QUADTREE-BASED TEMPORAL TRAJECTORY FILTERING

Marko Esche, Alexander Glantz, Andreas Krutz, Michael Tok, Thomas Sikora

Communication Systems Group, Technische Universität Berlin
Sekr. EN1, Einsteinufer 17, D-10587 Berlin, GERMANY
{esche, glantz, krutz, tok, sikora}@nue.tu-berlin.de

ABSTRACT

In both the HEVC draft and in H.264/AVC, in-loop filters are employed to improve the subjective and the objective quality of compressed video sequences. These filters use spatial information from a single frame only. Temporal Trajectory Filtering (TTF) constitutes an alternative approach which performs filtering in the temporal domain instead. In this work, a combination of the TTF with a quadtree partitioning algorithm for applying different filter parameters to different image regions is proposed and investigated. Experiments were conducted in the environment of the HEVC test model HM 3.0. Bit rate reductions of up to 9% for the low delay high efficiency setting of HEVC are reported.

***Index Terms*—** video compression, in-loop filtering, pixel trajectories

1. INTRODUCTION

In hybrid video codecs, block-artifacts are introduced at low bit rates due to the combination of motion compensated prediction and quantization of DCT coefficients [1]. Ever since the standardization of H.264/AVC, in-loop filters have been used to reduce these artifacts. They serve the dual purpose of subjectively improving the quality of the decoded frames and reducing the required bit rate through the use of objectively improved reference frames. In the draft description of the next-generation video codec High Efficiency Video Coding (HEVC) [2], three such in-loop filters are present. The first is the H.264/AVC deblocking filter (DF) as described by List et al. in [1] which performs one-dimensional filtering operations at block boundaries based on the boundary strength and the utilized prediction modes. The second is the Sample Adaptive Offset algorithm (SAO) [3]. This filter improves the quality of reconstructed frames by classifying each decoded pixel into a number of categories. According to the selected category, a simple offset is then added to the pixel. Average bit rate reductions of 2% produced by the SAO are reported for the low delay high efficiency setting of HEVC. The third filter is the Wiener-based Adaptive Loop Filter (ALF) which was first described by Wittmann et al. in [4]. The ALF applies a two-dimensional filter kernel of either 5×5, 7×7 or 9×9 pixels to certain image regions. Which pixels are to be processed by the ALF is signaled on coding unit (CU) basis, where the largest coding unit (LCU) roughly corresponds to the macroblocks used in H.264/AVC. None of the filters mentioned so far, however, perform filtering in the temporal domain and do not necessarily reduce the block flickering present at low bit rates. One possibility to perform temporal filtering instead, that utilizes block-based alignment, was detailed in [5]. In [6], the authors described a temporal filtering approach based on temporal pixel trajectories which requires three thresholds to perform effective filtering of both foreground objects and background. In this context, a pixel trajectory is interpreted as the locations through which a certain image point moves during a video sequence. The temporal trajectory filter (TTF) was previously tested in the context of the H.264/AVC baseline and extended profiles and was reported to produced average bit rate reductions of 3.6% and 2.0% for the two profiles respectively. In [7] additional modifications of the filter were introduced and first results in the context of the HEVC test model HM 3.0 were reported with an average bit rate reduction of 1.6% for the HEVC low delay high efficiency setting [2] for a different dataset. In this paper, the TTF is improved through the inclusion of a quadtree partitioning algorithm and the modified filter is investigated. The remainder of the paper is structured as follows. Section 2 reexamines some of the important properties of the TTF and provides a brief description of the previously conducted work. In Section 3, the quadtree partitioning algorithm and its rate-distortion (RD) optimization are discussed. The experimental setup and results produced by the algorithm are detailed in Section 4. Section 5 summarizes and concludes the paper.

