Jiménez-Moreno, A., Martínez-Enríquez, E. & Díaz-de-María, F. (2013). Standard compliant flicker reduction method with PSNR loss control. In *2013 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2013): Proceedings*. (pp. 1729 - 1733). IEEE
DOI: http://dx.doi.org/10.1109/ICASSP.2013.6637948

STANDARD COMPLIANT FLICKER REDUCTION METHOD WITH PSNR LOSS CONTROL

A. Jiménez-Moreno, E. Martínez-Enríquez, F. Díaz-de-María

Department of Signal Theory and Communications
Universidad Carlos III, Leganés (Madrid), Spain

ABSTRACT

Flicker is a common video coding artifact that occurs especially at low and medium bit rates. In this paper we propose a temporal filter-based method to reduce flicker. The proposed method has been designed to be compliant with conventional video coding standards, i. to generate a bitstream that is decodable by any standard decoder implementation. The aim of the proposed method is to make the luminance changes between consecutive frames smoother on a block-by-block basis. To this end, a selective temporal low-pass filtering is proposed that smooths these luminance changes on flicker-prone blocks. Furthermore, since the low-pass filtering can incur in a noticeable blurring effect, an adaptive algorithm that allows for limiting the PSNR loss -and thus the blur- has also been designed. The proposed method has been extensively assessed on the reference software of the H.264/AVC video coding standard and compared to a state-of-the-art method. The experimental results show the effectiveness of the proposed method and prove that its performance is superior to that of the state-of-the-art method.

Index Terms— Flicker, temporal filtering, H.264.

1. INTRODUCTION

Flicker is a common artifact in video coding that occurs especially at low and medium bit rates. Flicker happens as a result of the fact that the video encoder does not consistently treat the co-located blocks of consecutive frames, thus increasing the inter-frame difference with respect to that of the original video. Flicker is, therefore, a temporal artifact that is perceived as a sudden change in the reconstructed pixel values of the same area of two consecutive frames. As stated in [1], this is a consequence of the fact that these pixel values are coded through different coding processes. Specifically, flicker becomes a more serious problem when coding a periodic Intra-coded frame (I-frame), since it comes after an Inter-coded frame (either P- or B-frame) and the redundancy-removing methods used by the encoder notably change from one to another. Furthermore, this artifact can be more clearly perceived when static areas are coded at low or medium bit rates. In these cases, Inter frames tend to copy the pixel values from the previous frames, creating time consistency that is abruptly broken when an I-frame comes, since the prediction source changes (from temporal to spatial neighbors) and flicker becomes much more apparent.

Flicker reduction has been a relevant topic of research since many years ago; however, a definite solution has not been found yet. Next, we briefly summarize some previous works that have addressed this problem. In [2] the so-called Detented Quantization method was proposed. This method aimed to reduce the discontinuity in coding distortion patterns between consecutive Inter and Intra frames by proposing a quantization scheme that produces the Intra-coded macroblock (MB) most similar to an Inter-coded version of the same MB. In [3] a two-pass I-frame coding method was proposed. For each I-frame, the first pass involves obtaining a simplified P-frame that serves as no-flicker reference. In the second pass, the ficker-prone MBs are coded using this simplified P-frame as the target and a lower QP value, while the remaining MBs are coded using the normal Intra coding. Moreover, in [3] an effective flicker metric is proposed that will be explained later. [1] presented an Intra prediction mode selection method based on a modified distortion measure. In particular, in the flicker-prone MBs, the distortion term of the cost function used in the Rate Distortion Optimization (RDO) process [4] is modified by adding a flicker distortion measure. [5] described a rate control method that focuses on quality consistency. Specifically, exponential models are proposed that take into account the target rate and the buffer occupancy, and maintain the distortion as constant as possible, thus reducing the flicker effect. [6] proposed a flicker reduction method based on quantizing (in the frequency domain) the predicted version of a MB, prior to obtain the residue, so that predicted MB has little effect on the reconstructed MB, thus reducing the flicker effect. [7] dealt with various annoying effects (such as flicker, blocking, or ringing) by means of a post-filtering process. Specifically, the authors proposed to use a fuzzy filter that is able to reduce the artifacts by increasing the pixel correlation, while keeping the edges of the image. The method proposed in [8] is another example of a post-filtering process. In this case, a robust statistical temporal filter (RSTF) that relies on both spatial and temporal neighbors was proposed.

