Weighted Temporal Long Trajectory Filtering for Video Compression

Marko Esche, Alexander Glantz, Andreas Krutz, Michael Tok, and Thomas Sikora
Technische Universität Berlin
Communication Systems Group, Sekr. E-N 1
Einsteinufer 17, D-10587 Berlin, Germany
Email: {esche,glantz,krutz,tok,sikora}@nue.tu-berlin.de

Abstract— **In the context of the HEVC standardization activity, in-loop filters such as the adaptive loop filter and the deblocking filter are currently under investigation. Both filters work in the spatial domain only, despite the temporal correlation within video sequences. In this work a previously introduced filter, that uses temporal information for deblocking and denoising instead, is integrated into the HEVC test model HM 3.0. It is shown how the filter is to be adapted to work in combination with the adaptive loop filter for the HEVC low-delay profile. In addition, an optimal weighting function for the filtered luma samples based on the qunatization parameter is derived. Bit rate reductions of up to 7.6% are reported for individual sequences.**

I. Introduction

Ever since the standardization of H.264/AVC it has been a well-known fact that in-loop filters can be used to improve both the objective and subjective quality of a decoded video sequence through noise reduction. In [1], List et al. identified the main two sources of the noise introduced during the encoding process, namely block-based motion estimation (ME) and the quantization of DCT coefficients. In the current test model of the HEVC standardization activity HM [2] two in-loop filters are employed. The first is the deblocking filter (DF) originally described in [1], the second is the Wiener-based adaptive loop filter ALF [3]. The ALF can be easily adapted to changing video content through the use of varying filter kernels and the control of the filtered image regions by use of a quad-tree partitioning algorithm. Despite the fact that ME adds a temporal component to the noise in the encoded sequence, both filters use information from one frame only and thus remain inherently spatial filters. Attempts to overcome this deficiency were, for instance, described in [4] and [5]. In [4] temporal filtering along motion trajectories was performed as an analysis/synthesis step during 3-D subband coding, which combines both prediction and noise reduction. The global motion temporal filter detailed in [5] performs background filtering by warping neighboring frames into the current one through the use of homographies transmitted in the bitstream. In [6], the authors described the temporal trajectory filter (TTF) that makes use of individual pixel trajectories derived from the sequence's motion vectors to perform temporal filtering. In this context, a pixel trajectory is defined as the locations through which a certain image point moves over the duration of a video sequence. For a given dataset, bit rate reductions of up to 12% were reported for the H.264/AVC baseline profile. In this paper, the TTF is investigated in the environment of the HM. In addition, it is shown how the length of individual pixel trajectories can be increased to achieve better filtering performance. The remainder of the paper is structured as follows. Section II derives a mathematical model for the noise encountered along a pixel trajectory, if the trajectory is completely known. Based on this model, optimal filter coefficients are calculated. In Section III, three previously described thresholds are revisited that allow the calculation of pixel trajectories directly from the motion vectors conveyed in the bit stream. In addition, it is shown how the filter needs to be modified to extend the length of the temporal trajectories. Section IV presents the experimental evaluation conducted in the context of the HM. To this end, the filter's performance is also compared against the ALF and a combination of both filters is presented. Section V summarizes the paper.

II. Theoretical Basis

For any given pixel $(x_0, y_0)^T$ in frame j it is assumed that its locations $(x_i, y_i)^T$, $1 \leq i < N$, in $N-1$ previous frames are also known. If $Y_n(x, y)$ denotes the luminance component of frame n at location $(x, y)^T$ the distorted versions of the original sample $Y_j(x_0, y_0)$ in any of the $N-1$ previous frames given by

$$Y_{j-i}(x_i, y_i) = Y_j(x_0, y_0) + n_i, 1 \leq i < N. \qquad (1)$$

Even though the motion of the pixel is perfectly known, a noise term n_i with variance σ_i is introduced due to the reduced quality of the encoded sequence. As described in [6], it can be assumed that all n_i are uncorrelated. A filtered version of the original luma component can then be computed by calculating a weighted mean

