AUTOMATIC DETECTION OF VIDEO SYNTHESIS RELATED ARTIFACTS

Patrick Ndjiki-Nya, Michael Kootz, and Thomas Wiegand

Fraunhofer Institute for Telecommunications – Heinrich-Hertz-Institut
Image Processing Department
Einsteinufer 37, 10587 Berlin, Germany
{ndjiki/kootz/wiegand}@hhi.de

ABSTRACT

An automatic approach for the detection of artifacts that may be introduced in video analysis/synthesis coding is presented. It is assumed that textures in a video scene can be classified into two categories: textures with unimportant subjective details and the remainder. We use this idea for video coding with a texture analyzer at the encoder and a texture synthesizer at the decoder. The analyzer identifies detail-irrelevant textures and generates side information for the synthesizer, which inserts synthetic textures at the specified locations. Our approach can be integrated into any video codec. This paper focuses on improvements of the texture synthesizer and the avoidance of potentially subjectively annoying spatial artifacts. The presented automatic detector yields an artifact identification rate of up to 89%. The former enables automatic online evaluation of the quality of synthesized frames, which allows eventual fallback on the reference video codec built on, for coding erroneous macroblocks.

1. INTRODUCTION

In video analysis/synthesis coding, we assume that textures with a large amount of visible detail, shown with limited spatial resolution, do not need to be reconstructed precisely by the decoder. The viewer should just be able to identify the displayed texture without subjectively noticeable artifacts. Compared to other textures such textures (e.g. water, grass, trees, sand …) typically are requiring comparably large bit rates to code when using mean squared error (MSE) as the distortion criterion. The above-mentioned idea can be basically applied to any video codec by introducing a texture analyzer at the encoder side and a texture synthesizer at the decoder side.

The texture analyzer identifies detail-irrelevant texture regions (water, sand …), creates coarse masks corresponding to these regions, and signals these masks as side information to the decoder. The texture synthesizer replaces the identified and signaled textures via inserting synthetic textures.

For the texture analyzer, similarity criteria like MPEG-7 descriptors [1],[2] can be used, instead of MSE as coding distortion to describe detail-irrelevant textures as shown in [3]. Since the considered MPEG-7 descriptors evaluate overall similarity, the reproduced textures typically show different details as the original ones. These deviations between original and synthetic textures are not subjectively noticeable as long as the displayed spatial accuracy of the textures remains unchanged and are also much less annoying as if they were coded at a bit-rate which is equivalent to the bit-rate of the side information of the texture synthesizer. In [3], it is shown that substantial bit-rate savings can be achieved using our approach. The gains thereby increase with increasing video quality. E.g., bit-rate savings of up to 19.4% compared to an H.264/AVC video codec were measured for the Flowergarden test sequence (CIF resolution, 30 Hz progressive video and quantization parameter 16).

A similar wavelet-based analysis/synthesis video coding approach was introduced by Yoon and Adelson [4] and by Dumitraş and Haskell [5]. The algorithms presented in [4],[5] are optimized for textures with no or very slow global motion, whereas no such constraint is required for our system [3].

In this paper, we focus on improvements of the texture synthesizer and the corresponding potentially subjective disturbing spatial artifacts. It is shown that an automatic detection of these artifacts can be achieved, which enables online quality assessment of the partially synthesized frames. In case of erroneous synthesis, the reference codec can be selected as fallback alternative for coding the corresponding macroblocks.

The remainder of the paper is organized as follows. In Section 2, we present the texture synthesizer, while in Section 3 the corresponding spatial artifact classes are addressed. In Section 4, the automatic artifact detection tool is presented. Finally, in Section 5 the experimental results are shown.

2. TEXTURE SYNTHESIZER

The texture synthesizer considered in this paper was designed for rigid objects. It is assumed that the frame-to-frame displacement of the objects can be described using the perspective motion model:

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

where (x',y') represent the warped coordinates of the original sample which has coordinates (x,y). $a_1,\ldots,a_8$ are the eight model parameters. Non-rigid textures like water can not be handled with this implementation of the texture synthesizer. However, a texture synthesizer for non-rigid textures only was already proposed in [3].

The texture synthesizer warps the considered texture from a given reference frame towards the texture region to synthesize as illustrated in Figure 1.

