DECODING COMPLEXITY CONSTRAINED RATE-DISTORTION OPTIMIZATION FOR THE COMPRESSION OF CONCENTRIC MOSAICS

Eswar Kalyan Vutukuri, Ingo Bauermann, and Eckehard Steinbach
Institute of Communication Networks, Media Technology Group,
Technische Universität München, Munich, Germany
vutukuri@mytum.de,{ingo.bauermann,eckehard.steinbach}@tum.de

ABSTRACT

In this paper, a quantitative measure of decoding complexity for compressed image based scene representations is defined and is used to optimize the rate-distortion trade off during encoding subject to a decoding complexity constraint. The image-based representation used in this work is the concentric mosaic. Due to the huge amount of image data to be managed and the demands on just-in-time decoding and rendering of the compressed bit-stream, challenges arise in using concentric mosaics on low-end devices and for streaming applications over the Internet or other low bit-rate networks. Our approach allows for maximizing quality with respect to the available computational resources at the decoder and the bit rate.

1. INTRODUCTION

Interactive applications traditionally use computer graphics approaches based on geometric modeling to create virtual worlds the user can navigate in and objects he or she has to interact with. Textures are mapped on surfaces and view dependent warping is used to generate novel views. Additionally, surface properties and special rendering techniques can be defined to achieve lighting effects like shadows and reflections. These modeling steps are tedious and time consuming tasks and realistic models can only be rendered in real time when optimized graphics hardware is available. Generally speaking, increasing realism in geometry-based computer graphics applications is accompanied by increasing computational complexity.

Since the introduction of the plenoptic function [2] in 1991, more and more research interest is focused on using photographs of the real world as a reference for view generation, e.g., [7], [8], [9]. The plenoptic function describes a scene completely including lighting and surface properties like refraction and reflection. For every possible viewing position in space (x,y,z), for every possible viewing direction (φ,ϕ), and for every wavelength λ one intensity sample is taken. If changes of the structure of the scene or the lighting over the time t are considered the plenoptic function becomes a seven dimensional description of a scene:

$$P_7 = P(x,y,z,\varphi,\phi,\lambda,t)$$

As it is not possible in practice to sample the plenoptic function in all seven dimensions, simplifications have to be made. For this, some degrees of freedom are often not considered, e.g., discrete wavelength sampling, a static scene and constant intensity along a light ray are assumed for light fields and the lumigraph [8],[9]. Furthermore, for concentric mosaics [1] the y dimension is not considered which constrains user navigation to a fixed height. The acquisition process is easy in this case. A standard video camera is moved on a circle permanently facing normal to its trajectory. Novel view generation consists of interpolating reference images taken from different positions on the camera path.

Though concentric mosaics are an easy and effective way of accomplishing virtual reality, the huge number of images to be kept and managed in the memory is a big challenge. As an example, the *classroom* concentric mosaic we use in this work has 1523 images of CIF size which occupy approximately 450 MB. In this context compression is a key issue for real-time rendering. Standard compression schemes like MPEG are not suitable because of stringent requirements on the just-in-time (JIT) decodability of the bit stream [4].

Still image compression schemes like JPEG would allow for random access, but do not exploit the high correlation between neighboring reference images.

To provide a good compression rate along with random access in the compressed bit-stream many compression schemes were proposed for concentric mosaics like the vector quantization based scheme

in [1], a modified MPEG scheme in [4] and the 3D wavelet based scheme proposed in [5]. However, a complex compression scheme would mean that the decoding process together with the rendering process would pose severe requirements on the capabilities of the end device. A compression scheme which allows for a trade off between the rate, distortion, and the decoding complexity would be an ideal candidate for compression and rendering of concentric mosaics. With such a scheme, bit-streams suitable for decoding and rendering on devices of different capabilities can be produced.

The remainder of the paper is organized as follows. The basic coding scheme we use in our work is explained in Section 2. Then, a quantitative measure for decoding complexity at the receiver is defined in Section 3. Rate-distortion optimization subject to a decoding complexity constraint is explained in Section 4. Experimental results are presented in Section 5.

2. BASIC CODING SCHEME

The basic building block of the proposed codec is the DCT. We perform DCT on 8x8 image blocks and incorporate full pel motion compensation on these blocks.