2. TEMPORAL TRAJECTORY FILTERING

The underlying image model for the TTF can be described as follows. For simplification, it is initially assumed that the image content of a video sequence is only subject to translational motion. In this case, for every image point $(x_0, y_0)^T$ with a luminance value of $Y_j(x_0, y_0)$ in a given frame j, its location in $N-1$ previous frames can also be identified. These shall be denoted by $(x_i, y_i)^T, 0 < i < N$ with the associated luminance samples $Y_{j-i}(x_i, y_i)$. Even if the motion of every pixel is known, $\hat{Y}_{j-i}(x_i, y_i), 0 \leq i < N$ in the decoded sequence will not be identical to the original luma sample $Y_j(x_0, y_0)^T$ due to the noise introduced during the encoding process. Instead, every copy of the sample is distorted by a noise term that can be assumed to have properties similar to white noise [6].

$$\hat{Y}_{j-i}(x_i, y_i) = Y_j(x_0, y_0) + n_i, 0 \leq i < N. \tag{1}$$

In order to achieve noise reduction in reconstructed frames, the TTF calculates a weighted mean of the samples along a pixel's trajectory and replaces $\hat{Y}_{j-i}(x_i, y_i)$ with

$$Y_j^*(x_0, y_0) = \frac{1}{N} \sum_{i=0}^{N-1} \beta_i \cdot \hat{Y}_{j-i}(x_i, y_i). \tag{2}$$

Where the individual weight β_i associated with every sample $\hat{Y}_{j-i}(x_i, y_i)$ is inversely proportional to the variance σ_i^2 of the noise term n_i [7]

$$\beta_i = \frac{N/\sigma_i^2}{\sum_{k=0}^{N-1} \frac{1}{\sigma_k^2}}. \tag{3}$$

According to Wiegand and Girod [8], the noise variance in a reconstructed frame is given by

$$\sigma_i^2 = \frac{Q_{\text{step}}^2}{3} \text{ with } Q_{\text{step}} = 0.625 \cdot 2^{\frac{\text{QP}_i}{6}} \tag{4}$$

for a given quantization parameter QP_i. The optimal weight β depending on the frame QP can thus be computed

$$\beta(\text{QP}) = 3 \cdot \left(0.625 \cdot 2^{\frac{\text{QP}}{6}}\right)^{-2}. \tag{5}$$

To estimate the motion trajectory at the decoder the motion vectors conveyed in the bit stream are concatenated as long as the encoder deems them valid. Every pixel within a motion-compensated block is assumed to have the exact motion indicated by the associated motion vector, which yields two dense motion vector fields $(dx_{j,0}, dy_{j,0})^T$ and $(dx_{j,1}, dy_{j,1})^T$ for a given frame j for reference lists 0 and 1 respectively. Two possible new locations for a pixel $(x_0, y_0)^T$ in frame j are then for instance derived by

$$\begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} + \begin{pmatrix} dx_{j,0}(\lfloor x_0 \rfloor, \lfloor y_0 \rfloor) \\ dy_{j,0}(\lfloor x_0 \rfloor, \lfloor y_0 \rfloor) \end{pmatrix},$$
$$\begin{pmatrix} x_2 \\ y_2 \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} + \begin{pmatrix} dx_{j,1}(\lfloor x_0 \rfloor, \lfloor y_0 \rfloor) \\ dy_{j,1}(\lfloor x_0 \rfloor, \lfloor y_0 \rfloor) \end{pmatrix}. \tag{6}$$

As shown in Figure 1 the trajectory can theoretically split into two seperate paths at every B-frame. Especially in the case of the HEVC low delay setting with an IBBB coding structure, it becomes necessary to distinguish between those motion vectors that describe the true motion of a pixel and those that have purely been chosen due to RD-optimization. To this end the TTF uses three criteria with associated thresholds that are transmitted as side-information in the bit stream.

2.1. Absolute error along the trajectory

One possible indicator for a false motion trajectory is a sudden change in the color of the pixels along the trajectory. For every new pixel of the trajectory its luminance difference with its direct predecessor $\Delta Y_i = |Y_i - Y_{\text{pre}}|$ is calculated. In Figure 1 for example the predecessor of both Y_1 and Y_2 is the sample Y_0. The chrominance differences ΔU_i and ΔV_i are calculated similarly. A luminance sample is only included in the filtering process, if it satisfies

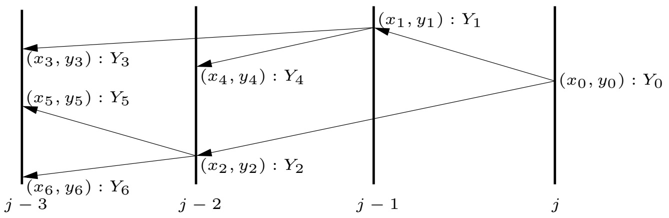

Fig. 1. Beginning with a pixel (x_0, y_0) with luminance Y_0 in a B-frame j, possible trajectory locations are estimated through the concatenation of motion vectors referencing previously encoded B-frames.

$$\Delta Y_i < T, \Delta U_i < T, V_i < T, T = \begin{cases} 2T_Y, \text{QP} < 30 \\ 4T_Y, \text{QP} \geq 30 \end{cases} \tag{7}$$

for a given threshold T_Y, $0 \leq T_Y \leq 7$. Otherwise, the sample is not used for averaging but the trajectory formation is continued.