$$\begin{aligned} Y_j^*(x_0, y_0) &= \frac{1}{N} \sum_{i=0}^{N-1} \beta_i \cdot \hat{Y}_{j-i}(x_i, y_i) \\ &= \frac{1}{N} \sum_{i=0}^{N-1} \beta_i \cdot Y_j(x_0, y_0) + \frac{1}{N} \sum_{i=0}^{N-1} \beta_i \cdot n_i \qquad (2) \\ &= \frac{1}{N} Y_j(x_0, y_0) \sum_{i=0}^{N-1} \beta_i + \frac{1}{N} \sum_{i=0}^{N-1} \beta_i \cdot n_i. \end{aligned}$$

Where β_i are the individual weights per frame with $\sum_{i=0}^{N-1} \beta_i = N$ to make the filter unbiased. This leads to the definition of a new noise term $\tilde{n}_j$ for the filtered pixel

$$Y_j^*(x_0, y_0) = Y_j(x_0, y_0) + \underbrace{\frac{1}{N}\sum_{i=0}^{N-1} \beta_i \cdot n_i}_{\tilde{n}_j}. \tag{3}$$

The variance of the filtered noise $\tilde{n}_j$ is subsequently given by

$$\begin{aligned} \sigma_{\tilde{n}}^2 &= E\left[\tilde{n}_j \tilde{n}_j\right] \qquad (4)\\ &= E\left[\frac{1}{N^2}\sum_{l=0}^{N-1}\beta_l n_l \cdot \sum_{k=0}^{N-1}\beta_k n_k\right] = \frac{1}{N^2}\sum_{m=0}^{N-1}\beta_m^2\sigma_m^2. \end{aligned}$$

As the filter is to minimize $\sigma_{\tilde{n}}^2$ constraint to $\sum_{i=0}^{N-1}\beta_i = N$, the minimum may be found by Lagrangian minimization

$$\begin{aligned} &\frac{\partial}{\partial\beta_i}\left[\sum_{m=0}^{N-1}\beta_m^2\sigma_m^2 - \lambda\left(N - \sum_{k=0}^{N-1}\beta_k\right)\right] = 0 \qquad (5)\\ &2\beta_i\sigma_i^2 + \lambda = 0 \Rightarrow \beta_i = -\frac{\lambda}{2\sigma_i^2}\\ &\sum_{k=0}^{N-1}\beta_k = N \Leftrightarrow \sum_{k=0}^{N-1} -\frac{\lambda}{2\sigma_k^2} = N\\ &\lambda = \frac{N}{\sum_{k=0}^{N-1} -\frac{1}{2\sigma_k}} \qquad (6)\\ &\beta_i = -\frac{\lambda}{2}\frac{1}{\sigma_i^2} = \frac{N/\sigma_i^2}{\sum_{k=0}^{N-1}\frac{1}{\sigma_k^2}}. \end{aligned}$$

In theory, the reconstruction error variance σ_k^2 for every pixel along the trajectory would be required to calculate the optimal filter weight β_i. According to Wiegand and Girod [7] the distortion variance in a reconstructed frame is given by

$$D_{\text{REC}} = \frac{Q_{\text{step}}^2}{3} \text{ with zero mean.} \tag{7}$$

Where Q_{step} is the quantizer step size selected by the quantization parameter QP. Both in H.264/AVC and in HEVC, Q_{step} is roughly

$$Q_{\text{step}} = 0.625 \cdot 2^{\frac{\text{QP}}{6}}. \tag{8}$$

Subsequently, the optimal filter weight for frame i according to its QP may be calculated

$$\begin{aligned} \sigma_i^2 &= D_{\text{REC},i} = \frac{1}{3}\left(0.625 \cdot 2^{\frac{\text{QP}_i}{6}}\right)^2\\ \beta_i(\text{QP}_i) &= 3 \cdot \left(0.625 \cdot 2^{\frac{\text{QP}_i}{6}}\right)^{-2}. \qquad (9) \end{aligned}$$

In the case of the H.264/AVC baseline profile this yields identical weights for every frame. When varying QPs are used as in the HEVC low-delay setting, the optimal weights can be calculated at the decoder requiring no additional side information.

