# FLICKER REDUCTION FOR MOTION JPEG2000 USING WAVELET THRESHOLDING

*Xi ZHANG*

Dept. of Information and Comm. Eng.
University of Electro-Communications
Chofu-shi, Tokyo, 182-8585 Japan
E-mail: xiz@ice.uec.ac.jp

*Atsushi TANAKA*

Dept. of Information and Comm. Eng.
University of Electro-Communications
Chofu-shi, Tokyo, 182-8585 Japan
E-mail: atanaka@ice.uec.ac.jp

**ABSTRACT**

Flickering is a temporal visual artifact that impairs significantly the visual quality of the compressed videos. Motion JPEG2000 is a wavelet-based intraframe video codec, and has also a drawback of flicker artifact. In this paper, we introduce a new approach to flicker reduction of videos compressed by Motion JPEG2000 by using wavelet thresholding. We view this flicker reduction as a denoising problem in the temporal domain, and apply the excellent denoising property of wavelet thresholding in solving this flicker reduction problem. Compared with the conventional methods reported in the literature, It is shown from our experimental results that our wavelet-based method both suppresses the flicker artifact and achieve an improvement in PSNR and in $TI$.

***Index Terms*—** Flicker reduction, Wavelet thresholding, Motion JPEG2000

## 1. INTRODUCTION

Motion JPEG2000 is a wavelet-based intraframe video codec [1], [2], and has been adopted as the standard of digital cinema in Digital Cinema System Specification by DCI (Digital Cinema Initiative, LLC) [3]. Motion JPEG2000 employs the still image compression technique (JPEG2000) on each frame of a video without using any interframe coding techniques such as motion compensation, and thus has many advantages such as edibility, high quality requirement and so on.

Like MPEG video codecs, flicker artifact can appear in a video compressed by Motion JPEG2000 even at medium bit rate [11]. It is perceptible as a small variation in pixel value, and is extremely visible in static regions without moving objects [13]. This visual artifact impairs significantly the visual quality of the compressed videos. Therefore, it is important to suppress the flicker artifact and to improve the visual quality of videos.

There are a few methods addressing flicker reduction [13]∼[18]. These methods can be categorized into two types: (a) encoder-based [13], [16], [17], and (b) post-decoder [14], [15], [18] approaches. The encoder-based approaches increase or decrease the magnitude of wavelet coefficients in bit-plane [13], or modify the post-compression rate control scheme [16] and adjust the block truncation length [17] to suppress the flicker artifact. Since those processings were done in the encoder, however, the compression performance may be more or less influenced. On the other hand, The post-decoder approaches handle the decoded videos and thus do not influence the performance of Motion JPEG2000. In this paper, we address the post-decoder scheme also. In [14] and [15], the static and moving regions are firstly recognized in each frame of the decoded videos, then the flicker is estimated and suppressed in the static regions by using the corresponding pixel values of the forward and backward frames. In [18], the information of the modified forward frame is used to refine the pixel values in the current frame according to the difference between these two frames. However, the quality of the videos (e.g., PSNR) may be more or less influenced, although the flicker artifact can be reduced to a certain extent.

In this paper, we introduce a new approach to flicker reduction of videos compressed by Motion JPEG2000 by using wavelet thresholding. The flicker artifact can be thought of as Gaussian noise in the temporal domain, which is due to both lack of shift invariance of DWT (discrete wavelet transform) and quantization. We view this flicker reduction as a denoising problem in the temporal domain, and apply the wavelet thresholding algorithm in solving this flicker reduction problem. The wavelet thresholding algorithm was proposed in [5]∼[8] by Donoho *et al.*, and had exhibited the excellent denoising property. It is shown from our experimental results that our wavelet-based method suppresses the flicker artifact and achieve an improvement in PSNR and in $TI$ as well, compared with the conventional methods proposed in [14], [15] and [18].

