SIFT-flow-based color correction for multi-view video

Huanqiang Zeng a,⁎, Kai-Kuang Ma b, Chen Wang b, Canhui Cai a

a School of Information Science and Engineering, Huaqiao University, Xiamen, China
b School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore

A R T I C L E   I N F O

Article history:
Received 31 July 2013
Received in revised form 25 April 2015
Accepted 28 May 2015
Available online 11 June 2015

Keywords:
Multi-view video
Color variation
Color correction
SIFT flow
MVC

A B S T R A C T

During the multi-view video acquisition, color variation across the views tends to be incurred due to different camera positions, orientations, and local lighting conditions. Such color variation will inevitably deteriorate the performance of the follow-up multi-view video processing, such as multi-view video coding (MVC). To address this problem, an effective color correction algorithm, called the SIFT flow-based color correction (SFCC), is proposed in this paper. First, the SIFT-flow technique is used to establish point-to-point correspondences across all the views of the multi-view video. The average color is then computed based on those identified common corresponding points and used as the reference color. By minimizing the energy of the difference yielded between the color of those identified common corresponding points in each view with respect to the reference color, the color correction matrix for each view can be obtained and used to correct its color. Experimental results have shown that the proposed SFCC algorithm is able to effectively eliminate the color variation inherited in multi-view video. By further exploiting the developed SFCC algorithm as a pre-processing for the MVC, extensive simulation results have shown that the coding efficiency of the color-corrected multi-view video can be greatly improved (on average, 0.85 dB, 1.27 dB and 1.63 dB gain for Y, U, and V components, respectively), compared with that of the original multi-view video without color correction.

© 2015 Elsevier B.V. All rights reserved.

1. Introduction

With the rapid development of camera, display and network communication techniques, multi-view video systems, such as three-dimensional TV (3DTV) and free viewpoint TV (FTV), have been emerging [1,2], as these services provide customers with depth perception and freedom of choosing the viewpoint. Due to better interactivity and reality, the multi-view video systems are expected to be widely exploited in various domains, such as home entertainment, education, medical field, to name a few.

Unlike the single-view video, the multi-view video is composed of more than one view, and these views are acquired by using multiple cameras distributed in different viewpoints to synchronously capture the same scene. Consequently, noticeable color variation is often incurred across the views due to different camera positions, orientations, characteristics, and local lighting conditions during the multi-view video acquisition. In order to efficiently compress the huge amount of multi-view video data, the Joint Video Team (JVT) standardization body has standardized multi-view video coding (MVC) as an amendment of H.264/AVC—Annex H [3]. The MVC not only exploits the spatial and temporal correlation within a single view (i.e., intra-view) but also employs the correlation between the neighboring views (i.e., inter-view) for further improving the coding efficiency. However, more color variation across...
the views means that less inter-view correlation will be presented among them. Obviously, this will reduce the coding efficiency of the MVC. For that, an efficient color correction algorithm for mitigating color variation inherited among various views of the multi-view video would be highly desirable and can be used as a pre-processing stage for facilitating the follow-up multi-view video processing tasks.

Multiple color correction methods for multi-view video can be found in the literature. Based on the assumption that the DC component of each macroblock is affected by the local illumination change, Hur et al. [4] suggested a macroblock-based adaptive local illumination change compensation method. Fecker et al. [5] utilized a histogram matching approach for conducting color variation reduction based on the fact that the histogram of the target view should be similar to that of its reference view. For that, the cumulative histograms of the target view and its reference view are computed to establish a color mapping function. Shi et al. [6] presented a two-stage color correction method. In the first stage, a colorization process is exploited to expand the color of each 8 × 8 block in the target view relative to that of the corresponding block in the reference view. A coarse-to-fine scheme is then employed for those blocks that still present large color variation after the previous stage. By using the block-based disparity estimation based on a normalized cross-correlation criterion, Doutre et al. [7] identified the common corresponding points shared by all the views in order to compute the average color. For each view, a third-order polynomial color correction function is then established in reference to the computed average color so as to correct its color. Chen et al. [8] presented a histogram-offset-based color correction method. In this method, the histograms of the reference view and the target view are first computed within their maximally matched regions, which are identified by conducting disparity estimation in the rank-transformed domain. A histogram offset is then calculated to correct the target view by using an iterative thresholding approach. Shao et al. [9] jointly exploited color correction and chrominance reconstruction as the pre- and post-processing stages of MVC to improve its coding efficiency.