Motion Compensation

In our approach the motion compensation procedure is constrained to horizontal movement in the concentric mosaic video sequence. In general the motion vectors in a typical video sequence are two dimensional pointing in any direction in the image plane. Assuming a static scene for the concentric mosaic, motion is only due to the movement of the camera. The camera itself is constrained to move in the horizontal plane. Therefore, the motion vectors are assumed to be horizontal only.

Please note that the motion vectors in a concentric mosaic sequence are not all exactly horizontal as the camera motion contains both rotation and translation. However, as we show in this work, the restriction to horizontal motion vectors greatly reduces the decoding complexity without affecting the rate-distortion trade-off too much. Additionally, we constrain the motion vectors to be in the range of -1 to +6 pixels to allow 3-bit fixed length encoding of the motion vector fields. This range seems to be best for the specific angular distance (about 0.2°) between the reference images in our test sequence.

Residual error encoding

We use run-length coding with an end-of-block symbol (EOB) terminating the block for encoding the transformed and quantized DCT coefficients. The EOB allows for random access to the residual error as a specific block can be found without decoding other parts of the bit stream. The color planes (Cb and Cr) are not sub-sampled but a coarse quantization is used instead.

Another important feature of the concentric mosaic data is that there is no notion of time in the image dataset. As the compressed stream is not decoded sequentially we can choose the reference frame (GOP anchor) of one group of pictures (GOP) at any position. We selected a GOP size of 25 frames and the GOP anchor (the frame with all blocks in Intra mode) is in the middle of the GOP instead of being at the beginning as is the case in standard video coding.

3. QUANTIFYING THE DECODING COMPLEXITY

In order to optimize the compressed bit-stream in terms of rate, distortion and decoding complexity, a quantitative measure has to be defined for all the above three parameters. Quantifying rate and distortion is straight forward. Since the optimization algorithm is run on 8x8 blocks, we define the rate as the number of bits required to signal one 8x8 block (8x8 blocks in Y, Cb and Cr components together with the motion vector information). Distortion is defined as the sum of squared differences (SSD) of the block. In order to quantify the decoding-complexity, we have to look at the time spent in various stages of decoding. Since DCT is the basic building block of the proposed codec, it can be assumed that most of the decoding time is spent in the IDCT and the reconstruction steps (approximately 47.5 % of decoding time involves IDCT and reconstruction as shown in [6]). We define the decoding complexity of a given 8x8 block as *the total number of pixels that have to be decoded to reconstruct the current block completely.* This is

proportional to the total number of IDCT steps to be performed and also the reconstruction overhead. The rate, distortion and complexity of a given block in the m^{th} row, n^{th} column and in the p^{th} frame of the GOP are represented as

$$R(m,n,p),\ D(m,n,p) \text{ and } C(m,n,p)$$

from here onwards. Now, since we have the quantitative definitions of all the three parameters that are to be optimized, the next task is to identify suitable modes in which each block can be coded. Our codec uses 5 different modes:

3.1. Intra-mode

A block in this mode is coded without reference to any other block. Hence such a block can be decoded independently. From the above definition of complexity, the complexity of such a block is 64. This can be seen by the fact that there are 64 pixels in an 8x8 block and we need to decode the 64 pixels in order to decode the block completely. Hence, complexity of a block in Intra mode is given by

$$C(m,n,p) = Complexity\_Intra = 64$$

3.2. Inter-mode

A block in this mode is encoded with a motion vector referring to a block in the previous frame. The residual error is encoded in the intra mode as described above. Since the motion vector is constrained to be horizontal, it can be seen that a block in this mode refers to at most two different blocks in the previous frame as depicted in Figure 1. It is here that the assumption of horizontal motion vectors gives returns in terms of reduced complexity. If the motion vectors were not constrained to be horizontal, the number of blocks being referenced can be as large as 4. This means that to decode the current block we have to decode 4 previous blocks which in turn might be referring to other blocks. This leads to an explosion of complexity at the decoder. It has to be kept in mind that unlike normal video, all the previous blocks are not already decoded in case of concentric mosaics. Hence, when a given block is required to be decoded, all the previous blocks that are referenced by the motion vectors have to be decoded on demand (assuming that the previous blocks are not in cache).