## 2. CAUSE OF FLICKERING

Flicker artifact in Motion JPEG2000 was originally studied in [11] by Kuge, and investigated in [12] by Kato *et al.* and in [13] by Becker *et al.*. In the encoder of JPEG2000 shown in Fig.1, DWT is followed by quantization and EBCOT. The EBCOT rate allocation scheme performs the post-compression (PC) quantization through code-block bit-plane truncation to obtain any desired rates. Becker *et al.* analyzed in [13] that the flicker artifact was caused by quantization and PC quantization. A small change in magnitude of wavelet coefficients can be amplified by quantization. Furthermore, the code-block truncation length often differs from frame to frame due to this magnitude change. This is because the EBCOT runs for each frame independently. Therefore, the flicker artifact can appear even in the static regions, since the code-blocks are truncated at different lengths. The magnitude changes of wavelet coefficients may be due to the changes of the pixel values in videos (temporal content

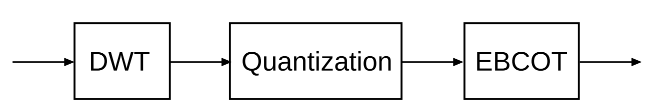


**Fig. 1**. JPEG2000 Encoder.

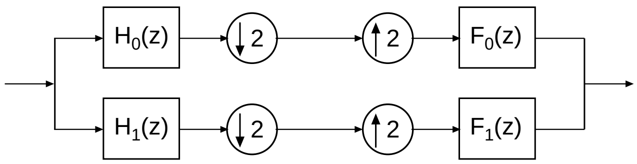


**Fig. 2**. Two band filter bank.

variations). However, Becker *et al.* had carried out experiments on synthetic videos and pointed out that a very small motion can cause flicker too. This means that a very small motion can cause the changes in magnitude of wavelet coefficients besides a spatial movement. If the wavelet coefficients do not change in magnitude except a spatial movement, we cannot observe flicker artifact in the synthetic videos. Why does a small motion cause the magnitude changes of wavelet coefficients? It is due to lack of shift invariance of DWTs [4].

It is well-known that DWTs are implemented by a two band filter bank shown in Fig.2. In Fig.2, analysis filters $H_0(z)$ and $H_1(z)$ are followed by a decimator. This decimation is not a shift-invariant operation. If an input signal to the decimator shifts just a few samples, its output is totally different, depending on the number of sample shifted. In general, the frequency responses of $H_0(z)$ and $H_1(z)$ are not ideal, thus the decimation introduces an aliasing component. In Motion JPEG2000, DWTs are performed in the spatial domain (intraframe). Therefore, when a motion exists, i,e., a spatial moving of objects, the wavelet coefficients will change in magnitude due to the aliasing component, which cause flicker artifact. As a result, we conclude that the flicker artifact is caused by both lack of shift invariance of DWTs and quantizations including PC quantization.

## 3. FLICKER REDUCTION USING WAVELET THRESHOLDING

In the previous section, we have analyzed that the flicker artifact is caused by a small change in magnitude of wavelet coefficients, and the magnitude change is due to lack of shift invariance of DWTs. The amount of this change is dependent on the aliasing component, which is decided by the frequency responses of analysis filters $H_0(z)$ and $H_1(z)$. In general, $H_0(z)$ and $H_1(z)$ are a pair of lowpass and highpass filters well-designed, for example, Daubechies-5/3 and Daubechies-9/7 wavelets used in JPEG2000. Thus the amount of aliasing component is relatively small. Flicker can be observed as a temporal visual artifact in a compressed video, where quantization noise is introduced by compression. Therefore, we can think of the flicker artifact as Gaussian noise in the temporal domain and view this flicker reduction as a denoising problem.

In this paper, we apply the excellent denoising property of wavelet thresholding in solving this flicker reduction problem. The wavelet thresholding algorithm was proposed in [5]∼[8] by Donoho *et al.*. In the wavelet domain, noise is uniformly spread throughout the wavelet coefficients, while most of the image information is concentrated in the few largest ones (sparsity of the wavelet representation). Thus, the most straightforward way of distinguishing information from noise in the wavelet domain is to threshold the wavelet coefficients. Moreover, the better denoising performance was reported in [9] by Lang *et al.* using overcomplete rather than complete DWTs. The overcomplete DWTs do not perform decimation operation and then are shift-invariant. Therefore, we use the overcomplete (undecimated) DWTs in this paper.