In addition, since the inception of scale invariant feature transformation (SIFT) [10], some SIFT-based color correction methods for multi-view video are developed due to its effectiveness on finding correspondences. Among them, Yamamoto et al. [11] proposed a method to correct the luminance and the chrominance of other view frames for inter-view prediction and view-interpolation prediction based on the lookup tables of each RGB color channel. Yamamoto et al. [12] exploited the SIFT to detect the correspondences, calculated lookup tables with an energy-minimization approach and then corrected the multi-view video with these tables. Tehrani et al. [13] randomly chose a view and the next view as the reference view and the target view, respectively, for performing iterative color correction. In this method, the color correction transformation is calculated based on the corresponding points obtained by using modified SIFT with outlier suppression. The iteration will be stopped, when the average change between the reference and target views is smaller than a threshold. Lu et al. [14] presented a color correction method by jointly using the SIFT and general regression neural network. Jung et al. [15] proposed a color correction method based on the camera characteristic curve. This camera characteristic curve is first modeled by using nonlinear regression (with outlier removal in advance) based on the correspondences between the reference and target views and is then utilized to generate the lookup tables to correct the color distribution of the target views. Fezza et al. [16] exploited the combination of the SIFT and RANSAC methods to effectively define the common regions across views, and then developed an improved histogram matching using only common regions and temporal sliding window to achieve a robust color correction performance.

In this paper, an effective color correction algorithm for multi-view video, called the SIFT-flow-based color correction (SFCC), is proposed. In our approach, the SIFT flow [17] technique (where SIFT stands for scale invariant feature transform [10]) is firstly exploited to identify all common corresponding points across all the views of the multi-view video. The average color is then computed based on these identified common corresponding points and used as the reference color. The color of each view is then corrected by multiplying a color correction matrix, which is generated by minimizing the energy of the difference yielded between the color of those identified common corresponding points in each view with respect to the reference color using the least mean-square approach. Experimental results have demonstrated the accuracy and effectiveness of the proposed SFCC algorithm on the mitigation of color variation inherited in the multi-view video, compared with the state-of-the-art color correction methods.

The rest of this paper is organized as follows. The proposed color correction algorithm, SFCC, is presented in Section 2 in detail. Extensive simulation results are documented and discussed in Section 3. Finally, conclusion is drawn in Section 4.

2. Proposed sift-flow-based color correction (SFCC)

The proposed color correction algorithm, SFCC, consists of two stages as described in the following two sub-sections, respectively. In the first stage, the SIFT flow technique [17] is exploited to identify all the common corresponding points across all the views of the multi-view video. In the second stage, the average color is computed based on those detected common corresponding points across all the views. By using this computed average color as the reference color, the color correction matrix for each view is then established and employed to correct the color of each view, respectively. It should be pointed out that the proposed SFCC algorithm is performed picture by picture and view by view for the entire multi-view video sequence. In other words, the pictures from different viewpoints at the same time instant are individually corrected.

2.1. Finding common corresponding points

Intuitively, it is fairly essential to identify accurate corresponding points as many as possible for color correction. In our view, approaches to find the corresponding
points between two views can be classified into two broad categories: the block-based approach and the feature-based approach. In the first category, both the current and reference views (or pictures) are divided into smaller blocks such that each block in the current picture is subject to be matched at all locations within the pre-imposed search window of the reference picture according to a criterion (e.g., sum of absolute differences, SAD). Although the block-based approach can generate dense corresponding points, however, this approach is sensitive to illumination change since such illumination change quite large and these resulted corresponding points tend to be more accurate [19]. Therefore, the feature-based approach is further investigated here.

To identify accurate corresponding points between two views as many as possible, the SIFT flow technique [17] is exploited in this paper. Inspired by the optical flow [20,21], which can produce dense corresponding points, SIFT flow follows the same principles on the computing of the optical flow, except that the matching process is conducted based on the SIFT descriptors extracted on both current and reference pictures, rather than using the pixel intensity as performed in the optical flow. Therefore, it combines the advantages of both SIFT features and optical flow on providing accurate and dense corresponding points. Another unique and highly desirable merit of exploiting SIFT flow is that it is especially suitable for those scenarios that undergo a small viewpoint variation. This is in line with the fundamental nature of the content across the views of multi-view video.

