DYNAMIC TEXTURE SYNTHESIS FOR H.264/AVC INTER CODING

Aleksandar Stojanovic, Mathias Wien, and Jens-Rainer Ohm

RWTH Aachen University
Aachen, Germany
Email: {stojanovic,wien,ohm}@ient.rwth-aachen.de

ABSTRACT

Dynamic textures are sequences of frames exhibiting certain stationarity properties over time; examples are sea-waves, whirlwind or moving crowds. We present an algorithm for dynamic texture extrapolation using only few training frames. A dynamic texture synthesizer using this algorithm has been integrated into a state-of-the-art H.264/AVC coding system, such that synthesized frames can be used by the encoder and decoder for inter prediction. For sequences not containing dynamic textures the same performance as with the conventional encoding system was achieved. In the case of sequences containing dynamic textures, intra coded macroblocks can be avoided by using the synthesized frame. Bitrate savings of up to 10 % have been observed experimentally.

Index Terms— Dynamic textures, H.264/AVC video coding, Inter frame prediction, System identification

1. INTRODUCTION

Dynamic texture is a major research topic in the field of computer vision. In [1], an algorithm for dynamic texture synthesis is presented, which is easy to implement and fast in computation. Many variations on this algorithm have been proposed, including dynamic texture modeling from Fourier descriptors [2], decomposition using a higher order SVD [3] or phase PCA [4].

Doretto et al. already mentioned the possible application of their algorithm in the field of video compression. Their idea was based on the fact that for long sequences, the total number of components of the model that are necessary for the re-creation of an approximation of the sequence is less than the total number of pixels in the sequence. Still the number of bits used to represent the different model components was not taken into account, which may be higher than the usual 8 bits for one pixel. Furthermore this compression concept can only be applied to sequences containing dynamic texture exclusively.

At the same time, many proposals have been made on how texture synthesis algorithms can be used in the field of video and image coding, especially by allowing intra prediction by template matching, like in [5] and [6], or motion estimation using template matching, see [7]. An approach classifying sequences into detail-relevant and detail-irrelevant textures is given in [8].

In this paper we propose to use an adapted dynamic texture extrapolation algorithm, in order to synthesize dynamic texture that can be used in a conventional coding system for prediction. This dynamic texture extrapolation algorithm is based on the representational dynamic texture model developed by Doretto et al. [1], where tools from system identification are used to learn dynamic texture models.

The rest of this paper is structured as follows. In Section 2, we present how we adapted the algorithm from [1] for the use in a conventional coding system. In Section 3, a description is given of how the synthesizer was integrated into H.264/AVC. Experimental results are reported in Section 4. Finally the paper concludes with Section 5.

2. DYNAMIC TEXTURE SYNTHESIS

2.1. General approach

Let $\{y(t)\}_{t=1\ldots\tau}$, $\{y(t)\} \in \mathbb{R}^m$ be a noisy video sequence, $y(t) = I(t) + w(t)$, where $w(t) \in \mathbb{R}^m$ is the noise in the sequence. Furthermore, let $y_\mathrm{m} \in \mathbb{R}^m$ be the temporal mean of the sequence and $y_\mathrm{d}(t) = y(t) - y_\mathrm{m}$. A dynamic texture sequence of frames $y(t)$ can then be modeled as an ARMA-process (autoregressive moving average):

$$\begin{cases} x(t) = Ax(t-1) + Bv(t) \\ y(t) = Cx(t) + y_\mathrm{m} + w(t), \end{cases} \tag{1}$$

with initial condition for the state vector $x(0) = x_0$, an unknown stationary distribution $q(\cdot)$ with $v(t) \overset{i.i.d.}{\sim} q(\cdot)$, and given noise $w(t) \overset{i.i.d.}{\sim} p_w(\cdot)$. In this case $y_\mathrm{d} = Cx(t)$. The main assumption leading to this representation is that individual frames in a sequence consisting of dynamic texture are realizations of the output of a dynamical system driven by an independent and identically distributed (i.i.d.) process.

In this model $A \in \mathbb{R}^{n\times n}$ is the state transition matrix, $C \in \mathbb{R}^{m\times n}$ the observation matrix, and n the order of the model and the dimension of the state vector $x(t)$.

(a) Frame number 9

(b) Extrapolated frame

(c) Frame number 10

Fig. 1. Dynamic Texture extrapolation example. Extrapolated frame from frames 5-9 in Foreman QCIF.