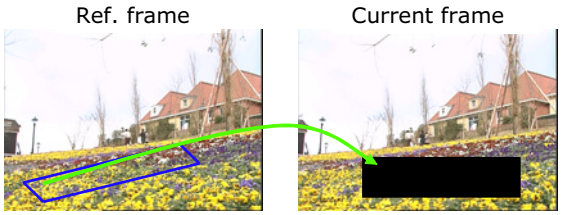

Figure 1 – Texture synthesizer filling texture region identified by texture analyzer using given reference frame

The detail-irrelevant texture region is thereby identified by the texture analyzer. A motion parameter set as well as a control parameter are required by the texture synthesizer for each synthesizable texture region. The control parameter indicates which reference frame to use to synthesize the current texture region. The motion parameters are necessary to transform the considered sample coordinates from one frame to the other, as explained above. Control and motion parameters are provided by the texture analyzer.

3. SPATIAL ARTIFACT CLASSES

It is obvious that, given a correct texture analysis within the selected detail-irrelevant regions, most spatial artifacts will occur at the transitions from synthesized to original textures in the form of false borders (cp. Figure 1). The main causes of the above-mentioned artifacts can be grouped into two classes:

- Suboptimal motion estimation,
- Texture region violation.

3.1. Suboptimal motion estimation

Suboptimal motion estimation occurs due to the limitations of the perspective motion model or/and the motion estimator [6],[7],[8] (cp. Figure 2).

The perspective motion model is suitable to describe arbitrary rigid object motion, if the camera operation is restricted to pure rotation and zoom. It is also suitable for rigid motion of planar objects with arbitrary camera operation. This assumption often holds approximately for the short-term motion estimation done within our coding scenario.

The considered motion model can not compensate camera lens distortion, which is usually no significant drawback for short-term motion estimation.

The perspective model leads to singularities, if the denominators of (1) are zero, i.e. if the camera rotation corresponds to ±90°. The content of the current image can not be projected onto the image plane of the selected reference image, using perspective transformation, in these cases. Due to extreme geometric distortion and to the fact that a pure rotation around the focal point seldom occurs in real world applications, the perspective model already leads to modeling problems for angles smaller than +90° or greater than –90° in practice.

Luminance is typically the observed quantity for motion estimation. It is assumed that luminance changes are exclusively due to motion. This approach is error prone if lighting conditions change or noise occurs in the video sequence. As a matter of fact, lighting variations and noise may also yield luminance variations independently of motion occurrence.

Occluded and uncovered regions also yield motion estimation problems, as in such cases the motion field is not defined.

3.2. Texture region violation

Texture region violation is given if parts of neighboring textures are assigned to the considered detail-irrelevant texture. This typically occurs at the borders of the detail-irrelevant texture, which may yield the fact that samples not belonging to the considered detail-irrelevant texture are picked from the reference frame to synthesize the corresponding texture (cp. Figure 2).

Figure 2 – Spatial artifacts due to suboptimal motion estimation (left) and texture region violation (right)

These artifacts are usually subjectively very annoying in case of significant dissimilarity between the detail-irrelevant texture and the neighboring one, even if the concerned image area is fairly small.

4. AUTOMATIC ARTIFACT DETECTION

The automatic detection of the spatial synthesis artifacts, described in the previous section, is a key issue for achieving online quality estimation capabilities. Online quality assessment can in turn help to switch between the reference codec and our texture analysis and synthesis, thereby ensuring good video quality at the decoder output.

4.1. Data selection and generation

Since the erroneous image material obtained from the selected test sequences was very limited, we reproduced the observed artifacts using images from Corel Gallery™ (US version, 07/1998).

45 color images were selected according to the textures they contained. We selected images with strongly (17 images), medium (15 images) and little (13 images)

texturized areas. These judgments were made based on subjective visual criteria.

For each image, artifacts corresponding to suboptimal motion estimation and texture region violation were simulated. The motion estimation artifact class was subdivided in two sub-classes reflecting typical observed artifacts. Each of the artifact (sub-)classes was subdivided in very, medium and little pronounced. This resulted in nine artifacts per image, which led to a database of 405 manipulated images. Both little pronounced artifacts corresponding to suboptimal motion estimation were subjectively not disturbing, so that we labeled the corresponding 90 manipulated images "non-erroneous synthesis". The remaining 315 images were labeled "erroneous synthesis".

4.2. Edge detectors

Two relatively simple linear anisotropic edge detectors, the Sobel and the Kirsch detector, are used for detecting subjectively disturbing edges. They are selected because they feature a single degree of freedom each, i.e. the sensitivity threshold, for a given edge directionality. They also differ in their respective motivations: The Kirsch edge detector represents an effort to model the type of gray level change near an edge, while the Sobel edge detector approximates the gradient of local luminance [9].

