A NEW GENERIC TEXTURE SYNTHESIS APPROACH FOR ENHANCED H.264/MPEG4-AVC VIDEO CODING

Patrick Ndjiki-Nya, Christoph Stüber, and Thomas Wiegand

Image Communication Group, Image Processing Department
Fraunhofer Institute for Telecommunications - Heinrich-Hertz-Institut
{ndjiki,stueber,wiegand}@hhi.de

ABSTRACT

A new generic texture synthesis approach, which is inspired by the work of Kwatra et al. [K03], is presented in this paper. The approach is hierarchical, non-parametric, patch-based and for that applicable to a large class of spatio-temporal textures with and without local motion activity. In this work, it is shown, how the new texture synthesizer can be integrated into a content-based video coding framework. That is, the decoder reconstructs textures like water, grass, etc. that are usually very costly to encode. For that, the new texture synthesizer in conjunction with side information that is generated by the encoder is required. The reconstruction of above-mentioned textures at the decoder side basically corresponds to stuffing holes in a video sequence. Spurious edges are thereby avoided by using graph cuts to generate irregular contours at transitions between natural and synthetic textures and preferably place them (the contours) in high-frequency regions, where they are less visible.

I. INTRODUCTION

Content-based video coding approaches typically partition a sequence into coherent regions given spatial, temporal, or spatio-temporal homogeneity constraints. For that, a texture analysis module is required at the encoder. Homogeneous segments are typically described using compact attributes like color, texture or motion features, which are transmitted to the decoder as side information. The decoder calls the synthesis counterpart of the texture analyzer to regenerate aforesaid regions using the side information. Significant bit rate savings can be achieved with this approach, while preserving high visual quality of decoded data, as is shown in [NN03]. This paper focuses on the texture synthesis module, assuming the texture analysis issue to be solved.
Texture synthesis approaches can be divided into two categories: parametric and non-parametric. In both synthesis categories, the pdf of given texture examples is approximated and sampled to generate new, perceptually similar texture samples. The example textures are thereby assumed to be large enough to capture the statistics of the underlying infinite texture. Parametric synthesis approaches approximate the pdf using a compact model with a fixed parameter set. Such approaches entail helpful hints w.r.t. identification and recognition scenarios. Non-parametric synthesis approaches do not explicitly model the pdf of the texture examples, they rather directly match the neighborhood properties of the given sample or patch to synthesize with the example texture. One of the selected candidates, with perceptually similar neighborhood properties to the sample or patch to synthesize, is then chosen for synthesis (pdf sampling). Not only do non-parametric synthesis approaches yield better synthesis results than parametric algorithms, also can they be successfully applied to a much larger variety of textures [K03]. In terms of compactness of texture representation, non-parametric synthesis algorithms are typically less efficient than parametric ones. Thus, in the content-based video coding framework, parametric synthesis methods are usually used [DH04]. We will show, however, that non-parametric synthesizers can be used at the decoder to achieve good video quality.
The remainder of the paper is organized as follows. In Sec. II, the hierarchical texture synthesis approach will be presented in-depth. The integration of the synthesizer into an H.264/MPEG4-AVC [ITU03] video encoder, w.r.t. the GOP (Group of Pictures) structure and the side information generation, is described in Sec. II.A. In Sec. II.B, video decoding using the texture synthesis module is described. In Sec. III, the experimental results are presented.

II. HIERARCHICAL TEXTURE SYNTHESIZER

The hierarchical texture synthesis algorithm presented in this paper is a non-parametric approach inspired by the work by Kwatra et al. [K03]. In this paper, Kwatra's algorithm is improved and adapted to a content-based video coding scenario. By selecting a non-parametric synthesis approach as baseline solution for our video codec, we opt for good video quality at the decoder and accept the potential drawback of increased side information compared to parametric algorithms.

A. Encoder

At the encoder, homogeneous texture segments are identified by the texture analyzer, which generates a mask sequence to delimit the identified texture clusters.

Inhomogeneous texture areas are coded using H.264/MPEG4-AVC, while the others are synthesized at the decoder.
The masks are transmitted to the decoder by coding the synthesis macroblocks as skipped macroblocks [ITU03]. That is, no transformation coefficients, residual error, nor motion vectors are transmitted for these macroblocks. However, a special skipped mode (TS mode) must be defined to distinguish between the H.264/MPEG4-AVC skip and the texture synthesis skip. The TS mode is enabled with a few bits for each macroblock, which is reasonably compact side information. This is yet achieved by transferring higher computational complexity to the decoder.