III. Filter Design

In the low-delay high efficiency setting of HEVC with an IBBB coding structure every B-predicted block can have up to two motion vectors pointing to one of the last four encoded pictures. It is assumed that the motion vector for a given block also describes the individual motion of every pixel within the block. The components of the two resulting motion vector fields for frame i shall be denoted by $(dx_{i,0}, dy_{i,0})^T$ and $(dx_{i,1}, dy_{i,1})^T$. Starting again with pixel $(x_0, y_0)^T$ in frame i, two possible locations of the pixel in the referenced frames are therefore given by

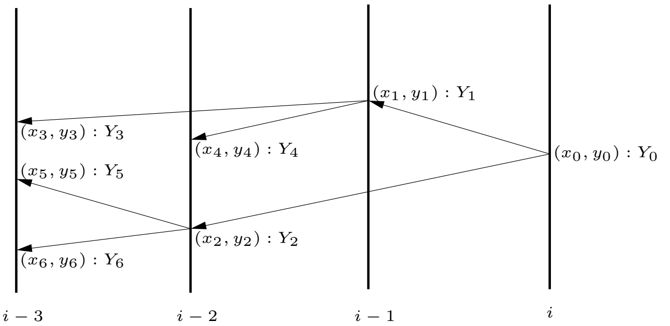


Fig. 1. Starting at a pixel (x_0, y_0) with luminance Y_0 in an arbitrary B-frame i, possible trajectory locations are derived through the concatenation of motion vectors pointing to previously encoded B-frames.

$$\begin{aligned} \begin{pmatrix} x_1\\ y_1 \end{pmatrix} &= \begin{pmatrix} x_0\\ y_0 \end{pmatrix} + \begin{pmatrix} dx_{i,0}(\lfloor x_0\rfloor, \lfloor y_0\rfloor)\\ dy_{i,0}(\lfloor x_0\rfloor, \lfloor y_0\rfloor) \end{pmatrix},\\ \begin{pmatrix} x_2\\ y_2 \end{pmatrix} &= \begin{pmatrix} x_0\\ y_0 \end{pmatrix} + \begin{pmatrix} dx_{i,1}(\lfloor x_0\rfloor, \lfloor y_0\rfloor)\\ dy_{i,1}(\lfloor x_0\rfloor, \lfloor y_0\rfloor) \end{pmatrix}. \end{aligned} \tag{10}$$

Figure 1 shows how the concatenation of motion vectors is used to derive possible pixel locations over a GOP of four frames. However, not all of these describe the true motion of the pixel. It becomes therefore necessary, to discard those motion vectors that have purely been chosen due to rate-distortion optimization and thus may not relate to the true motion of pixels. To this end three thresholds are used. In each of the following equations the motion vectors are scaled according to the temporal distance that they span.

A. Absolute Error Along the Trajectory

For every pixel the absolute difference of two consecutive luminance samples $\Delta Y_i = Y_{i+1} - Y_i$ together with the respective chrominance differences ΔU_i and ΔV_i are calculated. A sudden change in one of these differences is assumed to indicate that a motion vector no longer describes the true motion of a pixel. The trajectory is only continued, if

$$\Delta Y_i < T, \Delta U_i < T, \Delta V_i < T, T = \begin{cases} 2T_Y, \text{QP} < 30\\ 4T_Y, \text{QP} \geq 30 \end{cases} \tag{11}$$

for a given threshold T_Y, $0 \leq T_Y \leq 7$.

