BLOCK-BASED COLOR CORRECTION ALGORITHM FOR MULTI-VIEW VIDEO CODING

Boxin Shi, Yangxi Li, Lin Liu, Chao Xu

Key Laboratory of Machine Perception (Ministry of Education), Peking University

Email: {shiboxin, liyangxi, liulin, xuchao}@cis.pku.edu.cn

ABSTRACT

The color variations among different viewpoints in multi-view video sequences may deteriorate the visual quality and coding efficiency. Various color correction methods have been proposed, however, the color appearance and histogram of corrected target frames are not similar enough to the reference frames in details. Focusing on restoring more similar color, a block-based color correction algorithm is proposed. The blocks in reference frames are matched into target frames through spatial prediction, and the colorization scheme is then adopted to expand color as a coarse correction. Finally the mixture with global color transfer result yields the fine correction. The experiment results show this novel method can provide better visual effect in detail and also provide the corrected frames with histograms more similar to reference histograms.

Index Terms—Multi-view Video Coding, Color

1. INTRODUCTION

The emerging multi-view video applications such as free viewpoint television (FTV) [1] and 3DTV [2] provide people with more realistic experience and better interaction while watching videos. The multi-view video sequence is captured using a combination of synchronous cameras from different positions. Although the set of cameras has been adjusted to the same configuration as precisely as possible, it is still difficult to avoid the luminance and chrominance discrepancies among different viewpoints due to the scene illumination, camera calibration and jitter speed. These variations not only deteriorate the visual quality, but also degrade the coding efficiency.

Fecker [3] introduced the histogram matching scheme to solve this problem. A mapping function calculated from the cumulative histograms of the reference frames and target frames is established to correct the color. Chen [4] also used histogram and did YUV correction based on modulation and translation parameters to optimize the compensation parameters. Huo [5] suggested using spatial prediction for best matching, and linear correction factors were multiplied to the target frames. Shao [6] proposed a method considering the image content. The color correction matrix

was used as a mapping function and image segmentation was adopted to relate the corresponding color regions from different viewpoints.

The first three methods operate on the whole picture, so we call them *global correction*. The fourth method does image segmentation and establishes corresponding regions, so we call it *local correction*. The problem with global correction is sometimes the mapping function cannot recover the color details of the whole image or bring some over-correction to the whole image. They may also disturb the histograms of target frames: the histogram matching causes discontinuous segments in corrected histogram, while linear factor in [5] changes the boundary of histogram. And the problem with segmentation-based local correction is that image segmentation is a time-consuming labor which is not necessary in video coding.

This paper proposes a block-based color correction algorithm, which is a novel local algorithm. We focus more on restoring better color similarity using the corresponding blocks “borrowed” directly from reference frames to target frames. Furthermore, spatial prediction is used to match the blocks and this process can avoid the heavy computation from segmentation and also easily to be integrated into a multi-view video coding system. Comparing to existed works, the color corrected frames by the proposed method perform better visual quality especially in detail, more similar histogram, and more matching blocks during spatial prediction.

2. ALGORITHM DESCRIPTION

The entire work-flow for our algorithm is shown as a block diagram in Fig. 1. Our algorithm mainly works in the YUV color space which is often used in video compression.

The first step is extracting luminance from reference and target frames. Histogram equalization is done here to guarantee the same luminance scale. In the second step, macro-block searching is done between reference and target frames for matching parts. We use luminance SAD (Sum of Absolute Difference) to measure correspondence. This process is similar to the *temporal prediction* in subsequent frames, but it is done among different viewpoints, so it is called *spatial prediction*. However, the spatial disparity is much greater than temporal disparity, thus only very small

Fig. 1. Block diagram of the entire algorithm

SAD is allowed to guarantee the most similar blocks called *best matching blocks*. Under this restriction, it is not likely to relate every macro-block to a matching part, so the colorization is necessary in the third step to render a full-colored picture as *coarse correction* result. In the final step, a mixture with global correction is done to fix some errors as *fine correction*. The explanation of the third and final steps will be given in the next two sections.

2.1 Block-based Colorization

The colorization is a concept in computer graphics and it refers to add color to a monochrome picture. Levin [7] proposed a simple but very effective colorization algorithm under the assumption that neighboring pixels in space-time that have similar intensities should have similar color. Their method did not require image segmentation, and provided convincing colored pictures. But color scribbles had to be annotated manually to indicate the areas with desired color.