Only the vertical and horizontal directionalities of the detectors are used, since the synthesizable texture regions are composed of square macroblocks [10]. For each edge detector, the sensitivity threshold is varied and the optimal threshold determined for the given data set. I.e. edges that are not stronger than the selected threshold are ignored.

4.3. Quality measures

Reference edge masks showing the position of subjectively annoying edges were manually generated. These masks were then matched with the automatically generated edge masks.

We define true positives (TP) as being the edge samples detected in a given manipulated image which correspond to a subjectively disturbing edge. False positives (FP) are edge samples found in the corresponding original image without subjective disturbing edges. Both true and false positives were normalized with the total number of subjectively annoying samples in the reference edge mask.

Two performance measures were derived from TP and FP. The first measure is the "difference" measure, which is basically the difference between TP and FP. This measure allows the selection of a detector configuration w.r.t. TP and FP. The optimal value of this measure is 1.0, i.e. only true positives and no false positives are found. The smaller the difference is the poorer the edge detector configuration. Negative values of the difference measure mean that FP is greater than TP, which, of course, is not a good edge detection performance. We set negative values of the difference measure to zero, so that the latter is defined to lie in the interval [0,1].

The second measure corresponds to the ratio between TP and FP. This measure allows us to determine the critical ratio between TP and FP for the occurrence of subjectively annoying edges.

4.4. Selection of optimal edge detector configuration

The measures presented in the previous section can be used to determine the optimal edge detector configuration. The first step thereby consists in finding the optimal sensitivity threshold for each of the two considered edge detectors (cp. Section 4.2.). The ideal detector has a TP proportion of 1.0 and a FP proportion of 0.0. Thus the mean TP proportion was maximized and the mean FP proportion minimized for the 315 erroneous images. The variance of TP (FP) towards lower (higher) values was also minimized. These are, of course, contradictory requirements in practice. Hence they were prioritized as follows (decreasing relevance): mean FP, mean TP, variance. This corresponds to a rather conservative approach. The sensitivity threshold was varied with a step width of 0.002 for each edge detector. The optimal thresholds 0.022 and 0.058 were determined for Sobel and Kirsch respectively. Note that the optimal sensitivity threshold of a given edge detector depends on the degree of texturization of the erroneous test material. We considered the erroneous data irrespective of their content in order to achieve a single optimal edge detector configuration.

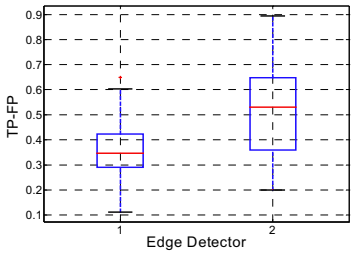

Figure 3 – Box plot of difference measure for the Sobel (1) and the Kirsch (2) edge detectors

Figure 3 depicts the difference measure values obtained for the edge detectors Sobel and Kirsch by comparing the erroneous data set and the corresponding original images. It can be seen that Kirsch leads to better results than Sobel with a mean difference measure value (median value, horizontal line within the corresponding boxes) of 0.53 vs. 0.35. Less than 25% of the test images yield a difference smaller than 0.35 for Kirsch (lower quartile corresponds to lower bound of the box), whereas a difference higher than 0.64 is obtained for more than 25% of the test images (upper quartile corresponds to upper bound of the box) for the same edge detector. The whiskers drawn from the lower (upper) quartile to the smallest (highest) difference cover the range of the data.

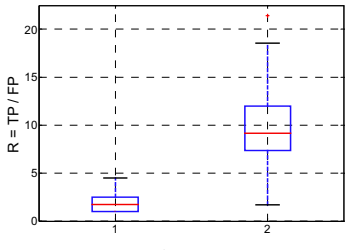

Figure 4 – Separability of non-erroneous (1) and erroneous (2) images using the Kirsch edge detector

The separability of erroneous and non-erroneous images (cp. Section 4.1.) using the optimal Kirsch configuration is shown in Figure 4. The former is determined based on the ratio between TP and FP (cp. Section 4.3.). Note that edge samples found in non-erroneous images were considered as TP here. It can be seen that both box plots overlap.