B. Temporal Motion Consistency

In addition, the similarity of consecutive motion vectors is tested. A trajectory is expected to be correct as long as its motion does not change significantly over time. When examining a new motion vector for list 0 for any given pixel of the trajectory, its Euclidean distance to the vector pointing to

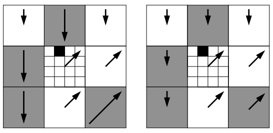


Fig. 2. On the *left* the original motion vectors for all 4×4 blocks surrounding the trajectorie's current location are shown. Vectors that span a temporal distance greater than 1 are marked in gray. After the scaling (*right*) the BV-metric (i.e. the number of vectors differing from the current one) for the current block is 5.

the current location is calculated. The trajectory is continued, as long as

$$\sqrt{(dx_{r0} - dx_i)^2 + (dy_{r0} - dy_i)^2} \leq T_{\mathrm{TC}} \qquad (12)$$

with $0 \leq T_{\mathrm{TC}} \leq 7$ in quarter-pel, where dx_{r0} and dy_{r0} are the components of the motion vector for list 0 pointing from the current frame to reference frame $r0$. The temporal motion consistency is also checked for the vectors of reference list 1.

C. Spatial Motion Consistency

Aside from a temporal comparison of motion vectors, spatial similarity is also examined, which is a measure for the reliability of a motion vector. At every frame the motion vector for a pixel of the trajectory is compared with its eight neighbors on 4×4 block level. In this context the block-vote metric BV denotes the number of neighboring motion vectors that differ significantly from the current one. The allowed maximum difference is 30% of the original motion vector's length or at least 0.3 quarter-pel. Both values were chosen empirically and proved to be well suited for all sequences. Both the scaling of motion vectors and the block-vote metric are illustrated by Figure 2. The filtering along the trajectory is continued only, if the block-vote metric for the current pixel satisfies

$$BV_{i,0}(x, y) \leq T_{\mathrm{SC}}, 0 < T_{\mathrm{SC}} \leq 8. \qquad (13)$$

D. Long Trajectories

In the previous implementation described in [6], a trajectory was interrupted as soon as the luminance difference became bigger than the threshold T_Y. This makes the coder biased towards shorter trajectories. In theory, however, the quality of the filtering process is increased with the number of samples used. The new design, therefore, does not stop the trajectory formation all together, but simply omits the luma sample in question from the filtering process and continues the trajectory. For the threshold T_Y the last filtered luminance samples and the respective chrominance samples are now used for the calculation of ΔY_i, ΔU_i, and ΔV_i.

E. Parameter Calculation

All possible parameter combinations can be tested simultaneously at the encoder. The parameter combination yielding the minimum mean square error is then selected. Each of the thresholds is transmitted to the decoder requiring 9 additional bits per frame. A tenth bit can be used to disable the filter for the current frame all together, in which case the other thresholds are simply omitted.

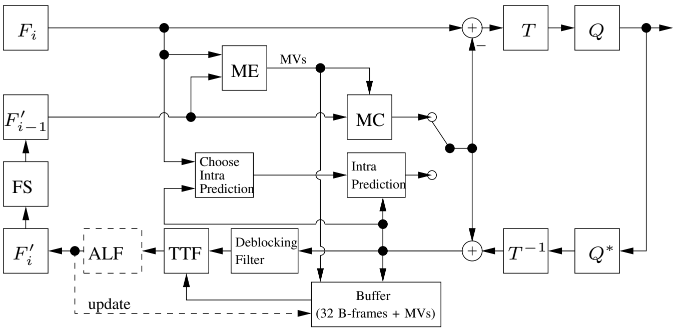


Fig. 3. The TTF is included in the local decoder loop of the encoder after the deblocking filter. For the first test the ALF was disabled. When both ALF and TTF are used together, the respective frame in the TTF's buffer is updated after the ALF has been applied.

IV. Experimental Evaluation

The TTF has been integrated into the HEVC test model HM 3.0. Up to 32 previously decoded unfiltered frames are kept in a buffer to be used for the trajectory formation. The resulting encoder is depicted in Figure 3, where the TTF is used after the deblocking filter. In this setting the ALF (dotted connections) was disabled. Tests have been conducted for a variety of sequences listed in Table I. The exact configuration for the low-delay high efficiency setting may be found in [2].

