure 1. As desired, there is no contour around the two people on the right hand side of the image because of the smooth transition to the background. It is the task of data points within the segments to restore such smooth flows in the reconstruction.

If we want to use an existing codec like the JBIG (*Joint Bi-level Image Experts Group*) standard [20] to encode the contour information we would have to store two binary images, i.e., the edges in x- and y-direction, respectively. Alternatively we store this information more efficiently with a chain code. The advantage of between-pixel edges is that there are only three possible directions, whereas we would have seven directions when considering pixel chains. Thus, the chain code is highly efficient.

In the beginning we extract all T-junctions in the edge map (see Fig. 2, types *i*-*iv*). To store them, we have to save the respective reference point coordinates along with the type number. Starting at these crossings we can build a chain code and stop whenever we reach an edge that has already been visited. Not needed starting elements can be removed afterwards. The only thing which remains are contours without any crossing. Therefore, we add two more starting element types representing only one edge, either in x- or in y-direction, respectively (see Fig. 2, types *v* and *vi*). The corresponding chain code has to be stored as well. Afterwards we employ a sophisticated lossless context-based entropy coder to encode the obtained edge information, namely the PAQ compression [21], version PAQ8o6.

3.2 Part 2 - Mask Points

So far we only focused on the location of segment boundaries in the image. However, we also have to encode some data for being able to reconstruct the grey values within each of the segments. Therefore, we use a mixture of regularly sampled mask points and points at freely chosen positions as described in the following subsections. The overall amount of selected mask pixels is a free parameter which allows us to steer the compression ratio.

(a) Hexagonal Mask

To reduce the coding costs for the location of prescribed grey values we initially choose points according to a specific pattern or algorithm to limit the coding overhead. One possibility would be to uniformly sample random points over the whole domain given a specific seed. A drawback of this method is that there are specific areas where points are clustered, i.e., the distance between neighboring points is not equal. Another approach would be to use mask points at a regular grid as done in [6]. However, since we want a good covering of the whole domain this is still not optimal. What we want is a mask where the minimum distance between two distinct points is maximised. To get a good approximation one could assume that no boundaries are present. In this context it is known that

Input: Original image f, initial mask m,
percentage p, percentage q, desired mask density r
Output: Mask m with density r

While density of m smaller than r do

1. Apply SBHD using the mask m and save result in u.
2. Choose randomly p percent of all non-mask points as a candidate set C.
3. At all locations in C, compute the squared difference between f and u
4. Let $T \subset C$ be the set containing the q percent of all locations from C where the error computed in Step 3 is largest.
5. Set all locations from T as mask points in m.
6. Clear T and C.

Fig. 3. Probabilistic densification algorithm

the hexagonal packing is the optimal one in the two-dimensional Euclidean plane [22]. This is why we make use of such an hexagonal grid pattern. In order to make sure that there is at least one mask point in each segment, we compute the hexagonal mask separately for each individual segment. Thereby a given density value determines the number of points.

(b) Probabilistic Densificaton and Nonlocal Pixel Exchange

In addition to the hexagonal mask we want to store some more mask points for a further quality gain. We accept the larger coding costs of these free points and place them at locations where the quality can be improved most. Therefore, we perform in a first step a so-called *probabilistic densification*, which is similar to the probabilistic sparsification process described in [18]. We consecutively select points as additional mask points, starting with the ones from the hexagonal mask as an initialisation. The decision where we insert new points in each step depends on where the difference of the respective reconstruction to the original image is largest. This process is repeated until we reach a desired density. The exact algorithm is depicted in Figure 3. Experiments have shown that a good choice for the parameters is $p = 10\%$ and $q = 0.1\%$.