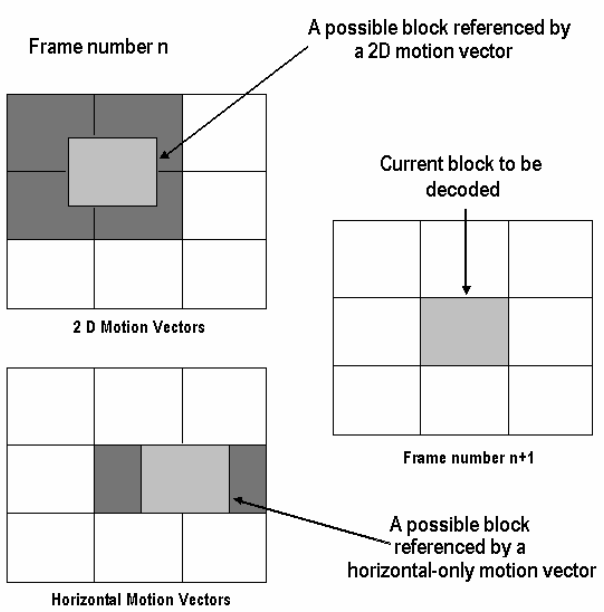

Figure 1: Referenced blocks in the previous frame for a typical video compression scheme (top) and in our case (bottom). The decoding complexity is reduced significantly for the 1D motion vector fields when further recursion is needed to decode the referenced blocks.

Hence, the horizontal motion vectors make the decoding a lot simpler. The complexity of a block in inter mode is hence given by the sum of the complexities of all the blocks that are to be decoded in turn. Hence, we obtain

$$C(m,n,p) = Complexity\_Inter =$$
$$= \begin{cases} 64 + C(m,n,p-1) + C(m,n+sign(mv),p-1), & mv \neq 0 \\ 64 + C(m,n,p-1), & mv = 0 \end{cases}$$

Where 64 is the complexity for decoding the residual error block that is in intra mode as explained in Section 3.1 and *C(m,n,p-1)* and *C(m,n+sign(mv),p-1)* are the complexities of the referenced blocks in the previous frame. "mv" is the motion vector of the current block. It should be noted that there are two referenced blocks only in the case when "mv" is non zero. This is a recursive formula which terminates only when all the referenced blocks are in intra-mode.

3.3. Anchor-referring inter-mode

This mode is similar to the mode in Section 3.2 except for the fact that here all the motion vectors refer to the GOP anchor instead of the previous frame. The advantage of this mode is that all the blocks in GOP anchor are coded in intra mode and the prediction process hence terminates after one step. Hence the complexity of the block in this mode is low. Since the GOP anchor is in the middle of the GOP, it is highly probable that many blocks in the frames at the middle of the GOP are encoded in this mode. This gives an additional choice for the mode decision process which is explained later. The complexity of a block in this mode is given by

$$C(m,n,p) = Complexity\_AR\_Inter =$$
$$= \begin{cases} 64 + C(m,n,13) + C(m,n+sign(mv),13), mv \neq 0 \\ 64 + C(m,n,13), mv = 0 \end{cases}$$
$$= \begin{cases} 64+64+64 = 192, mv \neq 0 \\ 64+64 = 128, mv = 0 \end{cases}$$

In the above formula we use 13 instead of the frame number within a GOP because in our work, the GOP anchor is always the 13th frame in a GOP of 25 frames. Moreover since all the blocks in the GOP anchor are in intra mode, the complexities are replaced by 64 as explained in Section 3.1. The first 64 is the complexity of the residual error which is also in intra mode. Hence the total complexity becomes 3x64 = 192 for the case when mv ≠ 0 and 128 when mv = 0.

3.4. Skip-mode

This mode is also similar to the inter mode explained in Section 3.2. However, the residual error is not encoded in this case. A block in this mode has only a motion vector referring to a block in the previous frame. This mode is highly efficient in terms of rate. We define the complexity of a block in this mode as

$$C(m,n,p) = Complexity\_Skip =$$
$$= \begin{cases} C(m,n,p-1) + C(m,n+sign(mv),p-1), mv \neq 0 \\ C(m,n,p-1), mv = 0 \end{cases}$$