The correctness of our algorithm is based on a simple intuition: the matching blocks from reference frames indicate the correct color values for the target frames. Given scribbles, the colorization can promise a full-colored picture. We utilize the output in our second step as scribbles in [7]; therefore we can colorize the image with partial color blocks into a full-colored one. We expand color from colored blocks to the uncolored blocks according to their luminance similarity to their neighbors. This expansion is controlled by minimizing the differences of a pixel and its neighbors around in the U and V channels of the YUV color space. Take U channel for example, we minimize the difference cost of a pixel $U(x, y)$ with its weighted average of color around neighboring area N (e.g. a 3×3 window), and m, n is the width and height of the picture:

$$\text{cost}(U) = \sum_{x=1}^m \sum_{y=1}^n \left(U(x, y) - \sum_{(x+\Delta x, y+\Delta y) \in N} w_N U(x + \Delta x, y + \Delta y) \right)^2 \quad (1)$$

The weight factor w_N is calculated from the square difference between different luminance in Y channel, σ_N is the standard deviation of pixel $U(x, y)$ around the neighboring area N:

$$w_N = \exp \left(\frac{-(Y(x, y) - Y(x + \Delta x, y + \Delta y))^2}{2\sigma_N^2} \right) \quad (2)$$

The optimization problem above can be solved using a common method such as least squares. After the colorization for both U and V channels, we obtain a full-colored picture with almost correct color. But it is just a coarse result, because the scribbles are indicated using square blocks which cannot describe the outline of complicated curves in a smooth and accurate way, so automatic colorization may annotate color outside some boundary areas. On the other hand, we do not promise the best matching blocks are able to cover all areas required to be colored, especially for some occluded blocks which are totally different and cannot find corresponding between the two viewpoints. Thus, a fix to this coarse result is necessary in the next step.

2.2 Mixture with Global Color Correction

During the block-based colorization step the correct color gets assigned to the coarse result from the reference frames for most of the pixels, especially in large color-consistent areas. Next, we only need to detect all the other blocks which are not or wrongly colored in the previous step. An empirical threshold T ($T = 15$ in our experiment) is introduced to detect these blocks. Let $C_{U/V}(x, y)$ and $G_{U/V}(x, y)$ denote the pixels of U/V channel in coarse result from previous step and global corrected result in this step respectively. Then the pixel (x_0, y_0) satisfies (3) is considered to be uncolored or wrongly colored:

$$\sqrt{(C_U(x_0, y_0) - G_U(x_0, y_0))^2 + (C_V(x_0, y_0) - G_V(x_0, y_0))^2} \geq T \quad (3)$$

For this pixel, we consider it has too large a departure from the global result and attach it with a value from global correction. Through the evaluation of visual effect to various methods, we choose the scheme of color transfer in [8] as our global strategy. This method is very simple and the corrected color is acceptable. But it has to be used after the colorization as a supplement, because it also has the same problems like other global method mentioned in the introduction. The color transfer is recommended to work in the $\alpha\beta$ color space, which minimizes correlation among channels for many natural scenes [9]. After the color space transformation, the correction procedure is done pixel by pixel by applying the mean value μ and standard deviations σ from the reference frames R to the target frames T,

Table I. Ratio and gain of best matching blocks in spatial prediction

Sequence /Frame	Uncorrected		Histogram matching		Linear Correction		Proposed method	
	ratio	gain	ratio	gain	ratio	gain	ratio	gain
flamenco2/32	9.16%	0%	16.32%	78.17%	14.40%	57.21%	17.07%	86.35%
flamenco2/256	22.15%	0%	21.98%	-0.77%	25.56%	15.40%	26.81%	21.04%
race1/0	53.88%	0%	55.49%	2.99%	50.30%	-6.64%	55.82%	3.60%
race1/48	24.79%	0%	34.01%	37.19%	33.85%	36.55%	35.62%	43.69%

$l_T(x, y)$ means the l value of target frame in $\alpha\beta$ space, the same operation is required to be done in α and β channels.

$$l_T^*(x, y) = \frac{\sigma_R}{\sigma_T} (l_T(x, y) - \mu_T^l) + \mu_R^l \quad (4)$$

Then the global corrected result should transform back to YUV space and the detected pixels according to (3) should be substituted by those values from (4). The processed picture after mixture is the fine correction result. Usually only a small part of pixels are substituted in this process, so the overall visual quality will not be disturbed.

3. EXPERIMENT RESULTS

We use the multi-view video sequences from KDDI [10] to verify our algorithm. The visual qualities as well as histograms of corrected images are depicted to show the improvement in subjective sense. We also calculate the ratio of best matching block as an objective measure. We use 8×8 blocks with a searching range of 16 pixels in spatial prediction. The correction method in [3] (without time-constant mapping function) and [5] are chosen as control group, and we name them as “histogram” method and “linear” method for short respectively in the figures.

Fig. 2 shows the comparison of visual quality. Our goal is to make the color of viewpoint1 (target) similar to viewpoint0 (reference), and the proposed method provides the closest similarity. Fig. 3 illustrates the YUV histograms of the sequences and frames tested in Fig. 2. The proposed method also reveals best similarity in histograms.