This model presented in [1] allows the extrapolation of a dynamic texture sequence to infinit length by driving the model with a noise process $v(t)$.

If we write a sequence in matrix form, each column in the sequence matrix Y is one frame of the sequence. The frames are converted by writing the pixels of the frames consecutively into one vector. For YUV sequences the two chroma components are appended, see [1] and [3]. $Y_k^l = [y(k), \ldots, y(l)] \in \mathbb{R}^{m \times (l-k+1)}$ denotes the part of the sequence containing the frames from k to l and m is the number of pixels per frame. Eq. 1 becomes

$$\begin{cases} X_1^\tau = AX_0^{\tau-1} + BV_0^{\tau-1} \\ Y_0^{\tau-1} = CX_0^{\tau-1} + Y_\mathrm{m} + W_0^{\tau-1}. \end{cases} \tag{2}$$

where $Y_\mathrm{m} \in \mathbb{R}^{m \times \tau}$ consists of τ equal columns y_m. A suboptimal but computationally very efficient method to determine the observation matrix C is computing a singular value decomposition

$$Y_\mathrm{d} = USD^\mathrm{T} \tag{3}$$

and set $C = U$ and $X = SD^\mathrm{T}$. The number of singular values is equal to the number of frames τ in the sequence. If singular values are omitted as in Doretto's approach, we have

$$Y = \tilde{C}\tilde{X} + Y_\mathrm{m} + W. \tag{4}$$

In fact Doretto et al. assume the order of the model n to be smaller than the number of training frames τ, so only n singular values are kept. Further a least squares approximation $\widehat{A}(\tau)$ can be found with:

$$\widehat{A}(\tau) = arg\min_A \|X_1^\tau - AX_0^{\tau-1}\|. \tag{5}$$

2.2. Extrapolation from very short training sequences

For our purpose, the extrapolation should be performed in a deterministic way, such that the same extrapolation is obtained at the encoder and decoder. In this case we can use the extrapolation result as a frame prediction in a conventional video coding system. This can be achieved by setting $V = 0$ in eq. 2. Furthermore, as we intend to perform an extrapolation from five frames only, no singular values should be omitted. We assume that in practice a model for dynamic texture should be of an order even higher than five.

Under the conditions just described, $V = 0$ and $W = 0$ (i.e. the order of the model $n = \tau = 5$), the extrapolated sequence is the repetition of the training sequence. The reason is that the mean of $y_\mathrm{d}(t)$ over time is zero. As a consequence, in our implementation we do not subtract the mean value from the frames, and perform a deterministic extrapolation generating new content obeying to the model.

As an example, we will now show how to obtain the extrapolated frame from the five first frames in a sequence using our algorithm. In particular, we compute:

$$Y_0^4 = CX_0^4 \tag{6}$$

using a SVD. Then

$$\widehat{A} = X_1^4 \left(X_0^3\right)^+ \tag{7}$$

where $\left(X_0^3\right)^+$ is the pseudo-inverse of X_0^3. Finally

$$\begin{cases} X_5^5 = \widehat{A}X_4^4 \\ Y_5^5 = CX_5^5. \end{cases} \tag{8}$$

In this case the extrapolated frame would be the vector Y_5^5.

Fig. 1 shows the result of the algorithm applied to the frames 5-9 from the Foreman QCIF sequence. As talking faces are regarded as dynamic texture, this example illustrates the performance of our algorithm on a macroscopic scale. The opening of the mouth is well extrapolated.

3. INTEGRATION INTO THE H.264/AVC CODING SYSTEM

Our goal is to integrate the extrapolation algorithm into H.264/AVC [9], in order to improve the coding efficiency of H.264/AVC for sequences containing dynamic texture. The basic idea is to create dynamic texture content that can be referenced in the encoding and decoding process. In particular, we extrapolate one frame from five frames in the decoded

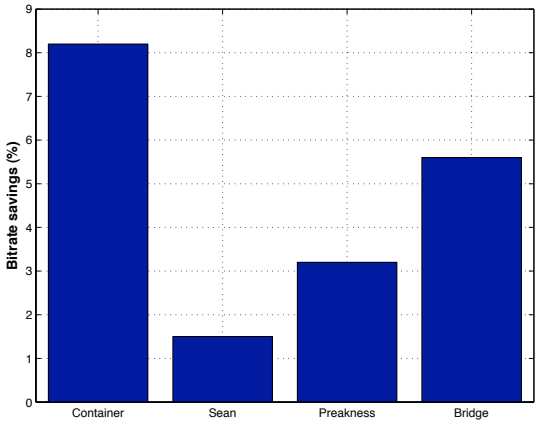

Fig. 2. Bitrate savings for sequences containing dynamic texture with QP = 23 and five reference frames.