3.5. Anchor-referring skip-mode

This mode is similar to the skip-mode defined in Section 3.4. However, in this mode the motion vector refers to blocks in the GOP anchor frame, instead of the previous frame. The residual error is not encoded as in the skip mode. This mode is very efficient both in terms of rate and complexity. The complexity of a block in this mode is given by

$$C(m,n,p) = Complexity\_AR\_Skip =$$
$$= \begin{cases} C(m,n,13) + C(m,n+sign(mv),13), mv \neq 0 \\ C(m,n,13), mv = 0 \end{cases}$$
$$= \begin{cases} 64+64 = 128, mv \neq 0 \\ 64, mv = 0 \end{cases}$$

Till now we have defined the complexities of the blocks in various modes. The rate and distortion have not been explicitly defined because their definitions do not depend on the mode definition. As mentioned earlier, the rate is defined as the number of bits required to signal an 8x8 block together with the motion vector (if any). We use Huffman coding of the run length pairs along with an end of block symbol as in JPEG. The Y, Cb and Cr components are written into three separate bit-streams. This allows easy access of the required block in the bit-stream. The mode and the motion vector are coded together using a fixed number of bits (6 bits for mode and motion vector together). This data is written into a separate stream. Since this is a fixed length code, random access at the floor level is guaranteed within this bit-stream. All the rate distortion plots shown in Section 5 have been generated with these settings.

4. RATE-DISTORTION OPTIMIZATION WITH DECODING COMPLEXITY CONSTRAINED

Once the rate, distortion and complexity of a given block in all the above modes are known, the encoder can perform the rate distortion optimization subject to a complexity constraint. It should be noted that the main idea in our work is to provide a bit-stream in which every block is guaranteed to have a decoding complexity which is at most equal to a fixed value

which we call the *maximum-allowed-complexity* of a block. This is a parameter to the encoder. Such a complexity constraint is useful for example in guaranteeing that a fixed number of stitched views per second can always be rendered on a given end-device. This is analogous to the number of frames per second decodable in a video sequence. If a complexity constraint is not imposed on the compression scheme, some areas in the concentric mosaic will be having a prediction process that is too deep and while viewing the concentric mosaics on a low-end-device the viewer has to spend an annoyingly long waiting time to get each view. While encoding, each frame of the image data is encoded in all the 5 modes and then the mode decision mechanism goes through each block and minimizes the following Lagrangian cost function over the 5 different modes mentioned in Section 3.

$$J = D(m,n,p) + \lambda_1 \times R(m,n,p) + K \times C(m,n,p)$$

Where D is the SSD of the block, R is the rate in bits per block and C is the complexity of the block. λ_1 is a parameter that is used to select the tradeoff between rate and distortion. K is the parameter that controls the mode decision process so that all the modes in which the current block has an over all complexity greater than the parameter *maximum-allowed-complexity* are eliminated from the mode decision process. In other words, K is set to zero if the complexity of a block in a given mode falls below the complexity constraint and it is set to infinity if the complexity of the block in a given mode exceeds this threshold. This ensures that the resulting bit-stream is optimized in terms of rate and distortion but the optimization is done only among the modes which are satisfying the complexity constraint.

5. EXPERIMENTAL RESULTS

In this section we show the rate distortion plots for various values of the complexity constraint. Figure 2 shows the curves for a complexity constraint varying from 250 till having no complexity constraint.

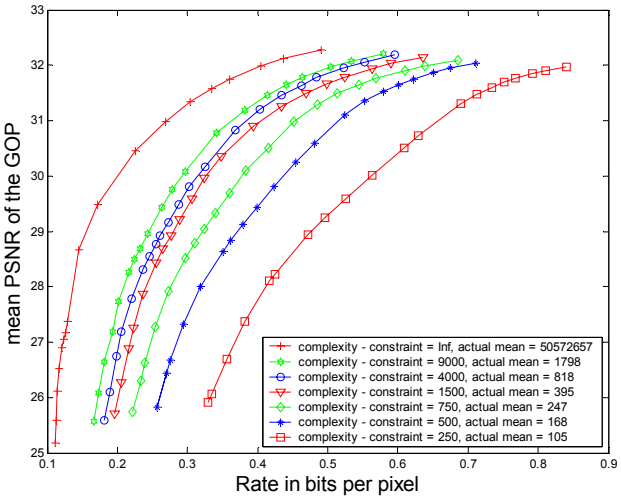

Figure 2: Operational rate-distortion curves with decoding-complexity constraint and actual mean complexity for the compressed bitstream.