2.2. Temporal motion consistency

Another possible method for identifying badly suited motion vectors is an examination of the vectors themselves. For better comparability the vectors are first scaled according to the temporal distance which they span. Whenever a new vector from list 0 $(dx_{r0}, dy_{r0})^T$ is added to the trajectory it is compared with the vector $(dx_i, dy_i)^T$ that led to the current trajectory location. The trajectory is only continued, if

$$\sqrt{(dx_{r0} - dx_i)^2 + (dy_{r0} - dy_i)^2} \leq T_{\text{TC}} \tag{8}$$

with a threshold $0 \leq T_{TC} \leq 7$ in quarterpel. New vectors for reference list 1 are examined in the same manner.

2.3. Spatial motion consistency

For QP 30 and higher the spatial similarity of motion vectors is also examined. In this context the block vote metric BV shall denote the number of neighboring 4×4 blocks surrounding the current trajectory position whose motion vectors differ by more than 30% of the current motion vector's length or at least 0.3 quarterpel from the current one. The block vote metric with scaled motion vectors is illustrated by Figure 2. The trajectory is only continued beyond the current pixel $(x, y)^T$ if the block vote metric for the new motion vector satisfies

$$BV(x, y) \leq T_{\text{SC}}, 0 < T_{\text{SC}} \leq 8. \tag{9}$$

2.4. Threshold optimization and signaling

All combinations of the three thresholds can be tested simultaneously at the encoder. The combination yielding the minimum sum of squared differences (SSD) is then selected and

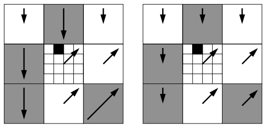

Fig. 2. *Left:* The 4×4 blocks marked in gray span a greater temporal distance than those colored in white. After the temporal scaling (*right*) the number of neighboring motion vectors differing from the current one (around the black pixel) corresponds to a block vote value of 5.

transmitted to the decoder requiring 9 additional bits. In addition, a flag is used to switch off the TTF for the current frame, in which case no thresholds are written to the bit stream. A complexity evaluation of the TTF-encoder and a more detailed description of the thresholds can be found in [7].

3. QUADTREE PARTITIONING ALGORITHM

As will be shown in Section 4 the TTF can produce bit rate reductions for many sequences, but provides only a small gain in the presence of large foreground objects at low frame rates. To improve this behavior a quadtree-based signaling of thresholds for certain image regions, which allows the filter to adapt itself to differently moving regions, is now proposed. The main idea for the following description is based on the Quadtree-based Adaptive Loop Filter (QALF) as described in [9]. The QALF uses a quadtree structure to signal, which image regions are to be processed by the ALF. For the Quadtree-based Temporal Trajectory Filter (QTTF) a set of thresholds is transmitted for every block as signaled by the quadtree. The encoding and decoding of the QTTF's quadtree flags shall observe the following rules:

- If a "1" is sent, the current frame or block is split into four subblocks of equal size by dividing it once horizontally and vertically. Afterwards, the four newly created subblocks are examined.
- If a "0" is sent, the current block is not split again. Instead, the three thresholds yielding the minimum SSD for the current block are transmitted. If T_Y is set to zero, the filter is disabled and the other two thresholds are omitted.