picture buffer, and replace the oldest frame in the reference picture buffer with the extrapolated frame. The encoder, and hence the decoder, will be able to use our predicted frame as a reference frame. By this, dynamic texture can be described by simple motion compensation when the model extracted from the preceding frames matches the real dynamic texture. A conventional H.264/AVC encoder will have difficulties to describe the local dynamics occurring in dynamic textures, and will need to signal a residual or even encode some macroblocks in intra mode, which is very expensive in terms of bitrate.

The comparison in section 4 is done using the same number of reference frames, i.e. H.264/AVC with k reference frames is compared to the modified algorithm using $k-1$ reference frames plus the extrapolated frame. This comparison is suitable because we have a comparable motion information signalling overhead.

4. EXPERIMENTAL RESULTS

For the experimental investigations, we use the system described in Section 3 with various test sequences, all at QCIF format and 30Hz. The presented algorithm is integrated into the JM12.4 reference software [10]. The synthesized reference frame is extracted from the five previously decoded frames.

For comparison, testing conditions based upon [11] are used. In particular, we use QP values 22, 27 and 32 for the I frame, and QP values of 23, 28 and 33 for the P frames. The search region is 32 pixels, rate-distortion optimization is on, and the entropy coding mode is CABAC. The prediction structure is IPPP and as five frames are used for the extrapolation, five reference frames are used.

Fig.2 presents the bitrate savings achieved for QCIF sequences that include dynamic texture. These savings are achieved at an equal PSNR level (QP=23). It should be mentioned, that slightly better results are obtained for the Container QCIF sequence when using only $k = 3$ reference frames (10% bitrate savings). In Fig. 3 we can observe that our algorithm performs better for low QP values, i.e. in the high PSNR range. For sequences not containing dynamic textures, no difference in bitrate and PSNR is observed.

In Fig. 4, the bit usage distribution for P frames in H.264/AVC and in our algorithm is given for the example of the sequence Container QCIF with $k = 5$ reference frames resp. with $k - 1 = 4$ reference frames and the extrapolated frame. It reveals how the bitrate savings are achieved. Significantly less bitrate is spent on luma coefficients. In fact the extrapolated frame was referenced with a frequency of 22% on average. So the decoder can choose the predicted frame as reference, which happens frequently in case the sequence contains dynamic texture. The conventional encoder needs to encode some macroblocks in intra mode (or inter with a residual), as no similar reference image region is available. Here our encoder can predict from the synthesized frame and transmit the macroblock in inter mode. Example sequences to see where and how often the synthesized reference is used, can be viewed at our web-site: http://www.ient.rwth-aachen.de/∼stojanovic/ICIP08/.

One drawback of the algorithm is the increased encoder and decoder complexity. The absolute complexity increase is equal for both encoder and decoder, but the relative increase for the decoder is much more important. In our current non-optimized implementation, we observered a mean decoding time of about 60 ms per frame on a desktop PC (2.16 GHz), compared to 5 ms using the JM reference decoder for QCIF resolution. For this reason we suggest to use the synthesis only for frames containing dynamic textures. This decision can be taken for each frame individually. A straightforward criterion would be the frequency of referencing the synthesized frame.

5. CONCLUSION AND FUTURE WORK

In this paper we introduced an algorithm that combines a state-of-the-art video coding system with dynamic texture synthesis. The first very promising results achieved for QCIF sequences show that, in terms of rate-distortion ratio, the use of dynamic texture synthesis can provide improvements in compression performance for sequences that contain dynamic textures. This can be explained by the fact that fine-granular non-uniform motion is very expensive to signal using conventional block-based motion compensation, while our algorithm automatically extrapolates the movement from past frames. However, fine-granular motion is only transmitted in the case of high bit rate, which explains that we currently obtain bit rate savings only for low QP values, i.e. in the high PSNR range.