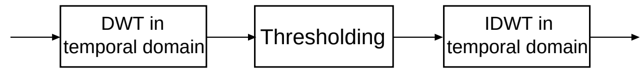


**Fig. 3**. Flicker reduction using wavelet thresholding.

As shown in Fig.3, our wavelet-based method does, 1) decompose the decoded videos by using the overcomplete DWTs in the temporal domain (interframe), 2) perform thresholding on its wavelet coefficients, and 3) reconstruct the flicker-reduced videos from the thresholded coefficients by computing IDWTs (inverse discrete wavelet transforms).

In the following, we illustrate the overcomplete wavelet decomposition, for example, by using the lifting scheme of Daubechies-5/3 wavelet. Let $x[n,m,t]$ be a lightness value at pixel $(n,m)$ in frame $t$ of the decoded video.

In the first stage ($j = 1$), we compute a detailed (high-frequency) component $d_1[n,m,t]$ by

$$d_1[n,m,t] = \frac{1}{2}x[n,m,t] - \frac{1}{4}(x[n,m,t-1] + x[n,m,t+1]), \quad (1)$$

and approximated (low-frequency) component $a_1[n,m,t]$ by

$$a_1[n,m,t] = x[n,m,t] + \frac{1}{2}(d_1[n,m,t-1] + d_1[n,m,t+1]). \quad (2)$$

In the $j$th stage ($j \geq 2$), all filters used here are upsampled by a factor of $2^{j-1}$ compared with those in the first stage, that is, $H_0(z^{2^{j-1}})$ and $H_1(z^{2^{j-1}})$. Thus, we have

$$\begin{aligned} d_j[n,m,t] = \frac{1}{2}a_{j-1}[n,m,t] - \frac{1}{4}(a_{j-1}[n,m,t-2^{j-1}] \\ + a_{j-1}[n,m,t+2^{j-1}]), \end{aligned} \quad (3)$$

$$\begin{aligned} a_j[n,m,t] = a_{j-1}[n,m,t] + \frac{1}{2}(d_j[n,m,t-2^{j-1}] \\ + d_j[n,m,t+2^{j-1}]), \end{aligned} \quad (4)$$

where $d_j[n,m,t]$ are a set of wavelet coefficients at scale $j$.

Next, we perform thresholding operation on the wavelet coefficients $d_j[n,m,t]$. Of the various thresholding strategies, hard-thresholding and soft-thresholding are the most popular.

For hard-thresholding, the wavelet coefficients are discarded if its magnitude is not larger than a threshold value $\tau$;

$$\tilde{d}_j[n,m,t] = \begin{cases} d_j[n,m,t] & |d_j[n,m,t]| > \tau \\ 0 & |d_j[n,m,t]| \leq \tau \end{cases}, \quad (5)$$

while for soft-thresholding, we have

$$\tilde{d}_j[n,m,t] = \begin{cases} d_j[n,m,t] - \tau & d_j[n,m,t] > \tau \\ 0 & |d_j[n,m,t]| \leq \tau \\ d_j[n,m,t] + \tau & d_j[n,m,t] < -\tau \end{cases}, \quad (6)$$

where the magnitude of $d_j[n,m,t]$ is reduced by a value of $\tau$ when it is larger than $\tau$. The threshold value $\tau$ can be estimated according to the noise variance.

Finally, the flicker-reduced video is reconstructed from the thresholded coefficients $\tilde{d}_j[n,m,t]$ by using IDWT in the temporal domain.

## 4. EXPERIMENTAL RESULTS

In this section, we present some experimental results to demonstrate the effectiveness of the wavelet-based flicker reduction method proposed in this paper. We have tested our wavelet-based method on some sequences listed in Table 1. First, we have compressed these videos at a compression ratio of $30 : 1$ by using Motion JPEG2000, whose software is downloaded from the OpenJPEG Web [19], and observed flicker artifacts in the compressed videos. We have then performed our method on the RGB components of the decoded videos separately. We used Daubechies-5/3 wavelet to decompose them to 3 stages in the temporal domain. Both of hard-thresholding and soft-thresholding were used to modify the wavelet coefficients. For a sequence "container" of 300-frame $352 \times 288$ pixels, the threshold value was set to $\tau = 15$ for hard-thresholding and to $\tau = 7$ for soft-thresholding.