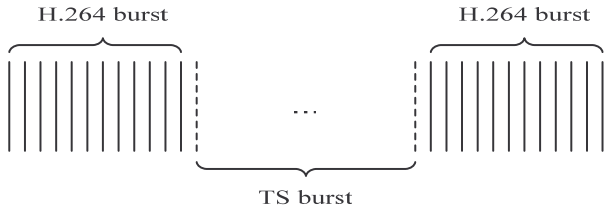

Figure 1: GoP structure of the proposed video coding framework

The GoP structure of the coding scenario presented in this paper is depicted in Figure 1. As can be seen, a burst of images coded using H.264/MPEG4-AVC is transmitted before and after the time interval that is to synthesize either partially or totally. The H.264/MPEG4-AVC coded frame bursts are used as texture examples for synthesis as is explained in the next section. Further eight motion parameters per burst frame are required at the decoder in case of a moving camera scenario as will be discussed in Sec. II.B.4.

B. Decoder

B.1. Filling a synthetic 3-D texture

The texture synthesis approach presented in this paper is a patch-based algorithm. That is, the synthetic texture is generated patch-wise in opposition to sample-wise synthesis methods [DB97].

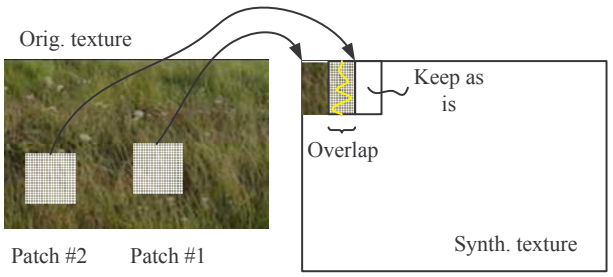

Figure 2: 2-D representation of synthetic texture filling

A patch can be seen as a cuboid that represents a sub-volume of the total 3-D spatio-temporal volume to synthesize. A typical patch size can be given as 32x32x20, where a reduced patch size of 20 frames is used in temporal direction to minimize the length of the required H.264/MPEG4-AVC burst (cp. Figure 1). The patches are inserted in an overlapping manner into the synthesis volume for reasons that will be explained in the next section. The overlap between the patches is typically chosen to be 16x16x10.
The first patch is selected at random from the example textures. The overlap volume (cp. Figure 2) is then matched with the texture examples using an adequate measure. In this work, the mean squared error (MSE) measure is applied to the luminance channel of both the overlap volume and the example texture at full resolution. The continuation patch is selected at one of the locations of the example texture minimizing the MSE. That is, to avoid monotony or exact reproduction of the original texture, one of the best candidates for continuation is selected at random for effective synthesis. This process is pursued until the synthetic texture volume is completely filled.

B.2. Patch adjustment using graph cuts

Once a continuation patch has been identified in the original texture, the probability of spurious edge occurrences at transitions between old and continuation (e.g. first and second) patch must be minimized. For that, an irregular shaped boundary is generated by using a graph cut approach [B99] (cp. Figure 2). The minimum cost path or cut from one end of the overlap volume to the other is determined. The cut thereby determines which patch contributes samples at different locations in the overlap region.

$$E = \frac{\|\mathbf{RGB}(p_{11}) - \mathbf{RGB}(p_{12})\| + \|\mathbf{RGB}(p_{21}) - \mathbf{RGB}(p_{22})\|}{\|\mathbf{g}_d(p_{11})\| + \|\mathbf{g}_d(p_{21})\| + \|\mathbf{g}_d(p_{12})\| + \|\mathbf{g}_d(p_{22})\|} \qquad (1)$$

A cut between any sample p_1 of the overlap volume and any adjacent sample p_2 yields to the costs E introduced by Kwatra et al. [K03] and formulated in (1). The numerator of (1) corresponds to the color difference between the sample pair p_1 and p_2.

Figure 3: Typical synthesis result given patch adjustment using graph cuts (right), example texture (left)

The difference is thereby measured in both old and continuation patch. p_{11} corresponds to sample p_1 in patch #1 (e.g. old patch), while p_{12} corresponds to sample p_1 in patch #2 (e.g. continuation patch). The same writing convention applies to sample p_2. **RGB**() corresponds to the RGB coordinates at the location

indicated in the brackets. The denominator of (1) takes into account that false boundaries are less visible in high frequency than in low frequency regions. This is done by determining the gradients $\mathbf{g}_d()$ at locations p_1 and p_2 along direction d, where d can be x, y or t. $\mathbf{g}_d()$ is measured in the luminance channel of the overlap volumes.