Moreover, it has been demonstrated that it pays off to apply a so called *non-local pixel exchange* [18], which tries to find better combinations for the mask points not lying on the hexagonal grid. Thereby a small number k of non-mask points is randomly selected, and the local error is computed. Then one mask point is chosen, exchanging its position with the position of the largest computed error. If this improves the reconstruction we use the new mask, otherwise the exchange is reverted. Thus, we can only improve our result. For more details we refer to [18]. We use the parameter $k = 10$, as this choice turns out to yield

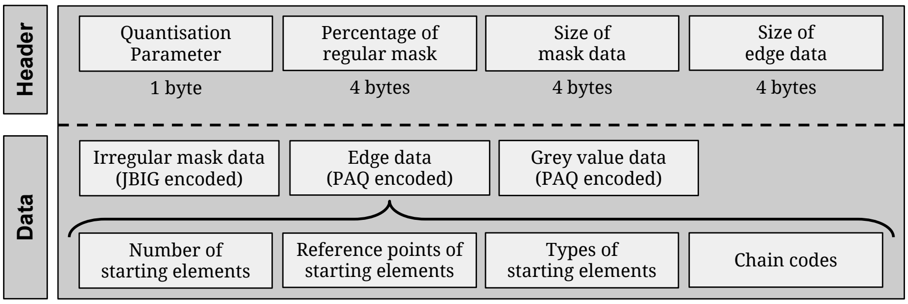

Fig. 4. File structure of our proposed codec.

good results. After only 1000 iterations, the reconstruction quality can already be improved tremendously. The pixel exchange step can also be left out if one is interested in a faster encoding method with lower quality.

While only having to store one density parameter for the hexagonal mask, the location of the additional free mask points is encoded using the JBIG encoder [20].

3.3 Part 3 - Grey Values

After the determination of the mask pixel locations it also pays off to optimise the grey values that are stored at these positions. Usually the respective grey values of the original image are adopted. However, this is not always the best solution with respect to the global reconstruction error. Instead of being fully exact at the selected pixels, it makes sense to allow for some deviation at these locations to achieve an overall smaller global error. In order to find optimal values we make use of a least squares approach as suggested in [18]. Note that, as we are using SBHD, the computations can be speeded up by computing the optimal grey values for the individual segments in parallel.

The resulting values at mask locations are then quantised to d different values and afterwards stored in a list. The order of the mask points is chosen such that we go from segment to segment, which has the advantage that subsequent values in the list are more likely to be similar due to the design of our segmentation. To obtain a compact representation, we use the PAQ encoder mentioned in Section 3.1.

3.4 Overview of File Structure

We are now able to write all the gathered information into one file having the structure as depicted in Figure 4. Note that we do not have to store the image dimensions in the header since they are already contained in the JBIG mask data.

4 Decoding

In the decoding phase we can follow a straightforward process chain. First of all we scan the stored header information and split the main data into the edge data, the irregular mask, and the grey value information. Edges can be restored by placing the starting points and following the single contour chains. Afterwards we compute the hexagonal mask and add the irregular mask points to it. The grey values are placed at the corresponding locations, and we finally obtain the reconstruction via SBHD.

5 Experiments

In this section, the potential of the proposed codec is presented by considering three different existing methods. Besides the well-established standard JPEG and its successor JPEG2000, we will also compare our method with the designated future standard HEVC (High Efficiency Video Coding), version HM-8.2. Although this codec is designed for the purpose of video coding, it also provides an intra-coding mode for the efficient compression of still images [15].

As test images we use the so-called *breakdancers* and *ballet* depth maps, both taken from the MVD sequence in [23] (respective image size: 1024×768). We determine that 80% of all mask points lie at hexagonal grid locations and set the quantisation parameter to $d = 64$. This choice has experimentally shown to yield a good trade-off between coding costs and quality gain. We keep these parameters fixed for all experiments. Figure 5 depicts the results for a compression rate of 0.045 bits per pixel (bpp), which roughly corresponds to a compression ratio of 180 : 1. The overall mask density in this case is 0.3% and 0.45% for the images *breakdancers* and *ballet*, respectively.

The transform-based methods JPEG and JPEG2000 often perform well when it comes to the compression of standard natural images. However, both methods suffer from artifacts around edges. When it comes to depth images, these block or ringing artifacts around object boundaries are visually perceived much more unpleasant than in smooth image regions. HEVC seems to overcome these problems, but tends to smooth out some of the edges, which can lead also to a distorted geometric perception. With our SBHD method it is hard to notice any difference between the original image and the reconstruction.

In a quantitative comparison we measure the error between the original image and the respective reconstruction by means of the peak-signal-to-noise ratio (PSNR). The resulting measurements can be seen in Figure 6. Note that for JPEG it is not possible to reach very high compression rates. As a larger PSNR value corresponds to a higher similarity between two images, one can see that our method clearly outperforms JPEG and JPEG2000. Our codec is even able to exceed the quality of HEVC for some compression rates.

