Bit-Depth Scalable Coding Using a Perfect Picture and Adaptive Neighboring Filter*

LU Feng (陆 峰), ER Guihua (尔桂花)**, DAI Qionghai (戴琼海), XIAO Hongjiang (肖红江)

Department of Automation, Tsinghua University, Beijing 100084, China

Abstract: Bit-depth scalability is a new research field in the on-going scalable extension of the H.264/AVC (SVC) video coding standard. The key point is to accurately predict the enhancement layer, whose bit depth is 10 or more, from the 8 bit base layer. An improved inter-layer prediction scheme for bit-depth scalability was developed that ensures compatibility with the standard and improves the encoding efficiency. The scheme uses a “perfect” 8 bit picture with an adaptive neighbor filter whose coefficients are optimized by minimizing the block distortion between the 8 bit reconstructed picture and the “perfect” picture to achieve a more precise 10 bit prediction based on the filtered picture. Double arithmetic precision is used to further improve the encoding efficiency. Experimental results show that the scheme outperforms the recent joint video team (JVT) proposal in the joint scalable video model (JSVM).

Key words: H.264; SVC; bit-depth scalability; inter-layer prediction

Introduction

The co-existence of conventional 8 bit and higher bit-depth video has stimulated the demand for bit-depth scalable coding, especially in fields such as digital arts, medical image processing, and high quality computer games. Bit-depth scalable coding ensures that coded video sequences can naturally switch from 8 bit to higher bit depths. Technical difficulties occur because various video/image processing devices and different tone mapping methods may be used to generate the input 8 bit sequence from the higher bit sequence[1,2] and because the 8 bit to higher bit prediction is actually an inverse tone mapping process without prior knowledge of the method used for the initial 8 bit generation stage. Several methods have been proposed to tackle these problems based on the scalable extension of H.264/AVC (SVC)[3], such as the simplest linear scaling[4], the combinational scheme of scale factors and offset[5,6], and look-up table (LUT) methods[7]. Recent results have shown that the last method outperforms the others. The basic idea of the LUT method is to let both the encoder and decoder refer to the same table which assigns each possible 8 bit value a determined higher bit value. However, in practice, different tone mapping methods may cause two pixels to have very different higher bit values in the higher bit picture even though their 8 bit values in the corresponding 8 bit picture are the same. This alterable mapping between the 8 bit picture and the higher bit picture is the main cause of the prediction precision reduction for the LUT.

This paper presents a “perfect” 8 bit picture method that produces more consistent mappings. The picture is directly down-converted from the higher bit picture by the pre-calculated LUT and is then used to precisely predict the higher bit picture with the same LUT. An adaptive neighboring filter is used to filter the reconstructed 8 bit picture. The filter coefficients are adaptively calculated to minimize the difference between the reconstructed 8 bit picture and the “perfect” picture. Unlike in the original LUT method, this method does not use the reconstructed 8 bit picture but the filtered picture to predict the higher bit picture. Since the filtered picture will approximate the “perfect” picture, the inter-layer prediction precision can be improved.

1 Improved Bit-Depth Scalable Coding Scheme

1.1 Scheme architecture

The framework of the bit-depth scalable coding including the proposed inter-layer prediction scheme is shown in Fig. 1. Without loss of generality, there are assumed to be only two bit-depth layers to be encoded: the original 10 bit video sequence as the enhancement layer and the conventional 8 bit version of the same video sequence as the base layer. The base layer encoding process is completely compatible with H.264/AVC and the generated 8 bit reconstruction picture serves as the input for the inter-layer prediction. The LUT is calculated for the 10 bit layer encoding when the 8 bit reconstruction is available. This LUT is then used to generate the “perfect” picture from the 10 bit input picture. Then this “perfect” picture and the 8 bit reconstruction are used to design an adaptive neighboring filter. After filtering, the filtered picture and the LUT are used to give the final prediction for the 10 bit picture. The enhancement layer encoder then encodes the difference between the original and predicted 10 bit pictures, as well as the LUT information and the quantized filter coefficients, into the enhancement layer stream. The final output of the scalable encoder is a mixed stream of the 8 bit and 10 bit layers. As shown in the dashed box in Fig. 1, this scheme consists of down-converting, filtering, and up-converting steps. Therefore, this scheme is called the DFU scheme.