In the SIFT flow, a SIFT descriptor is first established at each pixel position to characterize the local image structures for both the current and reference pictures. Similar to the optical flow [20,21], the SIFT descriptors from the corresponding frames of two views are then matched to obtain the point-to-point correspondences and their flow vectors. This is achieved by minimizing the following energy function for SIFT flow, $E(\mathbf{w})$ [17]:

$$E(\mathbf{w}) = \sum_p \min \left( \| s_1(\mathbf{p}) - s_2(\mathbf{p} + \mathbf{w}(\mathbf{p})) \|, t \right)$$

$$+ \sum_p \eta |u(\mathbf{p})| + |v(\mathbf{p})| \right)$$

$$+ \sum_{(\mathbf{p}, \mathbf{q}) \in z} \left[ \min(\alpha \cdot |u(\mathbf{p}) - u(\mathbf{q})|, d) + \min(\alpha \cdot |v(\mathbf{p}) - v(\mathbf{q})|, d) \right]$$

where $s_1(\cdot)$ and $s_2(\cdot)$ denote the SIFT descriptors extracted at each pixel position on the current and reference pictures, respectively, $\mathbf{p} = (x_p, y_p)$ and $\mathbf{q} = (x_q, y_q)$ are the spatial coordinates, $\mathbf{w}(\mathbf{p}) = (u(\mathbf{p}), v(\mathbf{p}))$ is the flow vector at $\mathbf{p}$, $\epsilon$ represents the four-neighbors of $\mathbf{p}$, $\eta$ and $\alpha$ are the regularization parameters that provide tradeoff among (1)–(3) and are empirically determined, $t$ and $d$ are thresholds, which are utilized to eliminate matching outliers and flow discontinuities, respectively. Note that the default parameter setting of SIFT flow [17] is used in this work, that is, $\epsilon = 2 \times 255$, $\eta = 0.005 \times 255$, $d = 40 \times 255$, and $t$ is disabled.

Furthermore, it can be seen that the energy function $E(\mathbf{w})$ is formulated as the similar form of the optical flow and consists of three terms: the data term, the small displacement term and the smoothness term. First, the data term shown in (1) is used to reflect the similarity of the local image structures between the current and reference pictures. This can be accurately measured in terms of the difference incurred between $s_1(\cdot)$ and $s_2(\cdot)$. Therefore, to find more accurate corresponding points, the data term is preferred to be as small as possible. Second, although the flow vector is allowed to be as large as the picture itself, the small displacement term shown in (2) is exploited to constrain the flow vector to be as small as possible, which encourages the smoothness of the flow field. Finally, the smoothness term shown in (3) is utilized to penalize any unsmoothness that might be incurred in the flow field. This can be simply achieved by minimizing the difference of the flow vectors of the adjacent pixels. Since all the above-mentioned three terms are favored to be as small as possible, the energy function $E(\mathbf{w})$ is subject to be minimized.

As mentioned above, SIFT flow can be directly applied to obtain dense corresponding points between two views. In this light, we further explore SIFT flow for the scenario of multi-view video to identify common corresponding points across all views as below and an illustration is further presented in Fig. 1. For the ease of explanation, let $N$ be the total number of views and $V_i$ represent the $i$th view of the multi-view video, where $i$ ranges from 1 to $N$.

1. For each two consecutive views of the given multi-view video $V_i$ and $V_{i+1}$ (e.g., View 1 and View 2 in Fig. 1), SIFT flow is performed between them and the resulted flow vector of view $V_i$ is denoted as $\mathbf{w}_i(\mathbf{p}) = (u_i(\mathbf{p}), v_i(\mathbf{p}))$, for $i = 1, 2, \ldots, N - 1$, respectively.

2. By using the flow vector $\mathbf{w}_i(\mathbf{p}) = (u_i(\mathbf{p}), v_i(\mathbf{p}))$ obtained in the previous step, for each point in the first view $V_1$ (denoted as $\mathbf{p}_1 = (x_1, y_1)$), its corresponding point in all the other views $V_i$ (denoted as $\mathbf{p}_i = (x_i, y_i)$) can be derived as follows: for $i = 2, 3, \ldots, N$, respectively:

