VIDEO CODING WITH SPATIO-TEMPORAL TEXTURE SYNTHESIS

Chunbo Zhu[1], Xiaoyan Sun[2], Feng Wu[2], and Houqiang Li[1]*

[1]University of Science and Technology of China, Hefei, 230027, China
[2]Microsoft Research Asia, Beijing, 100080, China

ABSTRACT

This paper presents a video coding scheme in which some texture regions are selectively removed at the encoder and recovered by synthesis at the decoder. We present region removal utilizing conventional block-based motion information rather than global motion field. Removed regions including their motion information are not coded at the encoder. We propose spatio-temporal patch-searching in texture synthesis at the decoder to recover the removed regions. Our approach is not content based and is flexible and generic to be implemented. The scheme has been integrated into H.264/AVC and achieves up to 38.8% bitrate saving at similar visual quality levels compared with H.264/AVC.

1. INTRODUCTION

It is well accepted that motion compensation, intra prediction and transform are employed in video coding to exploit spatio-temporal redundancy based on the mean squared error (MSE) criterion. But the types of redundancies exploited by current video coding schemes are rather limited since they mainly focus on pixel-wise redundancy rather than perceptual redundancy. It is exemplified by the inefficiency of coding texture regions with many details, e.g. water and grass. However, texture synthesis [1] and image/video inpainting [2] have shown their effectiveness on dealing with these regions.

Recently, technologies in texture synthesis and image inpainting have shown remarkable progresses and have been developed for different purposes such as error concealment [3] and object removal [4]. In fact, advancements in texture synthesis are leading to promising efforts to exploit visual redundancy. It has been reported that improvement is achieved by employing texture synthesis in compression even though in a straight-forward fashion [3]. A content-based video compression scheme is also presented in [5] in which texture synthesis and analysis are utilized based on a global motion model. Moreover, since original video sequences are always available at encoder side, it is reasonable to assume that some assistant information can be extracted to empower texture synthesis as well as inpainting. Such kind of image compression method is developed in [6] in which edge-based inpainting approach is proposed to reduce the redundancy in structure regions.

In this paper, we focus on texture regions and aim to benefit from texture synthesis to improve current mainstream coding schemes, e.g. H.264/AVC. The basis of our approach is that some texture regions can be well synthesized from their spatio-temporal neighboring samples without visible quality loss. Therefore, they can be selectively removed during encoding.

The contribution of this paper lies in twofold. First, we propose region removal (also called exemplar selection in this paper) utilizing conventional block-based motion information rather than global motion field. Second, spatio-temporal texture synthesis algorithm is presented for region recovery in which spatial and temporal smoothness are simultaneously considered. Moreover, our approach is not content based and is developed at non-overlapped block level. It is flexible and generic to be implemented into standard-compliant video coding schemes.

The rest of this paper is organized as follows. In Section 2, the framework of our scheme is presented. In Section 3 and Section 4, we will describe the encoder and decoder algorithms in detail. In Section 5, experimental results are shown. We will conclude this paper in Section 6.

2. FRAMEWORK OF OUR PROPOSED SCHEME

In our proposed scheme, I-pictures and P-pictures, called key pictures, are coded with H.264/AVC (H.264 for short). B-pictures are candidates for partial texture synthesis. Each B-picture is divided into non-overlapped 8x8 blocks. Some of the blocks are removable at the encoder and can be well recovered at the decoder. Other blocks are called *exemplars* and will be utilized to synthesize the removed blocks at the decoder.

Fig. 1 shows the framework of our scheme. The input of the encoder is a group of pictures (GOP) which consists of a pair of key pictures and several B-pictures. 8x8 blocks of all B-pictures are categorized into structural ones and textural ones. Structural blocks (denoted by Str.) are regarded as

[*] This work has been done when the author was with Microsoft Research Asia.

exemplars and are coded with H.264 encoder. Textural blocks (denoted by Tex.) are input into the Exemplar Selection module. At the same time, 8x8-block motion threading [7] is performed on original B-pictures. The threads which consist of motion-aligned textural blocks are treated as 3-D exemplar candidates and are selectively preserved. Remaining blocks are totally removed. At the decoder side, key pictures and exemplars in B-pictures are reconstructed by H.264 decoder. Then partially reconstructed B-pictures are recovered by the Texture Synthesis module. After texture synthesis, completed pictures including key pictures are stored in the Picture Buffer so that they can be used to recover other pictures.