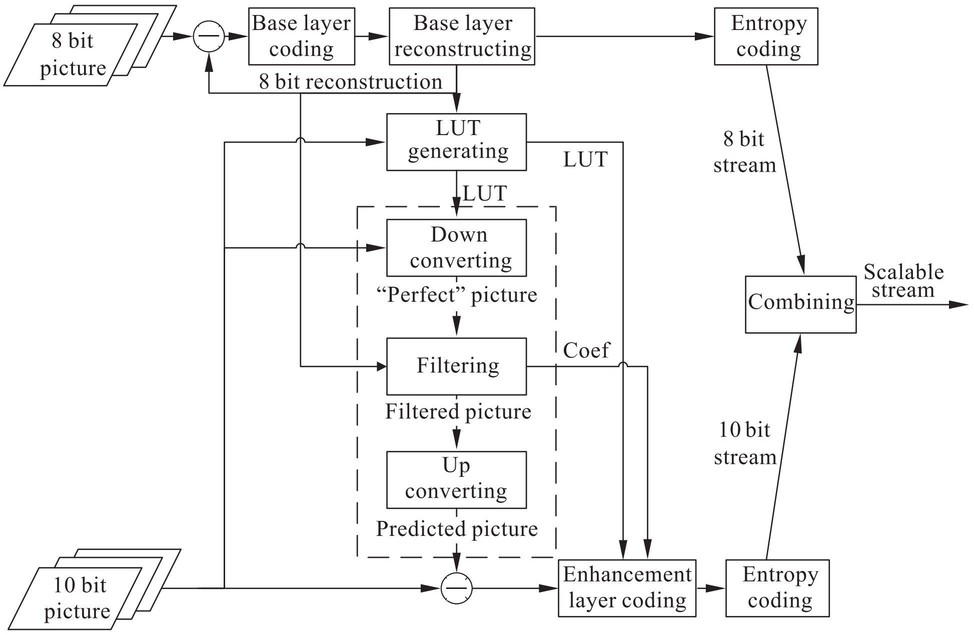


Fig. 1 Bit-depth scalable coding scheme

1.1.1 “Perfect” 8 bit picture by down-converting

The value of each component (i.e., Y, Cb, and Cr) is limited to the range of 0 to 255 in an 8 bit pixel and the range of 0 to 1023 in a 10 bit pixel. The LUT can only include 255 10 bit values because the numbers for the 10 bit and 8 bit values should be the same to form one-to-one mapping. Therefore, most of the 10 bit values cannot directly correspond to 8 bit values.

The prediction of the 10 bit pixel $V(x,y)$ from the pixel (x,y) in the 8 bit picture is then as follows. If $V(x,y)$ can be found in the LUT and its corresponding 8 bit value is v, then the value v is called the “perfect” 8 bit value of the pixel (x,y) because $V(x,y)$ can be predicted by the corresponding 10 bit value of v in the LUT without error.

However, as mentioned before, most 10 bit $V(x,y)$ cannot be found in the LUT. In such cases,

$$v = \arg\min_{\{v^-,v^+\}}(|V(x,y) - \mathrm{LUT}(v)|) \tag{1}$$

is used as the "perfect" value, where v^- and v^+ are constrained by

(a) $\mathrm{LUT}(v^-)$ and $\mathrm{LUT}(v^+)$ both exist in the LUT;

(b) $\mathrm{LUT}(v^-) < V(x,y)$ and $\mathrm{LUT}(v^+) > V(x,y)$;

(c) There are no other 10 bit values between $\mathrm{LUT}(v^-)$ and $\mathrm{LUT}(v^+)$ in the LUT.

This "perfect" v provides the most precise correlation to the 10 bit $V(x,y)$ with the minimum error $|V(x,y) - \mathrm{LUT}(v)|$.

An 8 bit picture is regarded as "perfect" only if each pixel in it has the "perfect" v relative to the 10 bit picture for that particular LUT. With this LUT, the "perfect" 8 bit picture provides the best prediction of the original 10 bit picture in terms of integer precision.

1.1.2 Adaptive neighboring filter

An adaptive neighboring filter was then used to filter the reconstructed 8 bit picture to make the filtered picture as similar to the "perfect" 8 bit picture as possible. This process makes the reconstructed picture more suitable for predicting the 10 bit picture.