These previous works have some drawbacks that we address in this paper. The algorithm proposed in [6] is not standard compliant, what seriously limits its field of application. Flicker reduction methods in [7] and [8] require to perform filtering processes at the decoder side, what prevents its use when you do not have control on the decoder and any of the available standard decoder implementation has to be used. In [1], the flicker-prone MBs are selected based on a fixed threshold, which does not guarantee a proper generalization ability. In this paper we propose a method that aims to overcome the above mentioned drawbacks. In particular, we suggest to use a temporal low-pass filter that is implemented in the encoder (thus, it is not a decoder-side post-processing technique) obtaining a standard compliant algorithm. Furthermore, the parameters of the filter are estimated *on-the-fly*, so that it is adapted to the video content avoiding generalization problems. Finally, our approach allows controlling the PSNR drop, keeping it under an user-defined threshold, to prevent the perceived visual quality from impoverishing due to the blurring effect caused by the temporal filtering.

The rest of the paper is organized as follows. Section 2 provides a detailed explanation of the proposed method. Section 3 describes the experiments conducted and presents and discuss the results obtained. Finally, Section 4 summarizes our conclusions.

2. PROPOSED METHOD

This paper proposes a novel method to reduce flicker that is compliant with conventional video coding standards. Given that flicker is perceived as a temporal artifact generated by a sudden change in the luminance values of certain blocks between two consecutive frames (generally, a P- or B-frame followed by an I-frame), we suggest to reduce this effect by making the temporal evolution of the luminance smoother. In particular, we propose to make the reconstructed pixel values of an I-frame more similar to those of the previous Inter-frame by means of a temporal low-pass filter. Furthermore, since the low-pass filtering could sometimes produce a perceptible blurring effect, an algorithm has been designed to limit the PSNR loss and consequently the blurring effect.

The algorithm description has been organized in three subsections: first, we describe the proposed filtering technique; second, we explain how to make it compliant with typical video coding standards; and third, we describe the blur control algorithm.

2.1. Selective temporal motion-guided filtering

2.1.1. Filter formulation

To reduce flicker distortion, our approach aims to make consecutive reconstructed frames more similar to each other. To this end, we use the following temporal filtering:

$$\hat{f}^{'j}_{n,i} = \alpha \hat{f}^{j}_{n,i} + (1-\alpha)\, \hat{f}^{j}_{n-1,MV}, \tag{1}$$

where $\hat{f}^{'j}_{n,i}$ is the filtered value of the j-th pixel of the i-th MB of the n-th frame; $\hat{f}^{j}_{n,i}$ is the reconstructed value of the same pixel; $\hat{f}^{j}_{n-1,MV}$ is the corresponding reconstructed pixel in the previous frame (the meaning of the subindex MV is explained below); and α is a parameter that balances the weight of the current- and the previous-frame reconstructed pixel values. The sub-index MV has been introduced in $\hat{f}^{j}_{n-1,i,MV}$ to make clear that the co-located pixel is not always the one used to filter. The reason is that using the co-located pixel ($\hat{f}^{j}_{n-1,i}$) produces a significant blurring effect when the pixel region undergoes any kind of motion (either camera or object motion) from one frame to another. To solve this problem, we propose a motion-guided filter that takes the pixel in the frame $n-1$ pointed by a calculated motion vector (MV). To this purpose, a motion estimation (ME) process should also be carried out in the I-frames. In our experiments, we have employed 16×16-pixel blocks for this ME process, which is only used for guiding the proposed filtering (this MV is not sent to the decoder).