In the following, we evaluate subjectively and objectively the quality of the flicker-reduced videos by our method, and compare them with the conventional methods in [14], [15].

### 4.1. Subjective Evaluation Based on Luminance Change

To evaluate subjectively the flicker reduction effects, we firstly examine luminance value changes before and after flicker reduction. The luminance value changes at pixel $(300, 80)$ of "container" are shown in Fig.4. It is seen in Fig.4 that the luminance values change greatly in the compressed video compared to ones in the original video, which are observed as flicker artifacts. Our experimental results show that its fast variation (flicker) has been suppressed by our wavelet-based method, while the slow change (motion) is almost preserved. However, the conventional method in [14] and [15] cannot suppress this fast variation completely, because only the corresponding pixel value of the forward or backward frame was used. Moreover, we have observed that the flicker artifacts are not visible, making the flicker-reduced video look visually much more pleasing by using our wavelet-based method, whereas it still remains in [14] and [15]. (See our results at http://www.xiz.ice.uec.ac.jp/MJ2K/).

### 4.2. Objective Evaluation Using PSNR

We evaluate the objective quality of the flicker-reduced videos with the help of PSNR, although PSNR may not be a reliable metric for measuring flicker artifacts. The average PSNRs of RGB components at each frame of "container" are shown in Fig.5. It is clear in Fig.5 that our wavelet-based method including hard-thresholding and soft-thresholding achieves an improvement in PSNR, whereas the conventional method in [14] and [15] has a little improvement. The average of PSNR over all frames is 27.425dB for the compressed video "container", and 27.545dB for the conventional method, while 27.939dB for hard-thresholding and 27.934dB for soft-thresholding. The average of PSNR for other videos are shown in Table 1 also. Therefore, it is noteworthy to point out that our wavelet-based method can improve the objective quality of the compressed videos also.

### 4.3. Objective Evaluation Using TI

Temporal information $TI[n, m, t]$ is defined in [10] as

$$TI[n, m, t] = x[n, m, t] - x[n, m, t-1], \quad (7)$$

which describes the difference (movement) between two adjacent frames $x[n, m, t]$ and $x[n, m, t-1]$.

**Table 1**. The average of PSNR (dB) and $TI_{RMSE}$ results

| sequence | Compressed | [14],[15] | [18] | Proposed |
|---|---|---|---|---|
| coastguard | 27.074 | 27.079 | 26.865 | **27.407** |
| | 13.033 | 12.733 | 12.383 | **11.966** |
| container | 27.425 | 27.545 | 27.794 | **27.939** |
| | 5.706 | 4.525 | 3.190 | **2.855** |
| crew | 29.042 | 28.977 | 28.438 | **29.302** |
| | 10.417 | 10.295 | 10.604 | **9.875** |
| football | 28.193 | 27.842 | 27.715 | **28.326** |
| | 12.609 | 12.823 | 12.867 | **12.360** |
| foreman | 28.570 | 28.593 | 28.080 | **28.830** |
| | 9.690 | 9.301 | 9.206 | **8.942** |
| ice | 32.899 | 33.080 | 32.660 | **33.548** |
| | 5.866 | 5.371 | 5.442 | **5.077** |
| tempete | 22.839 | 22.873 | 23.021 | **23.248** |
| | 14.667 | 14.280 | 12.151 | **11.749** |

Quality parameters based on temporal information $TI[n, m, t]$ can be interpreted as indicating added or lost motion in the flicker-reduced video compared to the original video. The RMSE (root mean square error) of $TI[n, m, t]$ between two videos is computed by

$$TI_{RMSE}[t] = \sqrt{\frac{1}{MN}\sum_{n=1}^{N}\sum_{m=1}^{M}(TI_o[n, m, t] - TI_r[n, m, t])^2}, \quad (8)$$

where $TI_o[n, m, t]$ and $TI_r[n, m, t]$ are temporal information of the original and the flicker-reduced videos, respectively, and $MN$ is a total number of pixels in each frame. A smaller $TI_{RMSE}[t]$ indicates that the motion in the flicker-reduced video is more faithful to that in the original video.