Initially, the optimal parameter set for an entire frame is computed yielding a minimum SSD_total and requiring an overhead rate R_total of 10 bits. Should no SSD improvement be possible on frame level, then only four bits need to be transmitted to disable the filter. SSD_total and R_total are combined into an RD-cost

$$C_\text{total} = \text{SSD}_\text{total} \cdot \lambda + R_\text{total}. \qquad (10)$$

Where the Lagrangian multiplier λ is identical to the one used during the optimization of the ALF. Afterwards the four subpartitions of the frame, as described above, are examined and a set of optimal thresholds for each of these is calculated yielding four new distortions SSD_1 to SSD_4 with an associated bit rate R_new. These can again be combined into a RD-cost for signaling the thresholds on subpartition level

$$C_\text{new} = (\text{SSD}_1 + \text{SSD}_2 + \text{SSD}_3 + \text{SSD}_4) \cdot \lambda + R_\text{new}. \qquad (11)$$

Should C_new be smaller than C_total then each subpartition is in turn again examined and can possibly be split. Otherwise the quadtree partitioning for the current frame or block is stopped. In addition, blocks with a width or height equal to or smaller than 16 pel are also not split again. Blocks are examined in top-down order. I.e. in a frame of 64×64 pel all 32×32 blocks would be examined before smaller block sizes are considered. The main advantage of this approach is a reduced search time for the optimal configuration. However, the best possible quadtree structure may not always be found, since blocks are only split as long as the RD-cost is decreased at the same time.

4. EXPERIMENTAL EVALUATION

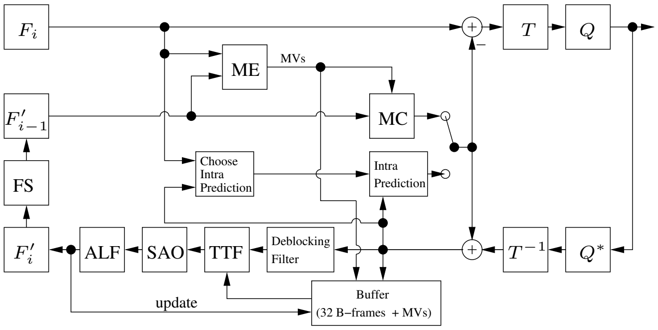

Fig. 3. The QTTF's reconstructed reference frames for trajectory formation are extracted before the deblocking filter is applied. Frames filtered by the QTTF are then processed by both SAO and ALF. Afterwards the most recent frame in the QTTF's buffer is updated.

The QTTF has been integrated into the HEVC reference software HM 3.0. The modified encoder is shown in Figure 3. Experiments have been conducted on all sequences listed in Table 1 for the proposed low delay high efficiency setting [2] of HEVC. Every sequence has been encoded at QPs 22, 27, 32, and 37 both with TTF and QTTF. The resulting bit rates and PSNR values are compared against the original HEVC settings using the Bjøntegaard metric [10]. The resulting BD-rates for the TTF are given in column 4 of Table 1. The respective BD-rates for the QTTF are listed in column 6. For the set of test sequences the TTF provides a bit rate reduction of 1.2% on top of the HEVC encoder. For sequences with large foreground objects, such as *BasketballPass*, the achieved gain is, however, relatively small. A slight loss of 0.18% occurs for the high-resolution sequence *Vidyo1*. The

Sequence	Resolution	TTF		QTTF	
		Δ_{PSNR} in dB	BD-rate in %	Δ_{PSNR} in dB	BD-rate in %
BasketballPass	416x240, 50Hz	0.00	−0.04	0.01	−0.19
BlowingBubbles	416x240, 50Hz	0.03	−0.64	0.04	−1.05
BQSquare	416x240, 60Hz	0.23	−6.06	0.26	−6.98
RaceHorses	416x240, 30Hz	0.01	−0.13	0.01	−0.15
BasketballDrill	832x480, 50Hz	0.00	−0.12	0.00	−0.11
BQMall	832x480, 60Hz	0.00	−0.09	0.00	−0.10
PartyScene	832x480, 50Hz	0.02	−0.42	0.05	−1.17
RaceHorses	832x480, 30Hz	0.00	−0.05	0.00	−0.09
Vidyo1	1280x720, 60Hz	0.00	0.18	0.02	−0.48
Vidyo3	1280x720, 60Hz	0.01	−0.27	0.01	−0.35
Vidyo4	1280x720, 60Hz	0.01	−0.38	0.02	−0.67
BQTerrace	1920x1080, 60Hz	0.01	−0.74	0.02	−1.31
ParkScene	1920x1080, 30Hz	0.01	−0.21	0.02	−0.56
Waterfall	704x480, 25Hz	0.23	−7.61	0.27	−8.95

Table 1. The sequences used for the experiments.