Furthermore, we can also evaluate the results considering a more perceptual error measure like the structural similarity index (SSIM) [24]. We use the available MatLab version with standard parameters from [24]. Figure 7 depicts the results. Note that a SSIM closer to 1 denotes a better visual similarity. Compared to JPEG or JPEG2000, it is visible that our proposed method reaches better results for almost all tested compression rates. HEVC and SBHD give reconstructions of comparable quality. This result is remarkable since, in contrast to HEVC, our algorithm makes use of relatively simple and straightforward concepts. We thus believe that it has potential for further improvement.

In our current proof-of-concept implementation, it takes several minutes for an image to be encoded, depending on its size and the number of mask points. The decoding of a depth image of size 1024×768 can be done within a second on a modern PC. It is important to mention that there is a lot of potential for accelerating this process. For example, one can incorporate bidirectional multigrid methods into SBHD [5]. In this way, we expect that real-time decoding becomes feasible for practical applications.

6 Summary

We have shown that a combination of two relatively elementary concepts can lead to a remarkable compression quality of depth maps: a region-growing segmentation method as well as a homogeneous diffusion inpainting with carefully selected data points. In our evaluation, this segment-based homogeneous diffusion (SBHD) codec clearly outperforms JPEG and JPEG2000. Moreover, it performs competitively with HEVC, especially in terms of perceptual quality.

In our ongoing work we are extending our framework to colour-valued data such as cartoon-like images. Moreover, we are going to incorporate additional information such as multiple views, combined colour / depth images, and their temporal extensions. We are optimistic that this will help to demonstrate the widely unexplored strength of diffusion ideas for data compression.

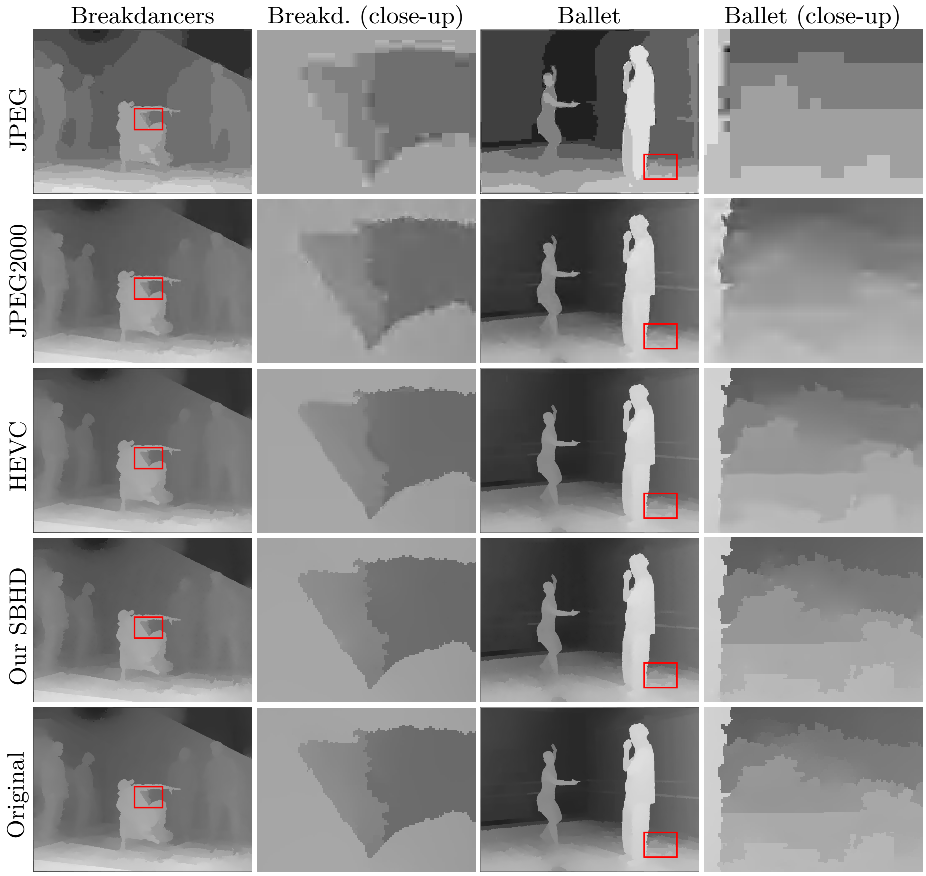

Fig. 5. Comparison of different compression methods for two depth images using a compression rate of 0.045 bpp. The boxes denote the area of the respective close-ups.