Let $p_{x,y}$ be a pixel in the reconstructed picture and $q_{x,y}$ be the corresponding pixel in the "perfect" picture. The filter coefficients are $\{c_{i,j}\}$ and the band is N. After filtering, each pixel $p_{x,y}$ will be replaced by

$$\hat{p}_{x,y} = \sum\nolimits_{i=-N}^{N}\sum\nolimits_{j=-N}^{N} c_{i,j} \times p_{x+i,y+j} \tag{2}$$

The total difference for every pixel (x, y) between the filtered 8 bit picture and the "perfect" picture is minimized by optimizing the MSE expression,

$$\arg\min \sum\nolimits_{x}\sum\nolimits_{y}(\hat{p}_{x,y} - q_{x,y})^2 \tag{3}$$

Differentiating Eq. (3) by each $c_{m,n}$ and setting the differentials equal to zero gives

$$\sum_{x,y}\sum_{i,j} p_{x+i,y+j} \times p_{x+m,y+n} \times c_{i,j} = \sum_{x,y} q_{x,y} \times p_{x+m,y+n},$$
$$\forall m,n = -N,\ldots,N \tag{4}$$

Gaussian elimination is then used to solve for the filter coefficients $\{c_{i,j}\}$ from these linear equations. The solutions are always small fractions, so they are transformed into a set of integers with the integer quantizer q,

$$\hat{c}_{i,j} = \mathrm{floor}(c_{i,j} \times (1 << q) + 0.5) \tag{5}$$

The encoder will finally perform filtering by

$$\hat{p}_{x,y} = \mathrm{clip}((\sum\nolimits_{i=-N}^{N}\sum\nolimits_{j=-N}^{N} \hat{c}_{i,j} \times p_{x+i,y+j}) >> q, 0, 2^{10} - 1) \tag{6}$$

The integer filter coefficients $\{\hat{c}_{i,j}\}$ and q are later inserted into the encoded stream and transmitted to the decoder to enable the same filtering process during decoding.

1.1.3 Prediction by up-converting

The up-converting processes the filtered 8 bit picture using the LUT to predict the 10 bit picture. For the up-converting, each 8 bit pixel with the value, v_8, has a corresponding 10 bit value, v_{10}, that is found by simply looking up the corresponding 10 bit value in the LUT. For example, the 8 bit value 180 gives the 10 bit prediction 500 using the LUT shown in Fig. 2. Thus, the 10 bit picture from the filtered 8 bit picture will be much closer to the original 10 bit picture than the one up-converted using the standard inter-layer prediction scheme[7].

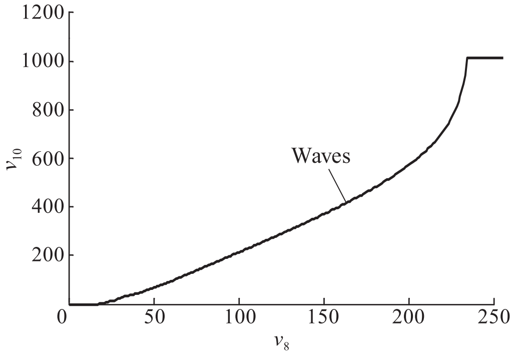


Fig. 2 Correspondence of 8 bit and 10 bit pixel values for the test sequence "Waves"