$$x_i = x_1 + \sum_{k=1}^{i-1} u_k(x_k, y_k)$$

$$y_i = y_1 + \sum_{k=1}^{i-1} v_k(x_k, y_k)$$

subject to

$$1 \leq x_i \leq H$$

$$1 \leq y_i \leq W$$

where $H \times W$ is the frame resolution of the multi-view video.
If point \( \mathbf{p}_i \) can find its corresponding points \( \mathbf{p} \) in all the other views \( V_i \) based on (4), these corresponding points (i.e., \( \mathbf{p}_i \), for \( i = 1, 2, \ldots, N \)) will be selected as one of the common corresponding points across all the views (e.g., the yellow circle in Fig. 1). On the contrary, if any point \( \mathbf{p}_i \) falls out of the image range, it means that \( \mathbf{p}_i \) fails to find its corresponding point in view \( V_i \) and thus these corresponding points (i.e., \( \mathbf{p} \), for \( i = 1, 2, \ldots, N \)) will not be considered as a common corresponding point (e.g., the red square in Fig. 1).

2.2. Establishing color correction matrix

Since most cameras deliver RGB color signal directly, the proposed SFCC algorithm is conducted in the RGB color space. In this work, the color of each pixel of the uncorrected picture in view \( V_i \), for \( i = 1, 2, \ldots, N \), respectively, can be corrected via a linear transformation; that is,

\[
\begin{bmatrix}
\hat{R}_i \\
\hat{G}_i \\
\hat{B}_i
\end{bmatrix} = \mathbf{A}_i
\begin{bmatrix}
R_i \\
G_i \\
B_i
\end{bmatrix}
\]

(5)

where \( R_i, G_i, \) and \( B_i \) are the original R-, G-, and B-component values of each pixel of the current picture in view \( V_i \) to be corrected, respectively; \( \hat{R}_i, \hat{G}_i, \) and \( \hat{B}_i \) denote the corrected counterpart components of the same pixel, respectively; the matrix \( \mathbf{A}_i \) is a \( 3 \times 3 \) color correction matrix of view \( V_i \) with real-valued entries, for \( i = 1, 2, \ldots, N \), respectively.

Now the question boils down to how to determine the color correction matrix \( \mathbf{A}_i \) for view \( V_i \). For this, the proposed SFCC algorithm makes full use of the detected common corresponding points across all views. Firstly, the average color is calculated as the reference color by simply averaging each color component (i.e., \( R, G, B \)) of the detected common corresponding points across all views individually. The color correction matrix \( \mathbf{A}_i \) is then computed by minimizing the energy of difference yielded between the color of the detected common corresponding points in view \( V_i \) and the average color.

Let matrix \( \mathbf{C}_i = [R_i, G_i, B_i]^T \) and matrix \( \hat{\mathbf{C}}_i = [\hat{R}_i, \hat{G}_i, \hat{B}_i]^T \) represent the original and corrected color of the common corresponding points located in view \( V_i \), and let matrix \( \mathbf{C} = [R, G, B]^T \) denote the average color of \( \mathbf{C}_i \). The \( R, G, \) and \( B \) are the vectors that contain the original R-, G-, and B-component values of the detected common corresponding points located in view \( V_i \), respectively. The \( \bar{R}, \bar{G}, \) and \( \bar{B} \) are the vectors that contain the average R-, G-, and B-component values of the detected common corresponding points identified from all views, respectively. Therefore, the average color \( \mathbf{C} \) can be calculated as

\[
\mathbf{C} = \begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix}
\frac{1}{N} \sum_{i=1}^{N} R_i \\
\frac{1}{N} \sum_{i=1}^{N} G_i \\
\frac{1}{N} \sum_{i=1}^{N} B_i
\end{bmatrix}
\]

(6)

The goal of color correction is to correct the color of all views so that the color of each view becomes similar to the reference color so as to eliminate color variation. In other words, by using the average color as the reference color, the difference yielded between the corrected color of the detected common corresponding points in each view \( \hat{\mathbf{C}}_i \) and this average color \( \mathbf{C} \) should be as minimum as possible. Hence, for view \( V_i \), the corresponding difference (i.e., matrix \( \mathbf{d}_i \)) can be defined as

\[
\mathbf{d}_i = \mathbf{C} - \hat{\mathbf{C}}_i
\]

(7)
According to (5), $d_i$ can be further expressed as

$$d_i = \bar{C} - A_i C_i$$  \hspace{1cm} (8)

Hence, the color correction matrix $A_i$ can be determined by exploiting the least mean square, which is to minimize the energy of difference, $d_i^T d_i$, as

$$A_i = \arg \min_{A_i} (d_i^T d_i)$$  \hspace{1cm} (9)

Consequently, the relative analytic solution towards (9) can be derived as

$$A_i = \mathbb{C} C_i (C_i C_i^T)^{-1}$$  \hspace{1cm} (10)

With the obtained color correction matrix $A_i$, the color of view $V_i$ can now be corrected based on (5), for $i = 1, 2, \ldots, N$, respectively.