Furthermore, the proposed filter is selectively applied: not every block is filtered, but only those for which the suggested filtering actually reduces certain flicker metric that is described next.

2.1.2. Flicker-prone block detection

To detect flicker-prone blocks we need to define a metric that allows us to measure flicker distortion. An accurate flicker metric was proposed in [3]:

$$D^{j}_{flicker,n,i} = \max\left(0, \left|\hat{f}^{j}_{n,i} - \hat{f}^{j}_{n-1,i}\right| - \left|f^{j}_{n,i} - f^{j}_{n-1,i}\right|\right), \tag{2}$$

where $f^{j}_{n,i}$ is the original j-th pixel of the i-th MB of the n-th frame, $\hat{f}^{j}_{n,i}$ is the reconstructed value of the same pixel, and $f^{j}_{n-1,i}$ and $\hat{f}^{j}_{n-1,i}$ are the co-located original and reconstructed pixel values in the previous frame. It should be noted that if the difference between the reconstructed pixels is similar or smaller than that between the original pixels, $D^{j}_{flicker,n,i}$ is small or zero, and thus flicker should not be perceived. This means that the encoding process is not producing abrupt luminance changes between consecutive reconstructed frames. More details about this metric can be found in [3].

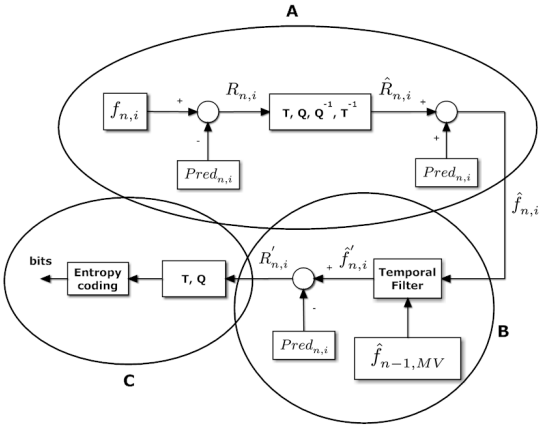

Fig. 1. Standard compliant implementation of the proposed temporal filtering.

The flicker distortion of a block is computed by accumulating pixel-wise distortions:

$$D_{flicker,n,i} = \sum_{j} D^{j}_{flicker,n,i}. \tag{3}$$

Our method relies on this metric to decide whether each block is filtered or not; thus, only those blocks for which the filtering reduces this flicker metric are actually filtered.

2.2. Standard compliant implementation

Our goal is to implement the proposed method in the encoder, so that the generated bitstream will be decodable by any standard decoder implementation. With this objective in mind, we propose to obtain a modified version of the Intra residue by considering the filtered version of the pixel block (1) as the target, i.e.:

$$R^{'j}_{n,i} = \hat{f}^{'j}_{n,i} - Pred^{j}_{n,i}, \tag{4}$$

where $R^{'j}_{n,i}$ is the modified residue, $\hat{f}^{'j}_{n,i}$ is the filtered pixel value, and $Pred^{j}_{n,i}$ is the corresponding Intra prediction. In this manner, in the absence of the quantization process, the decoder would reconstruct the filtered pixel values.

The implementation of the filtering process in a standard complying manner is summarized in Fig. 1, where T, Q, T^{-1}, and Q^{-1} stand for the transformation, quantization, inverse transformation, and inverse quantization processes, respectively. As can be observed, we need to generate the regular reconstructed version of the pixel block ($\hat{f}_{n,i}$) as shown in part A of the Figure, which illustrates the typical encoder-decoder loop at the encoder side. Then, we have to generate the low-pass filtered version ($\hat{f}^{'}_{n,i}$) and the modified residue $R^{'}_{n,i}$, as illustrated in part B. And finally, $R^{'}_{n,i}$ is transformed, quantized and entropy coded, as illustrated in part C, before including it in the bitstream.