T_R	FR	FV
]0, 1.73[	[0.51, 1.0]	0
[1.73, 2.5[	[0.25, 0.51[	]0, 0.04[
[2.5, 4.5[	]0, 0.25[	[0.04, 0.11[
[4.5, ∞ [	0	[0.11, 1.0]

Tab. 1 – False rejection (FR) vs. false validation (FV) of partially synthesized images

The attempt to determine the critical ratio T_R for erroneous synthesis yields a trade-off between false rejection (FR) and false validation (FV). FR represents non-erroneous synthesized frames that have been classified as being erroneous by the edge detector, while FV specifies erroneous frames classified as being non-erroneous. Tab. 1 shows some FR and FV values for given T_R intervals. It can be seen that minimizing FV yields an increase of FR and vice versa.

5. EXPERIMENTAL RESULTS

In our experiments, we use artifacts generated by the texture synthesizer in nine test sequences (Canoe, Flowergarden, Concrete, Football, Formula 1, Husky, Raven, Stefan, Table Tennis) as validation set for the optimal Kirsch edge detector (cp. Section 4.4.). The artifacts were generated semi-automatically by manipulating the segmentation masks (cp. Section 1) manually so as to achieve artifacts of several severity levels. A total of 80 erroneous frames is considered and all artifact classes are represented. In addition to the erroneous frames, 80 non-erroneous frames are used. Erroneous as well as non-erroneous frames are coded using an H.264/AVC video codec. The following set-up was used for the H.264/AVC codec (Joint Model 2.1): IBBBBBPBBBBBP..., i.e., five B frames, one reference frame for each P frame, CABAC (entropy coding method), rate distortion optimization, 30Hz progressive video at CIF resolution. The quantization parameter was set to 16, which corresponds to quite good video quality.

Tab. 2 shows the results obtained for three T_R settings. The best FV rate (2%) was obtained for a ratio threshold (T_R) of very conservative 1.73 as expected.

T_R	FR	FV
1.73	0.49	0.02
2.5	0.19	0.11
4.5	0.05	0.39

Tab. 2 – Results for three ratio threshold settings

The former is achieved at the price of a very high FR rate (49%). For a ratio threshold of 4.5, the opposite scenario can be observed. The best compromise obviously lies between 1.73 and 2.5.

6. CONCLUSIONS

We have analyzed two methods for automatic detection of texture synthesis artifacts in video sequences. Based on a test set of 405 images it is found that the Kirsch edge detector leads to significantly better artifact identification rates than the Sobel edge detector. The Kirsch edge detector is validated using 160 additional validation frames and leads to an artifact identification rate of up to 89%. Further edge detectors will be explored in order to maximize the achievable discrimination performance on the test and validation sets.

7. REFERENCES

[1] ISO/IEC JTC1/SC29/WG11/N4362, "MPEG-7 Visual Part of eXperimentation Model Version 11.0", Sydney, Australia, July 2001.

[2] ISO/IEC JTC1/SC29/WG11/N4358, "Text of ISO/IEC 15938-3/FDIS Information technology – Multimedia content description interface – Part 3 Visual", Sydney, Australia, July 2001.

[3] P. Ndjiki-Nya, B. Makai, G. Blättermann, A. Smolic, H. Schwarz and T. Wiegand, "Improved H.264 Coding Using Texture Analysis and Synthesis", *Proc. ICIP 2003*, Barcelona, Spain, September 2003.

[4] S.-Y. Yoon and E. H. Adelson, "Subband texture synthesis for image coding", *Proc. SPIE on HVEI III*, Vol. 3299, pp. 489-497, San Jose, USA, January 1998.

[5] A. Dumitraş and B. G. Haskell, "An Encoder-Decoder Texture Replacement Method with Application to Content-Based Movie Coding", *To be published in IEEE Trans. on CSVT*.

[6] E. P. Simoncelli, "Distributed Representation and Analysis of Visual Motion", PhD Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, January 1993.

[7] H. S. Sawhney and R. Kumar, "True Multi-Image Alignment and its Application to Mosaicing and Lens Distortion Correction", *IEEE Trans. on PAMI*, Vol. 21, March 1999.

[8] R. Y. Tsai and T. S. Huang, "Estimation of Three-Dimensional Motion Parameters of a Rigid Planar Patch", *IEEE Trans. on ASSP*, Vol. 29, December 1981.

[9] J. R. Parker, "Algorithms for Image Processing and Computer Vision", *Wiley Computer Publishing*, USA, 1997.

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