The difference between the curve with complexity 250 and the curve having no complexity constraint at a bitrate of 0.4 bits/pixel is almost 4 dB. It should be noted that the complexity constraint defines the maximum number of pixels allowed to be decoded to decode the current block. However, the average value of the actual complexity might be far lower than the constraint. Hence, we measured the actual average value of the decoding complexity and showed it in the legend as actual mean complexity for each rate-distortion curve.

In order to verify that the complexity measure adopted in our codec actually represents the decoding complexity, we have measured the time taken to decode parts of the images in a GOP and stitch a virtual view which is the actual thing done by the renderer. We use a Matlab prototype program to run our simulation. The decoder program is passed with image numbers and column numbers in the images to be stitched together to form a virtual view. The decoder then decodes the required parts of the images from scratch (assuming no part of the required block has already been decoded or cached) and forms a virtual view. The times taken to decode and stitch one virtual view are given in Table 1.

Complexity Constraint	Time to decode and stitch a view in sec
250	8.437
500	21.188
750	73.547
1500	203.359
4000	745.515
9000	1655.907
No constraint	124026.233

Table 1: Decoding time for the prototype implementation for different complexity settings.

It can be seen from the above table that a bit-stream encoded with a low value for the complexity constraint is easy to decode, hence proving the validity of our complexity measure. An example virtual view stitched from the *classroom* concentric mosaic is shown in Figure 3.

6. CONCLUSION AND FUTURE WORK

In this paper we have proposed a possible way of achieving rate-distortion optimization subject to a constraint on decoding complexity for the compression of concentric mosaics. We showed that there is a trade-off between the rate-distortion performance and the decoding complexity. A quantitative measure for the decoding complexity has been developed and it has been shown that the complexity measure is valid.

It has been assumed that the caching at the decoder is turned off for deriving the complexity measure. A probabilistic model for the decoding complexity in presence of caching can be developed and the proposed idea of rate distortion optimization with complexity constraint can then be extended to the case when the decoder has a cache of fixed size. This would be a possible extension to the work presented in this paper.

We have performed rate distortion optimization with a constraint on the maximum decoding complexity. Another approach would be to perform rate distortion complexity optimization using two lagrangian multipliers in the equation for the cost function and produce a bit-stream that is optimized with respect to rate, distortion as well as decoding complexity.

Figure 3: Example stitched view from the compressed dataset using 250 as maximum complexity and 30.5dB mean PSNR.

7. REFERENCES

[1] H. Shum and L. He, “Rendering with concentric mosaics,” *Computer Graphics (SIGGRAPH '99)*, pp. 299-306, Aug. 1999.

[2] E. H. Adelson and J. Bergen, “The plenoptic function and the elements of early vision,” *Computational Models of Visual Processing*, pp. 3–20, MIT Press, Cambridge, MA, 1991.

[3] Y. Wu, C. Zhang, J. Li and J. Xu, “Smart rebinning for compression of concentric mosaics,” *ACM Multimedia 2000,* Los Angeles, CA, pp. 201-209, Oct. 2000.

[4] C. Zhang and J. Li, “Compression and rendering of concentric mosaic scenery with reference block codec (RBC),” *IEEE/SPIE VCIP*, Perth, Australia, June 2000.

[5] L. Luo, Y. Wu, J. Li and Y. Zhang, "3D wavelet compression and progressive inverse wavelet synthesis rendering of concentric mosaics," *IEEE Trans. on Image Processing*, vol. 11, no. 7, Jul. 2002.

[6] Ketan Patel, Brian C. Smith, and Lawrence A. Rowe, “Performance of a Software MPEG Video Decoder,” *ACM Multimedia Conference*, Anaheim, CA, 1993.

[7] S. E. Chen, “QuickTime VR – An Image Based Approach to Virtual Environment Navigation,” *Proc. SIGGRAPH ‘95,* pp. 29-38, Aug 1995.

[8] M. Levoy and P. Hanrahan, “Light field rendering,”, *Proc. SIGGRAPH '96*, pp. 31–42, 1996.

[9] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen, “The lumigraph,” *Proc. SIGGRAPH’96*, pp. 43–54, 1996.