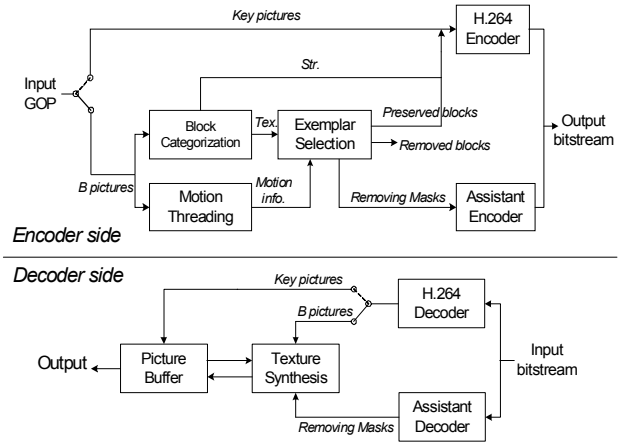

Fig. 1. Framework of our proposed scheme.

Note that, in our coding scheme, B-pictures are bi-directionally predicted from the nearest pair of key pictures which are coded and reconstructed with MSE criterion in H.264. In the following two sections, we will explain our encoder and decoder designs in detail.

3. ENCODER DESIGN

It is generally accepted that pure textures can be satisfactorily generated even given a small sample. With this knowledge, synthesis-based image compression technologies have been developed to exploit visual redundancy and have shown encouraging results [6]. In such image compression schemes, how to select removable regions and useful samples is significant to successful synthesis. This selection problem is also important and even more difficult for video applications, because temporal aliasing is typically much stronger than spatial aliasing in video sequences of dynamic scenes. The core of our encoder algorithm is the selection process which demands both spatial and temporal consistency. In the following of this section, we will explain how to select removable blocks and exemplars based on block categorization and motion information.

3.1. Block categorization

In our scheme, all B-pictures are input to the Block Categorization module shown in Fig. 1. A simple edge detector is used to detect edges in this module. The blocks which contain edge pixels are categorized into structural blocks. Remaining blocks are textural blocks. The textural blocks which are adjacent to structural blocks are called necessary textural exemplars and must be preserved and coded, because they contain the information of transition between different texture regions. Remaining textural blocks are called additional textural blocks, from which the removable blocks will be selected.

3.2. Motion threading

To avoid possible temporal inconsistencies of the synthesized result, we have considered motion estimation in selecting exemplars at the encoder side. We treat temporally sequential blocks on a motion trajectory as a candidate 3-D exemplar. In this way, the exemplar selection can be performed on a more global level to help synthesis keep spatial and temporal consistency.

We perform block-based backward motion estimation on original pictures in the Motion Threading module. Motion threading algorithm which was proposed in [7] is utilized so that additional textural blocks can be aligned in directions of different motion threads. In this step, we use conventional block-based motion estimation same as H.264/AVC with integer-pixel accuracy. Afterwards, motion threads in texture regions are pruned so that different threads will not overlap each other or fall into structure regions. Fig. 2 shows an example of motion threads within a GOP.

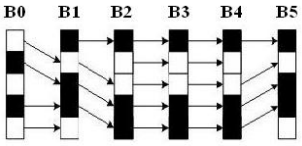

Fig. 2. An example of motion threads in one GOP. Shaded blocks indicate removed ones. Arrows indicate connection and direction within motion threads.

3.3. Exemplar selection

When motion threading is completed, all the threads are treated as 3-D exemplar candidates for exemplar selection. We arrange the average spatio-temporal variation (STV for short) of all the motion threads in descending order and choose those threads with higher variation according to a pre-defined ratio. The chosen blocks are preserved as exemplars and will then be coded with H.264 encoder. The remaining ones are totally removed and will be recovered at the decoder.

The average STV of a thread is defined as follows:

$$STV = \frac{1}{N}\sum_{i=1}^{N}[w_1\delta(B_i) + w_2 \sum_{B_j \in \mu_6(B_i)} |E(B_j) - E(B_i)|] \quad (1)$$

Here N represents the length of a corresponding thread, which consists of N blocks B_i (i=1...N). w_1 and w_2 in (1) are positive weighting factors. $\mu_6()$ indicates the spatio-temporal 6-neighboring (left, right, top, bottom, forward and backward) blocks of each block. The functions $\delta()$ and $E()$ are the variance and mean pixel value of a block.