QTTF performs even better than the TTF for all sequences despite the signaling overhead for the quadtree. On average it provides a bit rate reduction of 1.6%. For some sequences, the quadtree improves the performance of the TTF by close to 1%, despite the fact, that the quadtree requires up to 400 additional bits per frame. Most importantly, no loss is observed for any of the sequences. Visual examples and RD-curves for all sequences may be found on the accompanying website `www.nue.tu-berlin.de/research/qttf`. To illustrate the adaptability of the QTTF Figure 4 shows the quadtree partition together with the threshold T_Y for frame 36 of the *BQSquare* sequence. The black block ($T_Y = 0$) near the upper right corner indicates that the QTTF is switched off around the two walking persons, while the surrounding area is filtered. Currently the quadtree-structure is written as raw data into the bit stream.

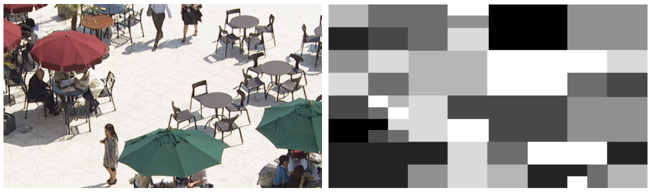

Fig. 4. *Left:* frame 36 of the *BQSquare* sequence, *right:* color coded quadtree structure for the threshold T_Y. Black blocks correspond to $T_Y = 0$ and white ones to $T_Y = 7$.

5. SUMMARY

The proposed QTTF algorithm combines the advantages of pixel-based temporal trajectory filtering with a RD-optimized quadtree-partitioning algorithm. It has been shown that the filter can significantly improve the performance of the HEVC encoder for the low delay high efficiency setting, even in combination with several other in-loop filters. The maximum additional bit rate reduction on top of HEVC provided by the QTTF is close to 9%. The results may be further improved by defining appropriate context-models and encoding the additional side-information with HEVC's CABAC algorithm. Future work will also include studies of the interactions between QTTF and the CU-based ALF.

6. REFERENCES

[1] P. List, A. Joch, J. Lainema, G. Bjøntegaard, and M. Karczewicz, "Adaptive deblocking filter," *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, vol. 13, no. 7, pp. 614–619, Jul 2003.

[2] K. McCann, T. Wiegand, B. Bross, W.-J. Han, J.-R. Ohm, J. Ridge, S. Sekiguchi, and G. J. Sullivan, "Hevc draft and test model editing," *ITU-T SG16 WP3, ISO/IEC JTC1/SC29/WG11 doc. JCTVC-D500-r1*, Mar 2011.

[3] C.-M. Fu, C.-Y. Chen, Y.-W. Huang, and S. Lei, "Sample adaptive offset for hevc," *IEEE 13th Intl. Workshop on Multimedia Signal Processing (MMSP)*, pp. 1–5, 2011.

[4] S. Wittmann and T. Wedi, "Transmission of post-filter hints for video coding schemes," *Proceedings of the 25th Picture Coding Symposium (PCS)*, pp. 81–84, Sep 2007.

[5] D. T. Vo and T. Q. Nguyen, "Optimal motion compensated spatio-temporal filter for quality enhancement of H.264/AVC compressed video sequences," *Proceedings of the 26th ICIP*, pp. 3173–3176, Nov 2009.

[6] M. Esche, A. Glantz, A. Krutz, and T. Sikora, "Adaptive temporal trajectory filtering for video compression," *IEEE TCSVT to appear*, vol. PP, no. 99, pp. 1, 2012, available on IEEExplore.

[7] M. Esche, A. Glantz, A. Krutz, M. Tok, and T. Sikora, "Weighted temporal long trajectory filtering for video compression," *submitted to PCS*, 2012.

[8] A. Golwelkar and J.W. Woods, "Motion-compensated temporal filtering and motion vector coding using biorthogonal filters," *IEEE TCSVT*, vol. 17, no. 4, pp. 417–428, April 2007.

[9] T. Chujoh, N. Wada, T. Watanabe, G. Yasuda, and T. Yamakage, "Specification and experimental results of quadtree-based adaptive loop filter," *ITU-T SG16/Q.6 VCEG document VCEG-AK22*, Apr 2009.

[10] G. Bjøntegaard, "Calculation of average PSNR differences between RD-curves," *ITU-T SG16/Q.6 VCEG document VCEG-M33*, Mar 2001.