The average $TI_{RMSE}[t]$ of RGB components at each frame of "container" are shown in Fig.6. It is seen in Fig.6 that our wavelet-based method has a further improvement in $TI_{RMSE}[t]$, although the conventional method in [14] and [15] had improved $TI_{RMSE}[t]$. The average of $TI_{RMSE}[t]$ over all frames is 5.706 for the compressed video "container", and 4.525 for the conventional method, while 2.855 for hard-thresholding and 2.863 for soft-thresholding. The average $TI_{RMSE}[t]$ for other videos are shown in Table 1[1]. It is clear that the flicker-reduced videos by our method are more faithful to the original videos, compared with the conventional methods.

## 5. CONCLUSIONS

In this paper, we have analyzed that the flicker artifact in Motion JPEG2000 is caused by both lack of shift invariance of DWTs and quantizations including PC quantization. Furthermore, we have introduced a new approach to flicker reduction of videos compressed by Motion JPEG2000 by using wavelet thresholding. Since the flicker artifact can be thought of as Gaussian noise in the temporal domain, we view this flicker reduction as a denoising problem in the temporal domain, and apply the wavelet thresholding algorithm in solving this flicker reduction problem. It has been shown from our experimental results that our wavelet-based method suppresses the flicker artifact and achieve an improvement in PSNR and in $TI$ as well, compared with the conventional methods.

[1] In Table 1, the upper and lower columns are PSNR and $TI$ respectively.

## 6. REFERENCES


[1] ISO/IEC 15444: Information technology — JPEG 2000 image coding system, 2000.

[2] D. S. Taubman, and M. W. Marcellin, “JPEG2000: Image compression fundamentals, standards and practice,” Kluwer Academic Publishers, Boston, 2002.

[3] Digital Cinema Initiative, LLC, Member Representatives Committee, “Digital cinema system specification V1.0,” July 2005.

[4] I. Daubechies, *Ten Lectures on Wavelets*, Philadelphia, PA: SIAM, 1992.

[5] D. L. Donoho, and I. M. Johnstone, “Ideal spatial adaptation via wavelet shrinkage,” Biometrika, vol.81, pp.425–455, Sep. 1994.

[6] D. L. Donoho, and I. M. Johnstone, “Adapting to unknown smoothness via wavelet shrinkage,” J. Amer. Statist. Assn., vol.90, pp.1200–1224, 1995.

[7] D. L. Donoho, I. M. Johnstone, G. Kerkyacharian, and D. Picard, “Wavelet shrinkage: asympotia? with discussion,” J. Royal Statistical Soc., Series B, vol.57, no.2, pp.301–369, 1995.

[8] D. L. Donoho, “Denoising by soft-thresholding,” IEEE Trans. Inform. Theory, vol.41, no.5, pp.613–627, May 1995.

[9] M. Lang, H. Guo, J. E. Odegard, C. S. Burrus, and R. O. Well Jr., “Noise reduction using an undecimated discrete wavelet transform,” IEEE Signal Processing Letters, vol.3, no.1 pp.10–12, Jan. 1996

[10] J. Berts, and A. Persson, “Objective and subjective quality assessment of compressed digital video sequences,” Master Thesis, Department of Signals and Systems, Chalmers University of Technology, Gö̈teborg, Sweden, 1998.

[11] T. Kuge, “Wavelet picture coding and its several problems of the application to the interlace HDTV and the ultra-high definition images,” Proc. ICIP2002, vol.3, pp.217–220, June 2002.

[12] T. Kato, S. Tanabe, H. Watanabe, and H. Tominaga, “On a relationship between flicker artifacts and quantization error for Motion JPEG2000,” Proc. FIT2003, J–039, Sep. 2003.