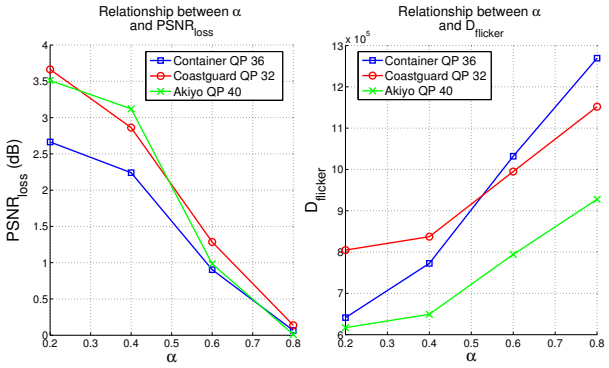

Fig. 2. Experimental evaluation of the relationship between α, $PSNR_{loss}$ and $D_{flicker}$ for several sequences.

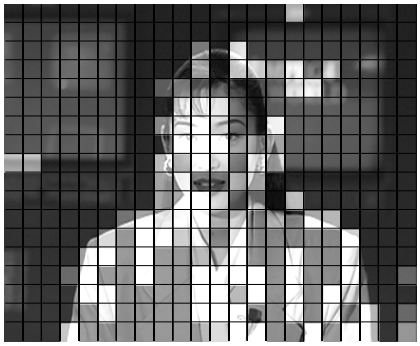

Fig. 3. Visual example of the selected $\alpha_{tar,i}$ for an Intra frame of the sequence Akiyo at QP 40.

2.3. *On-the-fly* estimation of the α filter parameter: controlling the blur

As can be easily inferred from (1) and (2), the lower the α value is, the more similar the filtered pixels are to the previous-frame corresponding pixels, and the lower flicker distortion is; and vice-versa. On the other hand, the lower the α value is, the stronger the low-pass behavior -and thus the PSNR loss-, and the more likely is that the blurring effect is noticeable.

Therefore, an algorithm that allows for controlling the α parameter on a block basis during the encoding process is necessary. The goal is to reduce the flicker distortion while controlling the loss in the PSNR for each block, so that the blurring effect remains (to some extent) unnoticeable.

2.3.1. Problem formulation and proposed solution

For each block i, we would like to minimize the flicker distortion $D_{flicker,i}$ (where we have removed the dependence with the frame number for convenience), while keeping the PSNR loss ($PSNR_{loss,i}$) under a given user-defined threshold, $PSNR_{loss,tar}$:

$$\min_{\alpha_i} \left\{ D_{flicker,i}(\alpha_i) \right\} \text{ subject to } PSNR_{loss,i}(\alpha_i) \leq PSNR_{loss,tar}. \quad (5)$$

To solve (5), we have experimentally studied the relationship between α_i, $PSNR_{loss,i}$, and $D_{flicker,i}$ from data collected from various sequences. For these experiments, we have implemented the proposed method in a reference software of the H.264/AVC video coding standard [9]. The results for three examples (Container at QP 36, Coastguard at QP 32, and Akiyo at QP 40) are shown in Fig. 2. Specifically, the average PSNR loss over the blocks, $PSNR_{loss}$, as a function of α is plotted in the left part of the Figure, while the accumulated flicker distortion $D_{flicker} = \sum_i D_{flicker,i}$ as a function of α is shown in the right part. It is worth noting that both measures have been calculated using only the filtered blocks.

Given that $D_{flicker}$ is a monotonic increasing function of α, the minimum α_i able to meet the PSNR loss restriction (the α_i that makes the $PSNR_{loss,i} = PSNR_{loss,tar}$), will produce the minimum flicker distortion subject to the PSNR loss constraint, thus solving (5). In this way, if we were able to model $PSNR_{loss,i}$ as a function of α_i for each block i, we could obtain the α_i value that meets $PSNR_{loss,i} = PSNR_{loss,tar}$.