### 3. Experimental results and discussion

In this work, the proposed color correction algorithm, SFCC, has been experimented using multiple multi-view video sequences as listed in Table 1 [22], and the corresponding results are described in the following three subsections. Since these multi-view video sequences have been stored in the YUV format while the proposed algorithm requires to be conducted in the RGB color space, they need to be converted to the RGB color space firstly.

#### 3.1. Visual quality evaluations

First, the visual quality comparison on multiple multi-view video sequences before and after color correction are shown in Figs. 2-5. To have a better illustration, the zoom-in areas indicated by the yellow bounding box in two multi-view video sequences, “Race1” and “Flamenco2,” (i.e., Figs. 2 and 3) are enlarged and presented in Figs. 6 and 7 as examples, respectively.

One can see that each original (i.e., uncorrected) multi-view video sequence has noticeable color variation across the views. For example, there is a significant change in brightness from view $V_1$ to view $V_2$ in “Race1” and so does the brightness of view $V_2$ of “Flamenco2” when compared with that of the other views. After applying the proposed

<table>
<thead>
<tr>
<th>Sequences</th>
<th>From</th>
<th>Resolution</th>
<th>Views</th>
<th>Frames</th>
</tr>
</thead>
<tbody>
<tr>
<td>Race1</td>
<td>KDDI</td>
<td>640 x 480</td>
<td>8</td>
<td>250</td>
</tr>
<tr>
<td>Flamenco2</td>
<td>KDDI</td>
<td>640 x 480</td>
<td>5</td>
<td>250</td>
</tr>
<tr>
<td>Crowd</td>
<td>KDDI</td>
<td>640 x 480</td>
<td>5</td>
<td>250</td>
</tr>
<tr>
<td>Rena</td>
<td>Tanimoto Lab</td>
<td>640 x 480</td>
<td>8</td>
<td>250</td>
</tr>
</tbody>
</table>

Fig. 2. Color correction result of multi-view video sequence “Race1” (the 25th frame): (a) original pictures; (b) corrected pictures by the proposed SFCC algorithm.

Fig. 3. Color correction result of multi-view video sequence “Flamenco2” (the 1st frame): (a) original pictures; (b) corrected pictures by the proposed SFCC algorithm.
SFCC algorithm, it can be observed that the color across all the views of the corrected multi-view video sequences looks much more consistent (i.e., with much less variations), compared with that of the uncorrected ones.

To further evaluate the color correction performance, the uncorrected multi-view video and the corrected one resulted from different color correction methods are evaluated by utilizing four visual quality assessment metrics—structural similarity metric, denoted as SSIM in [23], feature similarity metrics, denoted as FSIM and FSIMc in [24], and motion-based video integrity evaluation metric, denoted as MOVIE in [25], since these metrics are correlated well with the human visual perception and provide good subjective evaluation. Note that SSIM, FSIM and MOVIE employ only the luminance

Fig. 4. Color correction result of multi-view video sequence “Rena” (the 89th frame): (a) original pictures; (b) corrected pictures by the proposed SFCC algorithm.

Fig. 5. Color correction result of multi-view video sequence “Crowd” (the 128th frame): (a) original pictures; (b) corrected pictures by the proposed SFCC algorithm.

Fig. 6. An enlarged portion of the yellow bounding box region indicated in Fig. 2: (a) original pictures; (b) corrected pictures by the proposed SFCC algorithm.
information while FSIMC exploits both luminance and chrominance information. Besides, the higher SSIM, FSIM and FSIMC scores while the lower MOVIE score indicates the better visual quality. In our experiments, SSIM, FSIM, FSIMC and MOVIE are individually computed across two consecutive views (i.e., Vi and Vi+1) of the given multi-view video sequentially, where View Vi is used as the reference view of View Vi+1. Then, the computed results are averaged as the subjective result, which can effectively reflect the color consistency across the views of the multi-view video before and after applying various color correction methods.