Acknowledgements. We thank Marco Zamarin and Søren Forchhammer from the Technical University of Denmark for drawing our attention to this topic.

References

1. Ruiz-Hidalgo, J., Morros, J.R., Aflaki, P., Calderero, F., Marqués, F.: Multiview depth coding based on combined color/depth segmentation. Journal of Visual Communication and Image Representation **23**(1) (January 2012) 42–52
2. Galić, I., Weickert, J., Welk, M., Bruhn, A., Belyaev, A., Seidel, H.P.: Towards PDE-based image compression. In Paragios, N., Faugeras, O., Chan, T., Schnörr, C., eds.: Variational, Geometric and Level-Set Methods in Computer Vision. Volume 3752 of Lecture Notes in Computer Science. Springer, Berlin (2005) 37–48

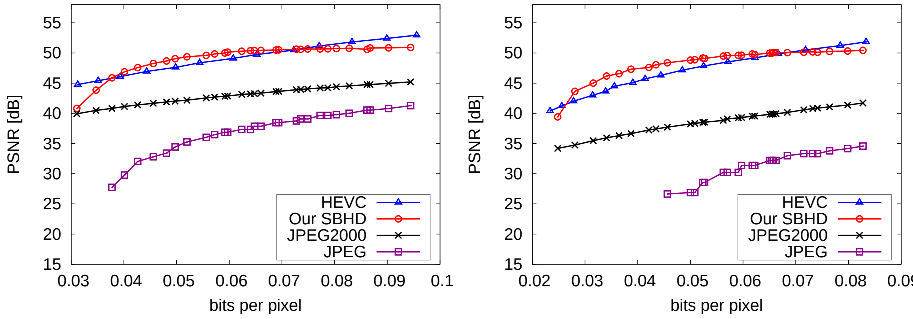

Fig. 6. Quantitative comparison to JPEG, JPEG2000 and HEVC. **Left:** *Breakdancers* image. **Right:** *Ballet* image.

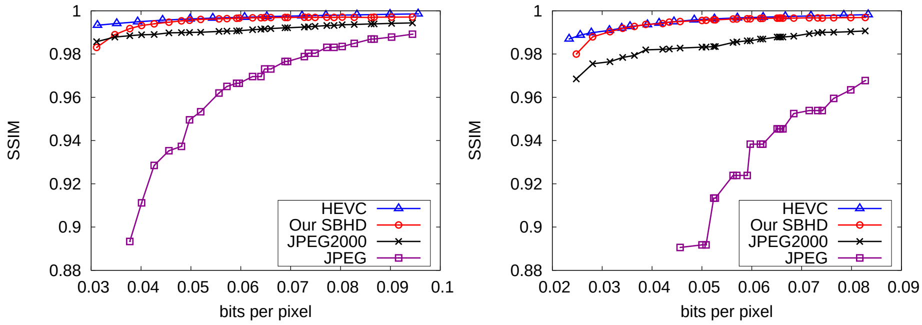

Fig. 7. Perceptual comparison to JPEG, JPEG2000 and HEVC using the SSIM measure. **Left:** *Breakdancers* image. **Right:** *Ballet* image.

3. Galić, I., Weickert, J., Welk, M., Bruhn, A., Belyaev, A., Seidel, H.P.: Image compression with anisotropic diffusion. Journal of Mathematical Imaging and Vision **31**(2–3) (July 2008) 255–269
4. Schmaltz, C., Weickert, J., Bruhn, A.: Beating the quality of JPEG 2000 with anisotropic diffusion. In Denzler, J., Notni, G., Süße, H., eds.: Pattern Recognition. Volume 5748 of Lecture Notes in Computer Science. Springer, Berlin (2009) 452–461
5. Mainberger, M., Bruhn, A., Weickert, J., Forchhammer, S.: Edge-based image compression of cartoon-like images with homogeneous diffusion. Pattern Recognition **44**(9) (September 2011) 1859–1873
6. Gautier, J., Meur, O.L., Guillemot, C.: Efficient depth map compression based on lossless edge coding and diffusion. In: Picture Coding Symposium, Kraków, Poland (May 2012) 81–84
7. Li, Y., Sjostrom, M., Jennehag, U., Olsson, R.: A scalable coding approach for high quality depth image compression. In: 3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video. (October 2012) 1 –4