As can be inferred from the curves shown in Fig. 2, the relationship between α and $PSNR_{loss}$ can be well fitted by means of a linear model. To obtain *on-the-fly* the parameters that define the linear model, we have to evaluate at least two (α_i, $PSNR_{loss,i}$) points for each block i. Once the parameters of the model are estimated, we can use it to obtain $\alpha_{tar,i}$, the value of α_i that allows for achieving $PSNR_{loss,i} = PSNR_{loss,tar}$ in the block i.

Fig. 3 shows an illustrative example of how the proposed method selects $\alpha_{tar,i}$. We use an I-frame of the sequence *Akiyo* at QP 40 with $PSNR_{loss,tar} = 1dB$. The higher the $\alpha_{tar,i}$, the lighter the color. As can be seen, the areas with more detail are filtered with higher $\alpha_{tar,i}$ values, as expected.

The complete algorithm is summarized in Algorithm 1. It is worth noticing that when the obtained $\alpha_{tar,i}$ produces a higher $PSNR_{loss,i}$ than the $PSNR_{loss,tar}$, the non-filtered version will be selected as the best solution.

Algorithm 1 Proposed coding process.

Require: $PSNR_{loss,tar}$: target PSNR loss
Require: $L = 2$: number of available α_i values for each block to estimate the $PSNR_{loss,i}(\alpha_i)$ linear model
Require: I: number of blocks
for $\forall i \in I$ **do**
  Calculate the non-filtered version of the reconstructed block and its flicker measure $D_{flicker,i,non-fil}$
  for $\forall l \in L$ **do**
    Calculate the filtered version of the reconstructed block
    Store the PSNR loss, $PSNR_{loss,i,l}$
  end for
  Calculate the parameters of the linear model
  Calculate $\alpha_{tar,i}$ using the $PSNR_{loss,tar}$ in the previous model
  Calculate the filtered version of the reconstructed block with the $\alpha_{tar,i}$ and its flicker measure $D_{flicker,i,fil}$
  if $D_{flicker,i,fil} < D_{flicker,i,non-fil}$ **then**
    Select as best the filtered version of the block
  else
    Select as best the non-filtered version of the block
  end if
end for

Table 1. Test conditions

Coding options	First evaluation	Second evaluation
Profile	Main	Main
Frame rate	25	30
RD Optimization	Enabled	Enabled
GOP pattern	IPPP	IPPP
QP values	32, 36, and 40	32, 36, and 40
Intra period	25	5
Symbol Mode	CABAC	CABAC
Number of Reference Frames	1	1
Frames to be encoded	100	100

3. EXPERIMENTS AND RESULTS

To assess the performance of the proposed method, it was integrated into the H.264 reference software JM15.1 [10].

We start by comparing two versions of the proposed approach: the first one uses a fixed α value, while the second one uses an α_i parameter that is estimated *on-the-fly*, on a block-by-block basis, to reduce flicker while meeting a $PSNR_{loss,tar}$ constraint. The test conditions are summarized in the second column of Table 1 (labeled as "First evaluation"). The experiments were conducted using a set of nine sequences of two resolutions (CIF, and SD), listed in Table 2. For the fixed-α version (Fα), we use $\alpha = 0.7$. For the *on-the-fly* estimated-α_i version (OEα), we have selected two PSNR loss constraints: $PSNR_{loss,tar} = 1dB, 2dB$.