Table 2 documents the SSIM, FSIM, FSIMC and MOVIE values of the uncorrected multi-view video sequences (denoted as “Uncorrected”) and the corrected ones by the methods recently proposed by Fecker et al. [5] (denoted as HM), Doutre et al. [7] (denoted as BDEPF), Fezza et al. [16] (denoted as FBCC), and the proposed SFCC algorithm. It can be seen that the proposed SFCC algorithm constantly achieves the highest SSIM, FSIM, FSIMC and the lowest MOVIE average scores, showing that the proposed color correction algorithm is able to yield better visual quality than the Uncorrected, HM, BDEPF and FBCC, respectively.

3.2. Coding efficiency evaluations

Color correction is normally considered as an image pre-processing scheme to facilitate the main task of intended image application. In this work, we shall further demonstrate that the color correction could be appreciably beneficial to multi-view video coding (MVC), on the aspect of coding efficiency.

To show the effect on coding efficiency contributed by color correction, experiments have been conducted both on the original (before correction) and on the corrected multi-view video sequences based on the MVC reference software—joint multi-view video coding (JMVC 8.5) [26]. In our experiments, various color correction methods are applied to the original multi-view video sequence to

<table>
<thead>
<tr>
<th>Sequences</th>
<th>Method</th>
<th>SSIM</th>
<th>FSIM</th>
<th>FSIMC</th>
<th>MOVIE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Race1</td>
<td>(A)</td>
<td>0.4609</td>
<td>0.6740</td>
<td>0.6634</td>
<td>0.0125</td>
</tr>
<tr>
<td></td>
<td>(B)</td>
<td>0.4637</td>
<td>0.6755</td>
<td>0.6658</td>
<td>0.0123</td>
</tr>
<tr>
<td></td>
<td>(C)</td>
<td>0.4670</td>
<td>0.6765</td>
<td>0.6683</td>
<td>0.0120</td>
</tr>
<tr>
<td></td>
<td>(D)</td>
<td>0.4745</td>
<td>0.6802</td>
<td>0.6726</td>
<td>0.0115</td>
</tr>
<tr>
<td></td>
<td>(E)</td>
<td>0.4938</td>
<td>0.6860</td>
<td>0.6786</td>
<td>0.0110</td>
</tr>
<tr>
<td>Flamenco2</td>
<td>(A)</td>
<td>0.4361</td>
<td>0.6517</td>
<td>0.6362</td>
<td>0.0151</td>
</tr>
<tr>
<td></td>
<td>(B)</td>
<td>0.4397</td>
<td>0.6554</td>
<td>0.6428</td>
<td>0.0146</td>
</tr>
<tr>
<td></td>
<td>(C)</td>
<td>0.4414</td>
<td>0.6538</td>
<td>0.6403</td>
<td>0.0145</td>
</tr>
<tr>
<td></td>
<td>(D)</td>
<td>0.4426</td>
<td>0.6562</td>
<td>0.6431</td>
<td>0.0139</td>
</tr>
<tr>
<td></td>
<td>(E)</td>
<td>0.4429</td>
<td>0.6577</td>
<td>0.6439</td>
<td>0.0137</td>
</tr>
<tr>
<td>Rena</td>
<td>(A)</td>
<td>0.8733</td>
<td>0.9013</td>
<td>0.8926</td>
<td>0.0027</td>
</tr>
<tr>
<td></td>
<td>(B)</td>
<td>0.8747</td>
<td>0.9025</td>
<td>0.8953</td>
<td>0.0024</td>
</tr>
<tr>
<td></td>
<td>(C)</td>
<td>0.8758</td>
<td>0.9036</td>
<td>0.8960</td>
<td>0.0022</td>
</tr>
<tr>
<td></td>
<td>(D)</td>
<td>0.8750</td>
<td>0.9028</td>
<td>0.8952</td>
<td>0.0022</td>
</tr>
<tr>
<td></td>
<td>(E)</td>
<td>0.8782</td>
<td>0.9068</td>
<td>0.8998</td>
<td>0.0019</td>
</tr>
<tr>
<td>Crowd</td>
<td>(A)</td>
<td>0.1455</td>
<td>0.5732</td>
<td>0.5605</td>
<td>0.0208</td>
</tr>
<tr>
<td></td>
<td>(B)</td>
<td>0.1478</td>
<td>0.5769</td>
<td>0.5648</td>
<td>0.0191</td>
</tr>
<tr>
<td></td>
<td>(C)</td>
<td>0.1579</td>
<td>0.5816</td>
<td>0.5729</td>
<td>0.0182</td>
</tr>
<tr>
<td></td>
<td>(D)</td>
<td>0.1658</td>
<td>0.5842</td>
<td>0.5756</td>
<td>0.0176</td>
</tr>
<tr>
<td></td>
<td>(E)</td>
<td>0.1766</td>
<td>0.5891</td>
<td>0.5804</td>
<td>0.0169</td>
</tr>
<tr>
<td>Average</td>
<td>(A)</td>
<td>0.4789</td>
<td>0.7001</td>
<td>0.6882</td>
<td>0.0128</td>
</tr>
<tr>
<td></td>
<td>(B)</td>
<td>0.4815</td>
<td>0.7026</td>
<td>0.6922</td>
<td>0.0121</td>
</tr>
<tr>
<td></td>
<td>(C)</td>
<td>0.4855</td>
<td>0.7039</td>
<td>0.6944</td>
<td>0.0117</td>
</tr>
<tr>
<td></td>
<td>(D)</td>
<td>0.4895</td>
<td>0.7059</td>
<td>0.6966</td>
<td>0.0113</td>
</tr>
<tr>
<td></td>
<td>(E)</td>
<td>0.4979</td>
<td>0.7096</td>
<td>0.7007</td>
<td>0.0109</td>
</tr>
</tbody>
</table>