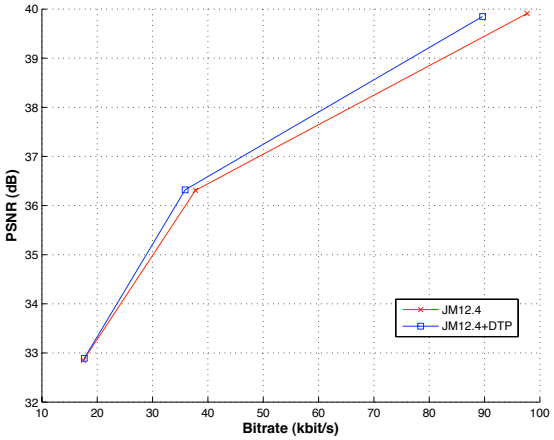

Fig. 3. Rate-distortion plot for Container QCIF at QP = 23, 28, 33 and with 5 reference frames comparing JM12.4 with and without the Dynamic Texture Prediction (DTP) algorithm.

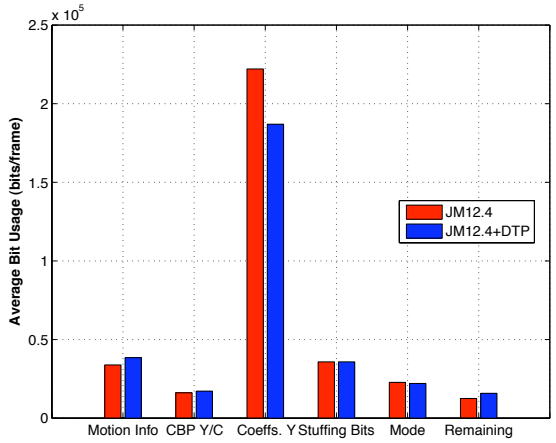

Fig. 4. Average bit usage in P frames for Container QCIF at QP = 23 and using five reference frames.

Further efforts will be made in order to decrease the computational complexity contained in the SVD which is applied globally to entire frames in the current implementation. For example, this inhibits usage of the algorithm for sequences with higher resolution. Localized application to only parts of the video frames is an envisaged option to achieve this. On one hand the algorithm could be executed in parallel, on the other hand this would give the possibility to use the extrapolation for only those parts of the frames which actually contain dynamic textures, and better adapt to their properties. Furthermore, we are planning to combine the algorithm with motion compensation, to enable proper modeling in cases of non-granular motion, caused by object or camera movements.

6. REFERENCES

[1] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto, "Dynamic textures," *International Journal of Computer Vision*, vol. 51, no. 2, pp. 91–109, 2003.

[2] B. Abraham, O. Camps, and M. Sznaier, "Dynamic texture with Fourier descriptors," in *Texture 2005: Proceedings of the 4th International Workshop on Texture Analysis and Synthesis*, September 2005, pp. 53–58.

[3] R. Costantini, L. Sbaiz, and S. Süsstrunk, "Higher Order SVD Analysis for Dynamic Texture Synthesis," *IEEE Transactions on Image Processing*, vol. 17, no. 1, pp. 42–52, 2008.

[4] B. Ghanem and N. Ahuja, "Phase PCA for Dynamic Texture Video Compression," in *Proceedings of the IEEE International Conference on Image Processing*, September 2007, vol. III, pp. 425–428.

[5] T. K. Tan, C.H. Boon, and Y. Suzuki, "Intra Prediction by Template Matching," in *Proceedings of the IEEE International Conference on Image Processing*, October 2006, pp. 1693–1696.

[6] J. Ballé and M. Wien, "Extended Texture Prediction for H.264/AVC Intra Coding," in *Proceedings of the IEEE International Conference on Image Processing*, September 2007, vol. VI, pp. 82–92.

[7] Y. Suzuki, C. S. Boon, and T. K. Tan, "Inter frame coding with template matching averaging," in *Proceedings of the IEEE International Conference on Image Processing*, September 2007, vol. III, pp. 409–412.

[8] P. Ndjiki-Nya, T. Hinz, A. Smolic, and T. Wiegand, "A generic and automatic content-based approach for improved H.264/MPEG4-AVC video coding," in *Proceedings of the IEEE International Conference on Image Processing*, September 2005, vol. II, pp. 874–7.

[9] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overwiev of the H.264/AVC Video Coding Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 291–294, July 2003.

[10] JM12.4: http://iphome.hhi.de/suehring/tml.

[11] T. K. Tan et al., "Recommended simulation common conditions for coding efficiency experiments," in *ITU-T / Study Group 16 / Question 6 - VCEG*. Document VCEG-AA10, October 2005.