Table 2 shows the results of the two compared versions of the algorithm [1]. We provide two performance indicators: flicker reduction and $PSNR_{loss}$. The flicker reduction (FR) has been computed in relative terms with respect to the flicker distortion of the reference H.264/AVC implementation, as follows:

$$FR(\%) = \frac{D_{flicker}(JM15.1) - D_{flicker}(Method)}{D_{flicker}(JM15.1)} \times 100. \quad (6)$$

It is worth noticing that FR has been computed taking into account only the filtered blocks, and averaging over the three considered QP values. The $PSNR_{loss}$ has been also calculated with respect to the reference software and averaged over the processed blocks and the QP values.

Some conclusions can be extracted from the results in Table 2. First, it can be seen that the proposed OEα actually achieves a $PSNR_{loss}$ close to the target $PSNR_{loss,tar}$. Consequently, we are minimizing $D_{flicker}$ subject to the $PSNR_{loss}$ constraint. Furthermore, the FR follows the expected behavior. In particular, the lower $PSNR_{loss,tar}$, the lower FR, and vice-versa. Second, let us compare Fα (columns 2 and 3) with OEα with $PSNR_{loss,tar} = 2dB$ (columns 4 and 5). As can be seen, both methods achieve a very similar FR. Nevertheless, OEα obtains an average $PSNR_{loss}$ of 1.5 dB, while Fα obtains an average $PSNR_{loss}$ of 2.4 dB. Additionally, the standard deviation of $PSNR_{loss}$ is much lower for the OEα (0.2 dB) than for the Fα (1 dB). The reason is that Fα does not take into account the video content. Therefore, in some sequences like Flower or Football, filtering high-frequency blocks with a fixed α value produces significant PSNR losses.

We have also compared our proposal with a state-of-the-art flicker reduction method proposed by Chun et al. [1]. To conduct a fair comparison, we have replicated the test conditions in [1]. The third column of Table 1 (labeled as "Second evaluation") summarizes these conditions. The results are shown in Table 3. It is worth noticing that the method by Chun et al. incurs in a negligible $PSNR_{loss}$, but, on the other hand, the achieved FR is quite modest. This is because Chun's method relies on a modified mode decision process that takes into account the flicker distortion in the RDO, but this strategy does not provide enough degrees of freedom to significantly reduce the flicker distortion. To conduct a fair comparison, we have configure the proposed method to achieve very low PSNR losses; in particular, we have used $PSNR_{loss,tar} = 0.2dB$. The results in Table 3 show that the proposed method achieves higher flicker reductions than those of Chun's method for almost all the sequences. On the other hand, our PSNR losses are slightly higher than those obtained by Chun's method. However, these losses are kept low and close to the target.

Finally, we must highlight that our method is able to properly work in different test conditions, as can be inferred from the two types of assessments presented in this section.

[1] Some visual examples obtained using the proposed method are available at http://www.tsc.uc3m.es/∼ajimenez/ICASSP2013.

Table 2. Comparative performance evaluation of two versions of the proposed method.

	Fα ($\alpha = 0.7$)		OEα ($PSNR_{loss,tar} = 2dB$)		OEα ($PSNR_{loss,tar} = 1dB$)	
Sequence	FR	$PSNR_{loss}$	FR	$PSNR_{loss}$	FR	$PSNR_{loss}$
Container (CIF)	31.53	2.6	41.10	1.5	27.41	0.9
Coastguard (CIF)	26.70	2.6	31.92	1.7	20.07	0.9
Bridge-far (CIF)	50.73	1.2	52.68	1.1	40.88	0.8
Bridge-close (CIF)	24.77	1.8	44.07	1.4	31.16	0.8
Flower (CIF)	34.30	3.3	20.03	1.6	12.74	0.8
Nature (CIF)	31.29	1.1	28.66	1.7	23.03	1.0
Akiyo (CIF)	34.91	2.3	26.38	1.7	17.16	1.0
Football (CIF)	31.11	4.2	33.68	1.4	23.73	0.8
Corvette (SD)	31.31	2.6	33.26	1.6	23.46	1.0
Average	**32.96**	**2.4**	**34.64**	**1.5**	**24.40**	**0.9**

Table 3. Comparative performance evaluation vs. [1].