1.2 Double floating point arithmetic for proposed scheme

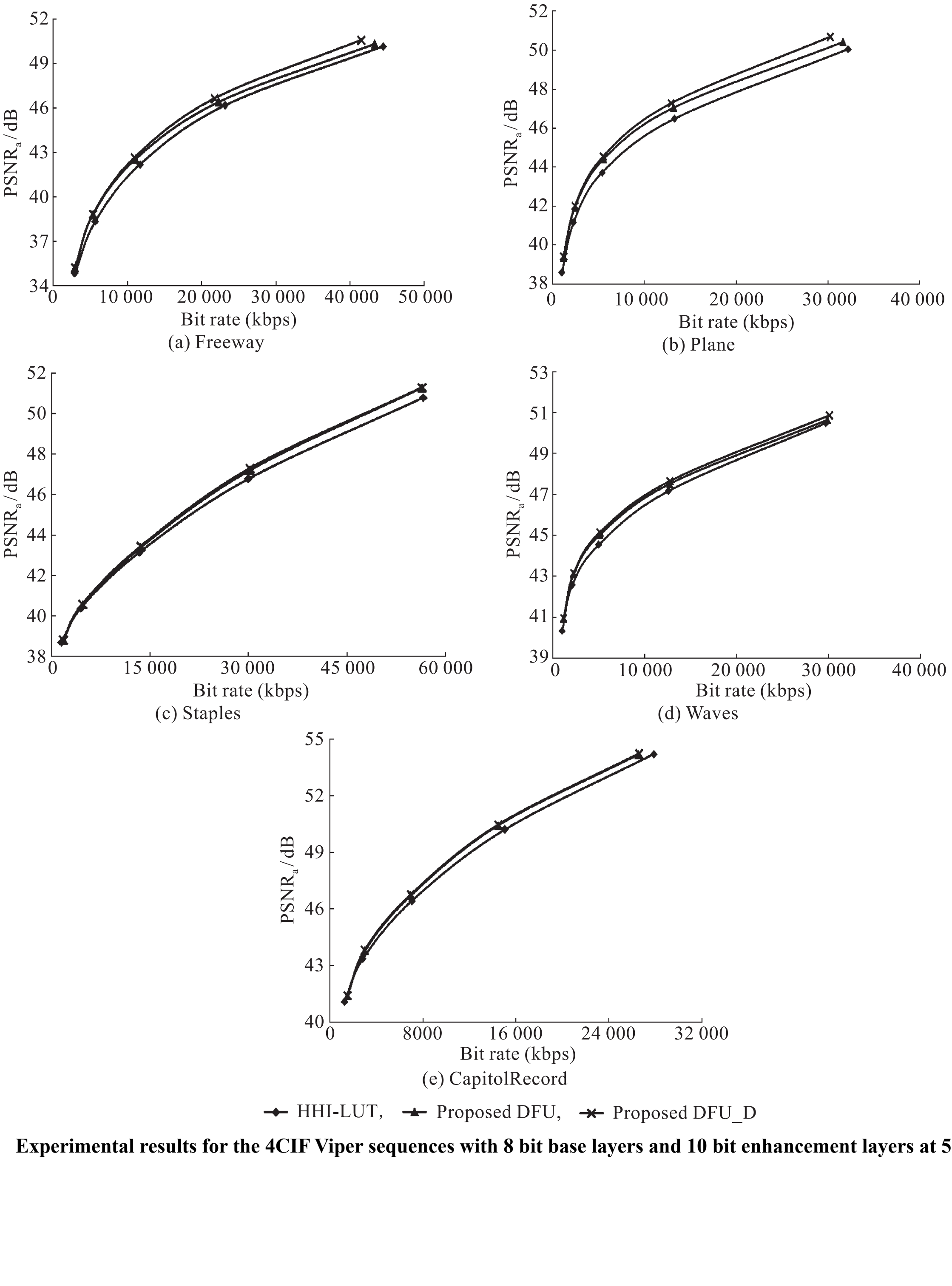
Section 1.1.1 described how the "perfect" 8 bit picture for inverse tone mapping is developed using a specific LUT. Even with this method 10 bit values which are not in the LUT still do not have a corresponding 8 bit value in the LUT, so there is some prediction error regardless of how the LUT is designed. This is due to the integer precision of the LUT and the magnitude of the 10 bit values (i.e., 1024) which are 4 times as large as the 8 bit values (i.e., 256).

Thus, double floating point arithmetic is used to calculate the "perfect" picture with an additional method to predict a 10 bit pixel $V(x,y)$ when $V(x,y)$ is not in the LUT. After using Eq. (1) to identify two 10 bit values, $\mathrm{LUT}(v^-)$ and $\mathrm{LUT}(v^+)$, an 8 bit integer v and its prediction $\mathrm{LUT}(v)$ are obtained which are the closest to but not equal to $V(x,y)$. However, any other 10 bit value between $\mathrm{LUT}(v^-)$ and $\mathrm{LUT}(v^+)$ will also be predicted by $\mathrm{LUT}(v)$. This conflict is corrected by calculating the "perfect" value using linear interpolation with double floating point precision,

$$v = v^- + \frac{V(x,y) - \mathrm{LUT}(v^-)}{\mathrm{LUT}(v^+) - \mathrm{LUT}(v^-)} \times (v^+ - v^-) \qquad (7)$$

Thus, the "perfect" picture becomes a double floating point picture rather than an 8 bit integer picture so that the mapping for each 10 bit value is unique. The subsequent filtering process later will also use the double floating point linear interpolation to map the filtered picture to the final 10 bit picture.