The determination of the optimal cut, given the overlap volume and the corresponding costs, is based on the algorithm by Boykov et al. [B99]. As a summary, it can be said that given E, the cut is typically chosen through high frequency regions, if any, to better dissimulate subjectively annoying discontinuities. A 2-D synthesis result using graph cuts for patch adjustment is depicted in Figure 3. The cuts generated at patch transitions are shown as white irregular lines in the synthesis texture.

B.3. Hierarchical texture synthesis

Although graph cut produces the best possible seam given two overlapping patches, visible artefacts can still be generated when no good match exists for the given overlapping region of the old patch. This can for instance be the case if the old patch was chosen at the edge of the example texture. For that, a new hierarchical texture synthesis approach is proposed that allows for implicit smoothing of disturbing seams. The key feature of our proposal is that good cuts are not smoothed, while bad cuts are. Note that Kwatra et al. [K03] use two different smoothing techniques, feathering and multiresolution splining, or do nothing in case the synthesis result is satisfactory. Which of the three options is used is decided using the trial and error method.

In the hierarchical texture synthesis approach, a Laplacian pyramid is built for each overlap region. The graph cut algorithm is first applied at the tip of the pyramid, that is, to the low frequency band of the signal. The synthetic texture, obtained after the cut, is filtered with a bandpass filter matching the frequency band of the current level of the pyramid. This is done to remove inappropriate frequencies that might have been introduced through synthesis.

Figure 4: Principle of hierarchical texture synthesis

This process is repeated at the downwards adjacent pyramid levels until the lowest level has been reached (cp. Figure 4). Note that the cut at a given pyramid level is constrained by the cut at the previous level. That is, cut variations in the current plane are forced to lie within the 8-neighborhood of the interpolated previous cut. This is done to minimize the effect of noise on the cut procedure. Finally, the synthetic overlap region is obtained by fusing the cut frequency bands according to the reconstruction rules of the Laplacian pyramid. Note that the approach presented in this paper can be used for 2-D textures as well as for video textures with local motion (water, smoke, etc.) or for stiff video textures (brick wall, flowers, etc.).

B.4. Temporal frame alignment

The algorithm presented up to this point works only given a static camera. It can be extended to a more generic approach by temporally aligning the H.264/MPEG4-AVC bursts. This can be done w.r.t. the median of the total time interval concerned by the synthesis including the two H.264/MPEG4-AVC bursts. The original frames must be used at the encoder for this operation. For this purpose, an approach based on dense motion fields is implemented. Robust statistics are operated on the estimated motion vectors to derive the apparent camera motion and compensate it.

Let $F_{t_{ref}}$ and $F_{t_{ref}\pm\alpha}$ be two frames of a video sequence. Then the dense motion field between the two is first estimated using the approach by Black and Anandan [BA96]. The samples belonging to background, i.e. regions without local motion activity, and thus underlying only global camera motion are determined using robust statistics, namely M-estimation [O04]. The latter is an iterative model-fitting approach that detects outliers (non-background samples) within a dataset a posteriori and without any prior knowledge of outlier characteristics. The observations are a set of motion vectors in our specific framework, while outliers can be seen as motion vectors that reveal different motion properties than the inliers. Motion homogeneity is defined w.r.t. the perspective motion model [O04]. That is, the observed motion field [BA96] is approximated using above-mentioned model. The perspective motion model (2) was selected due to its ability to describe translation, rotation, and scaling of a planar patch in 3-D as we assume this geometry for our synthesis textures. In (2), (x',y') represent the warped coordinates of the original sample (x,y), while $a_1,\ldots,a_8$ are the eight model parameters.

$$\begin{aligned} x' &= [(a_1 + a_3 x + a_4 y)/(1 + a_7 x + a_8 y)] \\ y' &= [(a_2 + a_5 x + a_6 y)/(1 + a_7 x + a_8 y)] \end{aligned} \qquad (2)$$

The M-estimator minimizes the influence of outliers on the model optimization by penalizing motion vectors yielding high modeling costs. The cost function is thereby defined as the deviation between the observed [BA96] and the modelled (2) dense motion field as shown in (3).

$$E_M = |v_x - \omega_x| + |v_y - \omega_y| \qquad (3)$$

In (3), (v_x,v_y) represent the observed, while (ω_x,ω_y) are the modelled motion vectors. After the global motion parameters $a_1,\ldots,a_8$ have been determined through M-estimation, $F_{t_{ref}\pm\alpha}$ is warped towards $F_{t_{ref}}$ achieving temporal alignment in that way. The range of α ($\alpha \in \mathbf{N}$) depends on the type of camera operations in the video sequence: Large α ranges can be used for slow global motion, while smaller ranges must be used for fast motion.