[13] A. Becker, W. Chan, and D. Poulouin, “Flicker reduction in intraframe codecs,” Proc. DCC’04, pp.252–261, Mar. 2004.

[14] Y. Itani, and H. Watanabe, “A study on flicker reduction based on distinction between static and moving region in Motion JPEG 2000,” IEICE Technical report, 2005AVM49–14, pp.77–82, July 2005.

[15] Y. Itani, and H. Watanabe, “A study on flicker reduction using forward and backward frames in Motion JPEG 2000,” Proc. FIT2005, J–061, Sep. 2005.

[16] A. Leontaris, Y. Tonomura, and T. Nakachi, “Rate control for flicker artifact suppression in Motion JPEG2000,” Proc. ICASSP2006, vol.2, pp.41–44, May 2006.

[17] A. Leontaris, Y. Tonomura, T. Nakachi, and P. Cosman, “Flicker suppression in JPEG2000 using segmentation-based adjustment of block truncation lengths,” Proc. ICASSP2007, vol.1, pp.1117–1120, Apr. 2007.

[18] Y. Murata, H. Kikuchi, and S. Muramatsu, “Reduction in perceptual distortion for wavelet transform-based video decoding,” IEICE Technical report, MoMuC2007–30, pp.31–36, July 2007.

[19] OpenJPEG (http://www.openjpeg.org/)


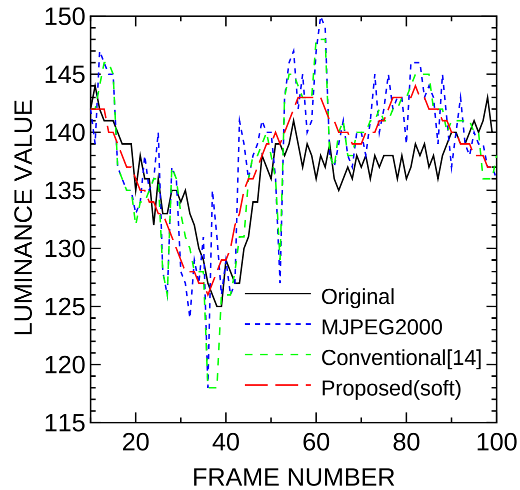


**Fig. 4**. Luminance value changes at pixel (300,80) of “container”.

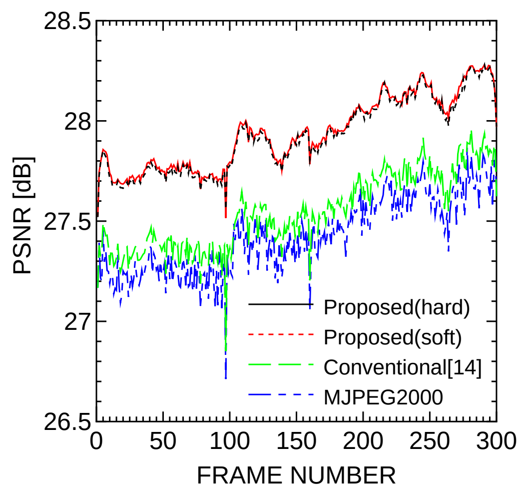


**Fig. 5**. PSNR results of “container”.

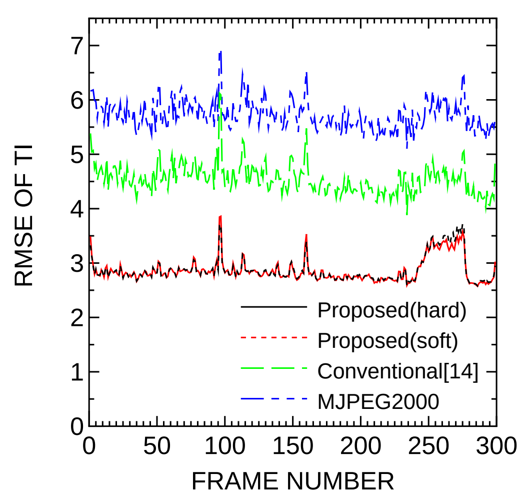


**Fig. 6**. $TI_{RMSE}$ results of “container”.