The memory for storing the double floating point "perfect" picture is reduced by combining the down-conversion and the filter coefficients calculation, with the filtering and up-conversion also combined together. Thus, there is no need to store the double floating point



(a) Freeway

(b) Plane

(c) Staples

(d) Waves

(e) CapitolRecord

Fig. 3 Experimental results for the 4CIF Viper sequences with 8 bit base layers and 10 bit enhancement layers at 50 Hz

values as they will be immediately used and dropped while they are available in the pipeline. This double floating point scheme is called DFU-D.

2 Experimental Results

These schemes were implemented in the recent JSVM codec[8] for comparison with the latest scheme reported by HHI[9]. The test sequences include Freeway, Plane, Staples, Waves, and CapitolRecords. All of these are standard sequences with nonlinear LUTs. The encoding used 2 bit depth layers, a frame rate of 50 Hz and QP = 12, 17, 22, 27, and 32.

The experimental results shown in Fig. 3 show that the average peak signal noise ratio ($\mathrm{PSNR_a}$) gain with the DFU-D scheme is 0.51 dB over the standard LUT scheme and the average bit rate saving is 12.23%. For the DFU scheme, the $\mathrm{PSNR_a}$ again is 0.39 dB and the bit rate saving is 9.47%. Thus, these schemes both increase the coding efficiency of the bit-depth scalable coding, with the double precision scheme being more effective for the inter-layer prediction.

In addition these schemes will have better performance when applied to sequences such as Plane which have larger error variances of “mismatch” during the mapping from their 8 bit to 10 bit pixel values. This “mismatch” is a major problem for the LUT which degrades the performance.

3 Conclusions

This paper presents an inter-layer prediction scheme for bit-depth scalable coding in the H.264/AVC SVC. The LUT method is used for the down-converting, filtering, and up-converting processes to observably improve the prediction efficiency. Double arithmetic is used instead of integer arithmetic to further improve the coding efficiency. Experimental results demonstrate the advantages of this scheme compared favorably to the recent JVT proposal.

References

[1] Drago F, Myszkowski K, Annen T, et al. Adaptive logarithmic mapping for displaying high contrast scenes. *Computer Graphics Forum*, 2003, **22**: 419-426.

[2] Mantiuk R, Myszkowski K, Seidel H P. A perceptual framework for contrast processing of high dynamic range images. In: Proceedings of APGV 2005: 2nd Symposium on Applied Perception in Graphics and Visualization. A Coruna, Spain, 2005: 87-94.

[3] ITU-T and ISO/IEC JTC1. Joint draft ITU-T Rec. H.264 ISO/IEC 14496-10 / Amd.3 scalable video coding. In: JVT-X201. Geneva, Switzerland, 2007.

[4] Gao Yongying, Wu Yuwen. Bit depth scalability. In: JVT-V061. Marrakech, Morocco, 2007.

[5] Liu Shan, Kim W S, Vetro A. Bit-depth scalable coding for high dynamic range video. In: SPIE Conference on Visual Communications and Image Processing (VCIP). San Jose, USA, 2008.

[6] Liu Shan, Vetro A, Kim W S. Inter-layer prediction for SVC bit-depth scalability. In: JVT-X201. Geneva, Switzerland, 2007.

[7] Segall A, Kerofsky L, Lei S. New results with the tone mapping SEI message. In: JVT-U041. Hangzhou, China, 2006.

[8] Joint Video Team (JVT) of ISO/lEC MPEG & ITU-T VCEG. Joint scalable video model JSVM-12. In: JVT-Y202. Shenzhen, China, 2007.

[9] Winken M, Schwarz H, Marpe D, et al. CE2: SVC bit-depth scalable coding. In: JVT-X057. Geneva, Switzerland, 2007.