Furthermore, because the variation is a local feature, removing large-scale regions should be avoided. Thus, we also check the connective degree of each block so that the removed blocks do not constitute a large region.

The output of the exemplar selection process includes a sequence of binary masks indicating which blocks are removed. The masks are then losslessly coded in the Assistant Encoder.

4. DECODER DESIGN

After decoding the binary masks which indicate the location of removed blocks, the decoder utilizes a patch-wise method to synthesize the removed texture regions. We borrow the main idea of patch-wise image inpainting [4] and extend it to our video coding application.

One direct approach to extend an image inpainting or texture synthesis method to video is to process each frame independently. But the temporal correlation among frames is neglected. On the one hand, a better matching patch may be found in temporally adjacent frames. On the other, to process video frames as independent images may ruin the temporal consistency and result in visible artifacts. For video texture synthesis, one can extend 2-D patches to 3-D volumes [8] but will introduce large amounts of data and computing complexity. In our synthesis scheme, we jointly utilize temporal and spatial reconstructed pixels to perform the synthesis process with 2-D patches.

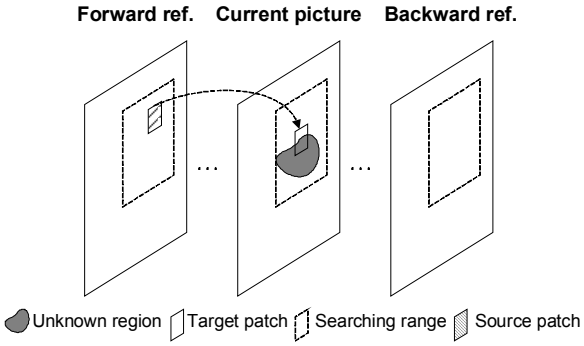

Fig. 3. Patch-wise texture synthesis in our approach.

Unlike the removing process at the encoder, the Texture Synthesis module at the decoder is designed for arbitrary-shaped regions and is performed frame by frame. We choose square patches as the fundamental elements while a confidence map [4] is introduced to guide the order of synthesis. As shown in Fig. 3, for each patch centered at a marginal pixel of unknown regions (denoted by target patch), we calculate the average confidence value of all pixels in this patch. Then the patch with the highest confidence will be synthesized first. Afterwards, a candidate patch which is most similar to the target patch is searched out in a certain spatio-temporal range centered at the target patch among the current picture, forward reference picture and backward reference picture.

The similarity between a candidate patch and a target patch is measured by S defined as follows:

$$S = SSD(W_t, W_c) + \alpha \cdot SSD(W_{t'}, W_c) \quad (2)$$

Where $SSD()$ denotes the sum of squared difference of known pixels between two patches. W_t and W_c represent the target patch and the candidate patch. $W_{t'}$ represents the patch which has the same location as the target patch in corresponding reference frame. α is a positive constant which controls the tradeoff between spatial and temporal smoothness. A patch that results in the least S will then be chosen as a source patch, which is often found in a temporally adjacent location. Then the source patch is merged into the target patch [9], [10]. After one patch is restored, the confidence map is updated. All the above operations are iterated until no unknown pixel exists.

5. EXPERIMENTAL RESULTS

We have integrated the proposed coding scheme into the H.264/AVC reference software JM10.2 [11] and have compressed the video sequences "Container", "Football" and "Bridge_far". In our experiments, we use YUV 4:2:0 sequence format under CIF resolution. All sequences are tested with 30Hz frame rate. Only one intra frame is coded for each sequence. The intra frame and all P-frames are key frames and are coded with H.264 for both testing cases. B-frames are periodically inserted between key frames and are predicted from the nearest pair of key frames. 15 B-frames are inserted in one GOP for "Container" and "Bridge_far". 4 B-frames are inserted in one GOP for "Football". Whole sequences are coded. Besides, rate distortion optimization (RDO) is turned on. CABAC is used for entropy coding.