The HM 3.0 without the ALF is compared against the HM with the added TTF using the Bjøntegaard metric described in [8]. For comparison, the individual gain provided by the ALF per sequence was also calculated. The resulting BD-rates for both filters may be found in Table I in columns 4 and 6. For all tested sequences except *Vidyo3* the TTF produces a bit rate reduction. For *Vidyo3* the increase of 0.07% is, however, only very slight. Nevertheless, the ALF produces a higher gain than the TTF for all tested sequences. This is only to be expected, as the ALF has undergone many significant improvements over the last years. Even though, both filters produce similar gains both for *BQSquare* and for *Waterfall*. The average bit rate reduction produced by the TTF for the dataset is 1.5%. The true potential of both filters can be exploited when both are used in combination. In this setting, the average encoding time is increased by 190% compared to the HEVC encoder with the ALF. The decoder complexity, however, is only increased by about 30%. In order to show that ALF and TTF are not mutually exclusive, columns 8 and 10 of Table I compare HM 3.0 with and without ALF against the combination of ALF and TTF. In this case, a modified encoder as depicted in Figure 3 with the dotted connections is used. The combination of both filters outperforms the test model HM 3.0 for almost all sequences. For *BQSquare* and *BlowingBubbles* in particular, the gain produced by both filters together equals the sum of their individual quality improvements. A possible explanation for this finding may be different noise sources that are compensated separately by both filters, so that there is no uninteded interference between the two approaches. In this combination, the TTF provides an average BD-rate of -1.4% when compared with the HEVC low-delay

TABLE I

BD-RATES AND AVERAGE PSNR-GAIN FOR THE SEQUENCES USED IN THE EXPERIMENTS

Sequence	Resolution, framerate in Hz	TTF vs. HEVC		ALF vs. HEVC		TTF+ALF vs. HEVC		TTF+ALF vs. HEVC+ALF		TTF+ALF simple average	
		Δ_{PSNR} in dB	BD-rate in %	Δ_{PSNR} in dB	BD-rate in %	Δ_{PSNR} in dB	BD-rate in %	Δ_{PSNR} in dB	BD-rate in %	Δ_{PSNR} in dB	BD-rate in %
BlowingBubbles	416x240, 50	0.03	−0.70	0.05	−1.29	0.08	−1.92	0.03	−0.64	0.02	−0.42
BQSquare	416x240, 60	0.18	−4.77	0.21	−5.71	0.44	−11.48	0.23	−6.06	0.08	−2.21
RaceHorses	416x240, 30	0.00	−0.05	0.05	−0.99	0.00	−0.05	0.01	−0.13	0.01	−0.12
PartyScene	832x480, 50	0.02	−0.53	0.13	−3.13	0.15	−3.53	0.02	−0.42	0.01	−0.17
Vidyo1	1280x720, 60	0.02	−0.60	0.17	−4.95	0.17	−4.79	0.00	0.18	0.01	−0.35
Vidyo3	1280x720, 60	0.00	0.07	0.37	−10.38	0.38	−11.09	0.01	−0.27	0.00	−0.15
Vidyo4	1280x720, 60	0.01	−0.39	0.17	−5.42	0.18	−5.80	0.01	−0.38	0.01	−0.33
BQTerrace	1920x1080, 60	0.02	−1.46	0.17	−9.73	0.19	−10.41	0.01	−0.74	0.00	−0.21
ParkScene	1920x1080, 24	0.01	−0.38	0.07	−2.07	0.07	−2.29	0.01	−0.21	0.01	−0.11
Allstars	704x576, 25	0.01	−0.25	0.25	−7.46	0.25	−7.66	0.01	−0.21	0.01	−0.26
BBC-pan-13	720x576, 25	0.00	−0.11	0.21	−6.53	0.20	−6.41	0.00	0.11	0.00	0.10
Waterfall	704x480, 25	0.26	−8.95	0.29	−9.49	0.51	−16.50	0.23	−7.61	0.20	−6.83
Average		**0.05**	**−1.52**	**0.18**	**−5.60**	**0.22**	**−6.83**	**0.05**	**−1.37**	**0.03**	**−0.92**