After the alignment has been done, the texture synthesis is operated as described in the previous sections, which results in a synthetic texture w.r.t. t_{ref}. That is, each synthesized frame has to be warped towards the genuine time instant by using the inverse mapping of (2), which can be easily derived from the same equation. The temporal alignment, required in a moving camera scenario, yields the necessity to transmit additional side information, i.e. a motion parameter set $a_1,\ldots,a_8$ for each frame of the H.264/MPEG4-AVC burst.

III. EXPERIMENTAL RESULTS

The experiments are conducted with three test sequences. Two of these correspond to a static camera scenario, while the third corresponds to a moving camera situation. A key frame of each sequence is depicted in Figure 5. The two static camera sequences, ocean and grass, are synthesized using the hierarchical texture synthesis approach presented in this paper. The results are available under
http://ip.hhi.de/imagecom_G1/HierarchicalTS.htm.

Figure 5: Key frames of the test sequences ocean (top left), grass (top right), and coast guard (bottom)

For the evaluation of the moving camera sequence, namely coast guard, the texture synthesizer is integrated into an H.264/MPEG4-AVC video codec. The following set-up is used for the codec. An unsynthesized burst of 21 frames is arranged at the beginning of the sequence, where the first frame of the burst is an I frame and the remaining frames are P frames. The 13 following GOPs which are composed of five B pictures and a P picture are partially synthesized (only water). One reference picture for each P picture, CABAC (entropy coding method), rate distortion optimization, 30 Hz progressive video at CIF resolution are also used. This setting results in a video sequence of a total length of 101 frames, as the video codec requires an additional P frame at the end of the sequence. Two configurations are used for the video codec without our approach. The first one is identical to the configuration described above, while the second one is entirely composed of GOPs of 5 B frames and a P frame.

It is found that better visual results can be achieved at the same bit rate (2661 kbps) for the H.264/MPEG4-AVC video codec with our approach compared to the codec without our algorithm. The decoding results can be down-loaded at the web-page mentioned above.

IV. CONCLUSIONS AND FUTURE WORK

In this paper, a hierarchical texture synthesis approach for content-based video coding is presented. It is shown that good video quality can be achieved at the decoder output for a large variety of video textures like water, grass etc. Compared to H.264/MPEG4-AVC, significant quality improvements can be achieved, at a constant bit rate, for video sequences containing such textures, as they are usually costly to code. The identification of continuation patches in the example texture will be improved to increase robustness of matches against noise. This can for instance be done by using a hierarchical matching procedure, instead of a full search at the highest signal resolution.

REFERENCES

[B99] Y. Boykov et al., “Fast Approximate Energy Minimization via Graph Cuts”, Proc. ICCV, p. 377-384, (1999).

[BA96] M. J. Black and P. Anandan, “The Robust Estimation of Multiple Motions: Parametric and Piecewise-smooth Flow Fields”, Computer Vision and Image Understanding, Vol. 63, No. 1, pp. 75-104, (1996).

[DB97] J. S. De Bonet, “Multiresolution Sampling Procedure for Analysis and Synthesis of Texture Images”, Proc. SIGGRAPH, p. 361-368, (1997).

[DH04] A. Dumitraş and B. G. Haskell, “An Encoder-Decoder Texture Replacement Method with Application to Content-based Movie Coding”, IEEE Trans. on CSVT, Vol. 14, No. 6, pp. 825-840, (2004).

[ITU03] ITU-T Rec. H.264 & ISO/IEC 14496-10 AVC: “Advanced Video Coding for Generic Audiovisual Services”, (2003).

[K03] V. Kwatra et al., “Graphcut Textures: Image and Video Synthesis using Graph Cuts”, Proc. SIGGRAPH, p. 277-286, (2003).

[NN03] P. Ndjiki-Nya, et al., “Improved H.264 Coding Using Texture Analysis and Synthesis”, Proc. ICIP, (2003).

[O04] J.-R. Ohm, “Multimedia Communication Technology”, ISBN 3-540-01249-4, Springer, Berlin Heidelberg New York, (2004).