Fig. 7. An enlarged portion of the yellow bounding box region indicated in Fig. 3: (a) original pictures; (b) corrected pictures by the proposed SFCC algorithm.
reduce the color variation among the views as the pre-processing stage, followed by conducting MVC. The test conditions are set as follows:

1. Each test sequence as listed in Table 1 is encoded using the hierarchical B picture prediction structure [27] under a group of picture length = 16.
2. The quantization parameter QP is set at 24, 28, 32, and 36, respectively.
3. The rate distortion optimization (RDO) is enabled.
4. The CABAC entropy coding is used.
5. The search range of motion estimation and disparity estimation is ± 64.

In addition to directly compressing those uncorrected multi-view video sequences (denoted as “Uncorrected”), the coding result of the proposed color correction algorithm, SFCC, is compared with that of methods proposed by HM [5], BDEPF [7] and FBCC [16]. Moreover, Table 3 further summarizes the average PSNR gains of each color correction method are respectively made with respect to the result of directly compressing the uncorrected multi-view video sequence, individually. Note that this average PSNR gains of each color correction method are respectively made with respect to the result of the Uncorrected and measured by the commonly-used BDPSNR as suggested in [28].

From the experimental results as shown in Fig. 8 and Table 3, it can be seen that the proposed SFCC algorithm constantly achieves 0.85 dB PSNR gain averaged over four test sequences in the Y component, compared with that of directly compressing the uncorrected multi-view video sequence. It is important to observe that higher PSNR gain is yielded on chrominance components U (1.27 dB on average) and V (1.63 dB on average) by using the proposed SFCC algorithm. This reflects the effectiveness and potential of the proposed color correction approach. Experimental results clearly indicate that the proposed SFCC algorithm obtains the highest PSNR improvement on average in Y, U, and V components respectively, showing that the proposed color correction algorithm consistently outperforms the HM [5], BDEPF [7] and FBCC [16].

### 3.3. Computational complexity

To evaluate the computational complexity, extensive simulation experiments have been conducted based on a set of multi-view video sequences as listed in Table 1 to obtain the average running time. The PC used for conducting these experiments is made up of 2.66 GHz Intel Core2 processor and 4 GB memory. The proposed color correction algorithm, SFCC, is implemented in the Matlab.

Experimental results have shown that the proposed SFCC algorithm takes 18 s per frame on average for the multi-view video with a frame resolution of 640 × 480. Although the current computational complexity does not allow for a real-time implementation for the proposed algorithm, it can be performed off-line, since it is meant to be utilized as a pre-processing stage to facilitate the follow-up multi-view video processing as mentioned earlier. Moreover, it should be pointed out that the computational complexity of the proposed algorithm can be further reduced, say, through an implementation by using the C language.