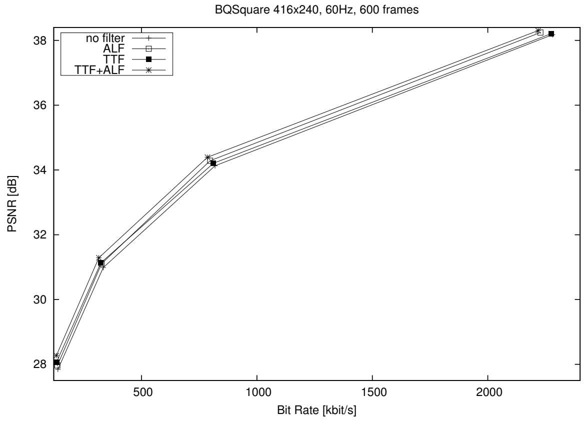
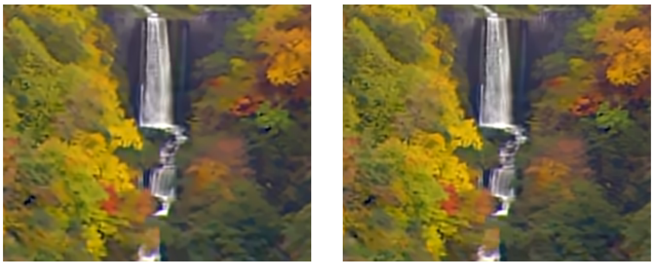


Fig. 4. RD-curves for all three tested settings for the *BQSquare* sequence, QP 22 to 37.

profile with ALF enabled. For comparison column 12 of Table I also shows the BD-rate produced by TTF and ALF if no long trajectories and a simple average instead of a weighted mean are used. The average BD-rate for the simplified filter is only -0.9%, which provides evidence for the effectiveness of the weighted filtering. An exemplary RD-curve for the *BQSquare* sequence is given in Figure 4. Below 500 kbit/s the TTF performs better than the ALF. For all depicted QPs the combination of both filters outperforms the simple ALF. Apart from objective quality improvements the TTF also increases the visual quality of the decoded video. A part of an exemplary decoded frame from the *Waterfall* sequence is shown in Figure 5. All decoded sequences together with the respective RD-curves may be found on the accompanying website `www.nue.tu-berlin.de/research/wtltf`.

V. Summary

The main objective of this work was to demonstrate the possiblity of further improving the HEVC test model through the use of a temporal filtering approach. In combination with the optimal sample weighting described in Section II, the proposed filter produces an average BD-rate of -1.4% when included in the HEVC test model. Additional improvements may be achieved by further investigating both long trajectories and weighted averaging seperately. Future work will focus both on possible interactions between in-loop filtering approaches and on reducing the encoder complexity of the TTF.



(a) ALF, 31.1 dB at 69.3kbit/s (b) TTF+ALF, 31.3 dB at 69.2kbit/s

Fig. 5. Exemplary decoded frames from the *Waterfall* sequence.

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 and ISO/IEC JTC1/SC29/WG11 document JCTVC-D500-r1.doc*, Mar 2011.

[3] 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.

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

[5] A. Glantz, A. Krutz, M. Haller, and T. Sikora, “Video coding using global motion temporal filtering,” *Proceedings of the 16th International Conference on Image Processing (ICIP)*, pp. 1053–1056, Nov 2009.

[6] M. Esche, A. Krutz, A. Glantz, and T. Sikora, “A novel in-loop filter for video-compression based on temporal pixel trajectories,” *Proceedings of the 26th PCS*, pp. 514–517, Dec 2010.

[7] T. Wiegand and B. Girod, “Lagrange multiplier selection in hybrid video coder control,” *Proceedings of the International Conference on Image Processing (ICIP)*, vol. 3, pp. 542–545, 2001.

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