	[1]		OEα ($PSNR_{loss,tar} = 0.2dB$)	
Sequence	FR	$PSNR_{loss}$	FR	$PSNR_{loss}$
Container (CIF)	3.37	0	12.16	0.2
Coastguard (CIF)	0	0	8.96	0.3
Bridge-far (CIF)	0.24	0	38.67	−0.1
Bridge-close (CIF)	5.44	0	16.95	0.2
Flower (CIF)	17.81	−0.1	4.77	0.3
Nature (CIF)	6.67	0	26.28	0.1
Akiyo (CIF)	10.60	0	12.55	0.2
Football (CIF)	9.97	0	9.08	0.4
Corvette (SD)	10.82	0	23.49	0.6
Average	**7.21**	**0**	**16.99**	**0.2**

4. CONCLUSIONS AND FURTHER WORK

In this work, we have proposed a flicker reduction method based on a temporal motion-guided low-pass filtering. Specifically, this filtering process reduces the difference between reconstructed pixels of the same region in consecutive frames, reducing flicker and improving the visual quality. Moreover, to avoid the appearance of side-effects owing to the low-pass filtering, such as blurring, we have proposed an *on-the-fly* method to control the PSNR loss.

We have proved experimentally that our method achieves high flicker reductions. For example, fixing a maximum PSNR loss constraint of 2 dB in the filtered blocks, the FR remains above 34 %. Moreover, we have shown that our method is able to properly work in different coding conditions, outperforming a state-of-the-art flicker reduction method.

An interesting future line of research will focus on reducing the computational cost associated with the *on-the-fly* selection of the filter parameter.

5. REFERENCES

[1] Seong Soo Chun, Jung-Rim Kim, and Sanghoon Sull, "Intra prediction mode selection for flicker reduction in H.264/AVC," *Consumer Electronics, IEEE Transactions on*, vol. 52, no. 4, pp. 1303 –1310, Nov. 2006.

[2] K. Chono, Y. Senda, and Y. Miyamoto, "Detented quantization to suppress flicker artifacts in periodically inserted intra-coded pictures in H.264 video coding," in *Image Processing, 2006 IEEE International Conference on*, Oct. 2006, pp. 1713 –1716.

[3] Hua Yang, J.M. Boyce, and A. Stein, "Effective flicker removal from periodic intra frames and accurate flicker measurement," in *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, Oct. 2008, pp. 2868 –2871.

[4] G.J. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *Signal Processing Magazine, IEEE*, vol. 15, no. 6, pp. 74 –90, Nov. 1998.

[5] A. Matsumura, S. Naito, R. Kawada, and A. Koike, "Effective rate control method for minimizing temporal fluctuations in picture quality applicable for MPEG-4 AVC/H.264 encoding," in *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, Sept. 2005, vol. 1, pp. I – 569–72.

[6] X.Fan, W.Gao, Y.Lu, and D.Zhao, "Flicking Reduction in All Intra Frame Coding," JVT-E070, ISO/IEC MPEG and ITU-T VCEG Joint Video Team, Geneva, Oct. 2002.

[7] D.T. Vo, T.Q. Nguyen, Sehoon Yea, and A. Vetro, "Adaptive fuzzy filtering for artifact reduction in compressed images and videos," *Image Processing, IEEE Transactions on*, vol. 18, no. 6, pp. 1166 –1178, Jun. 2009.

[8] Jie Xiang Yang and Hong Ren Wu, "Robust filtering technique for reduction of temporal fluctuation in H.264 video sequences," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 20, no. 3, pp. 458 –462, Mar. 2010.

[9] T. Wiegand, G.J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h.264/avc video coding standard," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 13, no. 7, pp. 560 –576, Jul. 2003.

[10] JVT H.264/AVC reference software v.15.1 [online], "http://iphome.hhi.de/suehring/tml/download/old_jm/," .