### 4. Conclusion

In this paper, a novel color correction algorithm for multi-view video, called the SIFT-flow-based color correction (SFCC), is proposed. In our approach, a set of common corresponding points across all the views in the multi-view video under correction are first identified by utilizing the SIFT flow technique. Based on the detected common corresponding points, the average color is calculated and used as the reference color so that the color correction matrix for each view can be thus obtained by minimizing the energy of difference between the color of the detected common corresponding points in each view and the reference color, respectively. With the obtained color correction matrix, the color correction is then performed for each view. Experimental results have demonstrated...

### Table 3

Average PSNR gains of Y, U, V components of four methods: (A) HM [5], (B) BDEPF [7], (C) FBCC [16], and (D) our proposed SIFT-flow-based color correction (SFCC) algorithm. All the incremental differences are resulted from comparing each method with respect to the result of directly compressing the uncorrected multi-view video sequences, individually.

<table>
<thead>
<tr>
<th>Sequences</th>
<th>Method</th>
<th>Y (dB)</th>
<th>U (dB)</th>
<th>V (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Race1</td>
<td>(A)</td>
<td>+0.43</td>
<td>+0.58</td>
<td>+0.70</td>
</tr>
<tr>
<td></td>
<td>(B)</td>
<td>+0.76</td>
<td>+0.92</td>
<td>+1.04</td>
</tr>
<tr>
<td></td>
<td>(C)</td>
<td>+0.89</td>
<td>+1.22</td>
<td>+1.36</td>
</tr>
<tr>
<td></td>
<td>(D)</td>
<td>+1.67</td>
<td>+1.64</td>
<td>+1.88</td>
</tr>
<tr>
<td>Flamenco2</td>
<td>(A)</td>
<td>+0.19</td>
<td>+0.59</td>
<td>+0.39</td>
</tr>
<tr>
<td></td>
<td>(B)</td>
<td>+0.39</td>
<td>+0.63</td>
<td>+0.64</td>
</tr>
<tr>
<td></td>
<td>(C)</td>
<td>+0.59</td>
<td>+0.82</td>
<td>+0.69</td>
</tr>
<tr>
<td></td>
<td>(D)</td>
<td>+0.57</td>
<td>+0.90</td>
<td>+0.75</td>
</tr>
<tr>
<td>Rena</td>
<td>(A)</td>
<td>+0.48</td>
<td>+0.91</td>
<td>+0.63</td>
</tr>
<tr>
<td></td>
<td>(B)</td>
<td>+0.58</td>
<td>+1.20</td>
<td>+1.45</td>
</tr>
<tr>
<td></td>
<td>(C)</td>
<td>+0.52</td>
<td>+0.93</td>
<td>+1.25</td>
</tr>
<tr>
<td></td>
<td>(D)</td>
<td>+0.68</td>
<td>+1.13</td>
<td>+1.36</td>
</tr>
<tr>
<td>Crowd</td>
<td>(A)</td>
<td>+0.20</td>
<td>+0.56</td>
<td>+0.43</td>
</tr>
<tr>
<td></td>
<td>(B)</td>
<td>+0.27</td>
<td>+0.83</td>
<td>+1.14</td>
</tr>
<tr>
<td></td>
<td>(C)</td>
<td>+0.36</td>
<td>+1.02</td>
<td>+1.39</td>
</tr>
<tr>
<td></td>
<td>(D)</td>
<td>+0.47</td>
<td>+1.41</td>
<td>+2.52</td>
</tr>
<tr>
<td>Average</td>
<td>(A)</td>
<td>+0.33</td>
<td>+0.66</td>
<td>+0.54</td>
</tr>
<tr>
<td></td>
<td>(B)</td>
<td>+0.50</td>
<td>+0.90</td>
<td>+1.07</td>
</tr>
<tr>
<td></td>
<td>(C)</td>
<td>+0.59</td>
<td>+1.00</td>
<td>+1.17</td>
</tr>
<tr>
<td></td>
<td>(D)</td>
<td>+0.85</td>
<td>+1.27</td>
<td>+1.63</td>
</tr>
</tbody>
</table>
that the proposed color correction algorithm, SFCC, can effectively reduce the color variation inherited in the multi-view video. By further exploiting the proposed SFCC algorithm as a pre-processing for the MVC, the coding efficiency of the corrected multi-view video has been appreciably increased (on average, 0.85 dB, 1.27 dB and 1.63 dB increments in PSNR of Y, U and V components, respectively), compared with that of the uncorrected one.
Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under the Grants 61372107 and 61401167, and in part by the High-Level Talent Project Foundation of Huaqiao University under the Grants 14BS201 and 14BS204.

References