Fig. 4 shows the reconstructed GOP of the sequence "Football". Compared with the reconstruction of H.264/AVC, our approach has a similar visual quality level. Fig. 5 shows bitrate savings of our approach for all testing sequences under different quantization parameters (QP). Noticeable bitrate savings of our scheme can be seen in this figure, up to 38.8% for sequence "Bridge_far". Note that, coded bitrates of binary masks have been included in the overall bitrates for our approach, although they are negligible.

Fig. 4. Four B-pictures in one GOP of sequence "*Football*": (a) original frames with removing masks (shaded blocks indicate motion-aligned removed ones), (b) reconstructions of our scheme with QP=18, (c) reconstructions of H.264/AVC with QP=18.

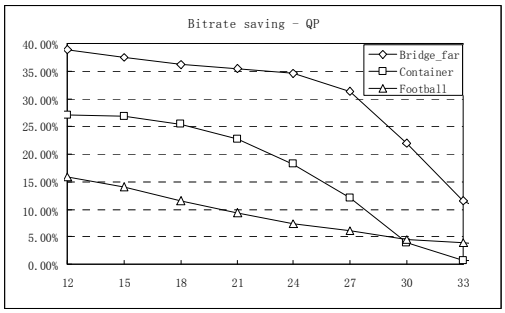

Fig. 5. Bitrate saving vs. QP curves.

6. CONCLUDING REMARKS

We present a video coding scheme based on spatio-temporal texture synthesis. The encoder removes visually unimportant blocks according to block categorization and block-based motion estimation. Remaining parts of video sequences are coded with H.264/AVC. The decoder recovers the removed blocks with patch-wise spatio-temporal texture synthesis. Inspired by image/video inpainting technologies, we introduced new patch-matching and filling-in algorithms to keep spatial and temporal consistency of video sequences.

ACKNOWLEDGEMENTS

This work is supported by NSFC General Program under contract No. 60672161, 863 Program under contract No. 2006AA01Z317, and NSFC Key Program under contract No. 60632040. Also, we would like to thank Dong Liu for his contributions and useful advices on this work.

REFERENCES

[1] A.A. Efros, and T.K. Leung, "Texture synthesis by non-parametric sampling," in *Proceedings of International Conference on Computer Vision*, pp. 1033-1038, 1999.

[2] M. Bertalmio, A.L. Bertozzi, and G. Sapiro, "Navier-stokes, fluid dynamics, and image and video inpainting," in *Proceedings of International Conference on Computer Vision and Pattern Recongnition*, vol. 1, pp. 355-362, Hawaii, 2001.

[3] S.D. Rane, G. Sapiro, and M. Bertalmio, "Structure and texture filling-in of missing image blocks in wireless transmission and compression applications," *IEEE Trans. Image Processing*, vol.12, no. 3, pp. 296-303, Mar. 2003.

[4] A. Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Transactions on Image Processing*, Vol. 13, no. 9, Sept. 2004.

[5] P. Ndjiki-Nya, T. Hinz, A. Smolic and T. Wiegand, "A generic and automatic content-based approach for improved H.264/MPEG-AVC video coding," *IEEE International Conference on Image Processing*, vol. 2, pp. 874-877, Sept. 2005.

[6] C. Wang, X. Sun, F. Wu, and H. Xiong, "Image compression with structure-aware inpainting," in Proc. *International Symposium on Circuits and Systems 2006*, pp. 1816-1819.

[7] L. Luo, F. Wu, S.P. Li, Z.X. Xiong, and Z.Q. Zhuang, "Advanced motion threading for 3D wavelet video coding," *Signal Processing: Image Communication*, Vol. 19, Issue 7, pp. 601-616, Aug. 2004.

[8] Y. Wexler, E. Shechtman, and M. Irani, "Space-time video completion," in Proc. *Computer Vision and Pattern Recognition*, vol. 1, pp. 120-127, Jun. 2004.

[9] V. Kwatra, A.A. Schödl, I. Essa, G. Turk, and A. Bobick, "Graphcut textures: image and video texture synthesis using graph cuts," in Proc. *ACM SIGGRAPH 2003*, pp. 277-286.

[10]P. Pérez, M. Ganganet, and A. Blake, "Poisson image editing," in Proc. *ACM SIGGRAPH 2003*, pp. 313-318.

[11]The H.264/AVC Joint Model, version 10.2.