In order to give a quantitative analysis of color similarity after correction, we calculate the *ratio of best matching blocks* over total number of blocks after applying three different correction methods (higher ratio corresponds to closer similarity). The ratios calculated from the two frames in Fig. 2 and another two frames with different scenes are listed in Table I, and the gains of three kinds of ratios after correction are also given. We constrain the searching range for matching blocks within two viewpoints and use the same threshold for best matching blocks as in the second step of our algorithm. For example, for every block in three channels, if allowed minimum difference δ ($\delta = 3$ in our experiment) is given, then $SAD = 8 \times 8 \times 3 \times \delta$. It can be seen the proposed method owns the highest ratio comparing to other methods. Although this is not the same condition in the practical multi-view video coding environment, it can obviously reflect the color similarity brought by different color correction methods.

In the special case when only spatial prediction is performed the ratio of best matching increases, but the average PSNR of multi-view video coding is still dominated by temporal prediction, as the results shown in [5]. Our method also faces this limitation. The time-constant mapping function in [3] could improve the result, but this is a topic beyond the discussion of this paper. However, as a pre-processing operation before compression, the potential contribution of color correction may be the visual improvement to viewpoint interpolation and rendering.

4. CONCLUSION

A novel block-based color correction algorithm for multi-view video coding is introduced. It improves the color appearance in detail comparing to global correction method and makes the histogram more similar and continuous. This visual consistency also contributes to spatial prediction.

The most time-consuming part of our method happens during the spatial prediction, which is an unavoidable part in multi-view video coding, so the extra operation for our color correction is less complicated than the segmentation-based local correction. In future works, we will try to incorporate the correction algorithm into the coding pipeline with consideration of temporal consistency in order to improve the average PSNR, and also relate the visual improvement to viewpoint interpolation and rendering.

5. REFERENCES

- [1] M. Tanimoto, “Free viewpoint television—FTV”, in *Proc. Picture Coding Symp. (PCS 2004)*, San Francisco, USA, Dec. 2004.
- [2] W. Matusik, H. Pfister, “3DTV: A Scalable System for Real-time Acquisition, Transmission, and Autostereoscopic Display of Dynamic Scenes”, in *Proc. ACM SIGGRAPH*, Los Angeles, USA, pp. 814-824, Aug. 2004.
- [3] U. Fecker, M. Barkowsky, A. Kaup, “Histogram-Based Prefiltering for Luminance and Chrominance Compensation of Multiview Video”, *IEEE Trans. Circuits & Systems for Video Technology*, vol. 18(9), pp. 1258 – 1267, Sep. 2008.
- [4] Y. Chen, C. Cai, J. Liu, “YUV Correction for Multi-View Video Compression”, in *Proc. ICPR*, Hong Kong, pp. 734 – 737, Aug. 2006.
- [5] J. Huo et al., “Color Compensation for Multi-view Video Coding based on Diversity of Cameras”, *J. Zhejiang Univ. Sci A*, vol. 9(12), pp. 1631– 1637, Sep. 2008.
- [6] F. Shao et al., “A Content-Adaptive Multi-View Video Color Correction Algorithm”, in *Proc. ICASSP*, Honolulu, HI, pp. 969 – 972, Apr. 2007.

[7] A. Levin, D. Lischinski, Y. Weiss, “Colorization using Optimization”, *ACM Trans. Graph.*, vol. 23(3), pp. 689 – 694, Aug. 2004.

[8] E. Reinhard, M. Ashikhmin, B. Gooch, and P. Shirley, “Color transfer between images”, *IEEE Computer Graphics Applications*, vol. 21(5), pp. 34 – 41, 2001.

[9] D. Ruderman, T. Cronin, and C. Chiao, “Statistics of Cone Responses to Natural Images: Implications for Visual Coding,” *J. Optical Soc. of America*, vol. 15(8), pp. 2036 – 2045, 1998.

[10] ISO/IEC JTC1/SC29/WG11, “KDDI Multi-View Video Sequences for MPEG 3DAV Use”, MPEG2004/M10533, Munich, Mar. 2004.

Fig. 2. Visual quality comparison in detail (enlarged parts and thumbnails). Adjust color of viewpoint1 to viewpoint0: (a) flamenco2 frame32: Histogram method cannot wipe off the remaining red in target while linear method does over-correction by adding too much blue; (b) race1 frame 48: the green in proposed method is more similar to reference than others.

Fig. 3. Histogram comparison. (a) flamenco2 frame32; (b) race1 frame 48. The histogram method may cause discontinuous segments in the corrected histogram; the linear method changes the boundary of the histogram; the proposed method can fix these problems and produce more similar histograms to the reference frames (viewpoint0).