8. Chen, J., Ye, F., Di, J., Liu, C., Men, A.: Depth map compression via edge-based inpainting. In: Picture Coding Symposium, Kraków, Poland (May 2012) 57–60
9. Morvan, Y., de With, P.H.N., Farin, D.: Platelet-based coding of depth maps for the transmission of multiview images. In Woods, A.J., Dodgson, N.A., Merritt, J.O., Bolas, M.T., McDowall, I.E., eds.: Proceedings of SPIE: Stereoscopic Displays and Applications. Volume 6055., San Jose, California, USA (January 2006)
10. Sarkis, M., Zia, W., Diepold, K.: Fast depth map compression and meshing with compressed tritree. In Zha, H., Taniguchi, R., Maybank, S., eds.: Asian Conference on Computer Vision. Lecture Notes in Computer Science, Xi'an, China (September 2009) 44–55
11. Jager, F.: Contour-based segmentation and coding for depth map compression. In: Visual Communications and Image Processing, Tainan City, Taiwan (November 2011) 1–4
12. Zanuttigh, P., Cortelazzo, G.M.: Compression of depth information for 3D rendering. In: 3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video, Potsdam, Germany (May 2009) 1–4
13. Pennebaker, W.B., Mitchell, J.L.: JPEG: Still Image Data Compression Standard. Springer, New York (1992)
14. Taubman, D.S., Marcellin, M.W., eds.: JPEG 2000: Image Compression Fundamentals, Standards and Practice. Kluwer, Boston (2002)
15. Nguyen, T., Marpe, D.: Performance analysis of HEVC-based intra coding for still image compression. In Domanski, M., Grajek, T., Karwowski, D., Stasinski, R., eds.: Picture Coding Symposium, Kraków, Poland (May 2012) 233–236
16. Iijima, T.: Basic theory on normalization of pattern (in case of typical one-dimensional pattern). Bulletin of the Electrotechnical Laboratory **26** (1962) 368–388 In Japanese.
17. Morton, K.W., Mayers, L.M.: Numerical Solution of Partial Differential Equations. Cambridge University Press, Cambridge, UK (1994)
18. Mainberger, M., Hoffmann, S., Weickert, J., Tang, C.H., Johannsen, D., Neumann, F., Doerr, B.: Optimising spatial and tonal data for homogeneous diffusion inpainting. In Bruckstein, A.M., ter Haar Romeny, B., Bronstein, A.M., Bronstein, M.M., eds.: Scale Space and Variational Methods in Computer Vision. Volume 6667 of Lecture Notes in Computer Science. Springer, Berlin (June 2012) 26–37
19. Grewenig, S., Weickert, J., Bruhn, A.: From box filtering to fast explicit diffusion. In Goesele, M., Roth, S., Kuijper, A., Schiele, B., Schindler, K., eds.: Pattern Recognition. Volume 6376 of Lecture Notes in Computer Science. Springer, Berlin (2010) 533–542
20. Joint Bi-level Image Experts Group: Information technology – progressive lossy/lossless coding of bi-level images. ISO/IEC JTC1 11544, ITU-T Rec. T.82 (1993) Final Committee Draft 11544.
21. Mahoney, M.: Adaptive weighing of context models for lossless data compression. Technical Report CS-2005-16, Florida Institute of Technology, Melbourne, Florida (December 2005)
22. Chang, H.C., Wang, L.C.: A simple proof of thue's theorem on circle packing. ArXiv e-prints (September 2010)
23. Zitnick, C.L., Kang, S.B., Uyttendaele, M., Winder, S.A.J., Szeliski, R.: High-quality video view interpolation using a layered representation. In Hart, J.C., ed.: ACM Transactions on Graphics. Volume 23., New York, USA (August 2004) 600–608

24. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: From error visibility to structural similarity. IEEE Transactions on Image Processing **13**(4) (April 2004) 600–612