Colorimetric Correction for Stereoscopic Camera Arrays

Clyde Mouffranc and Vincent Nozick

Gaspard Monge Institute, UMR 8049
Paris-Est Marne-la-Vallee University, France

Abstract. Colorimetric correction is a necessary task to generate comfortable stereoscopic images. This correction is usually performed with a 3D lookup table that can correct images in real-time and can deal with the non-independence of the colour channels. In this paper, we present a method to compute such 3D lookup table with a non-linear process that minimizes the colorimetric properties of the images. This lookup table is represented by a polynomial basis to reduce the number of required parameters. We also describe some optimizations to speedup the processing time.

1 Introduction

In recent years, stereoscopic technologies have been subject to an impressive growth and became incontrovertible in the movie maker industry. More recently, this technology has advanced from stereoscopic to autostereoscopic displays, involving more than two views and hence more than two cameras. The use of these multiview devices emphasizes technical issues in term of video stream synchronization, camera calibration, geometrical issues or colorimetric correction. This paper deals with this last problem, i.e. how to represent each object of the scene with a coherent colour in every view. Indeed, colorimetric inconsistencies in stereoscopic images may cause some perception troubles, as described in [1] as well as issues for multi-view video coding [2] or video-based rendering [3]. The goal of this paper is to get a uniform colour response among the camera and not to perform a colorimetric calibration [4] to get an absolute colour accuracy.

Most of the colorimetric inconsistencies mainly come from the camera sensors. Even if the cameras are the same model and come from the same factory, the sensor response is often quite different. Thus, selecting the same settings for each camera, i.e. gain, brightness or shutter speed may not solve the problem. Moreover, the camera response for two identical cameras may differ according to their respective position on the scene, where the illumination is not perceived the same or where the camera temperature is different. These colorimetric issues are clearly apparent on low cost cameras but are also visible with professional grade devices.

The usual requirement for a colorimetric correction technique is to be fast and accurate. Moreover, the process should be efficient on high resolution images

J.-I. Park and J. Kim (Eds.): ACCV 2012 Workshops, Part I, LNCS 7728, pp. 206–217, 2013.

for the movie industry as well as for daily applications such as teleconference running on consumer grade hardware. Most of the professionals claim that using separated 1D-LUT for each channel are not recommended since it is known that the colour channels are not independent. The standard approach for colorimetric corrections is to use 3D LookUp Tables (LUT) implemented on the GPU. Indeed, the GPU implementation is straightforward and the performances are excellent. Finally, for images with 256 colour levels per channel, a full size 3D-LUT would have a 256^3 voxel resolution. In practice, the 3D-LUT may have a much lower resolutions since the missing data are linearly interpolated by the GPU. As an example, a 32^3 3D-LUT is preforment enough and requires few memory (e.g. less memory than a RGB 200×200 image). The goal of this paper is to propose a 3D-LUT computation to perform a colorimetric correction between multiple cameras of a camera array.

2 Related Work

The problem of transferring the colorimetric properties of a source image to a target image, namely *colour transfer*, has been the starting point of numerous methods dealing more specifically with multiple view colorimetric correction. A survey of the related works for these two approaches is presented in the following parts.

2.1 Colour Transfer between Two Images

Reinhard et al. [5] present a method that matches the colour mean and variance of the target image to the source image. This operation is performed on the $l\alpha\beta$-colour space where the colour channels are not correlated. However this method is limited to linear transforms. Papadakis et al. [6] describe a variational formulation of the problem using cumulated histograms under colour conservation constraints, but provides 1-D transformations that is not suitable for our purpose.

Morovic and Sun [7] present a method to match the 3D colour histogram of the two images. Neumann and Neumann [8] have te same approach but also apply a smoothing and a contrast constraint to limit unexpected high gradients artifacts. Finally, Pitié et al. [9] matches the probability density function between the two images using a N-dimensional transfer function. These methodes are specially designed to perform colour transfer from images with very different colorimetric properties.

Finally, Abadpour and Kasaei [10] use a principal component analysis (PCA) to generate a new colour space where the channels are decorrelated. In [11], they use the PCA to compute a colour space from some specific image regions selected manually. PCA-based approaches will perform well on a static images, but can fail in video sequences where the variation of the colours may not match the initial colour space.

2.2 Colour Correction and Camera Array

Camera arrays dedicated to stereoscopic rendering are subject to several geometrical constraints (i.e. the cameras should be correctly aligned [12]) and hence the acquired images represent approximatively the same scene, with similar colorimetric properties.

Yamamoto et al. [13] extract SIFT correspondences [14] in order to handle scene occlusions and perform a multiview colorimetric correction. Yamamoto and Oi [15] use the same approach using an energy-minimization function on the 2D correspondences. Tehrani et al. [16] propose an iterative method based on an energy minimization of a nonlinearly weighted Gaussian-based kernel density function applied on SIFT corresponding feature points. The main drawback of these two methods is the fact that the colorimetric correction is performed on each RGB channels independently.

Shao et al. [17] distinguish the foreground and the background parts from a precomputed disparity map. They perform a PCA-based colorimetric correction only on the forground parts that are more likely to appear on each view. Shao et al. [18] also requires a precomputed disparity map to perform the correction using a linear operator on the YUV colour space.

Finally, Shao et al. [19] present a content adaptive method that performs a PCA on the data in order to select the relevant colours of the scene and generate a 3×3 correction matrix. This method does not require any disparity map but is limited to linear correction.

2.3 Outline of Our Method

As specified above, 1-dimension LUTs applied independently on each RGB channel are known for their limited colorimetric correction accuracy, whereas 3D-LUT based methods are much more accurate, still fast and easy to use. We propose a method to generate such 3D-LUT by a non-linear process that minimizes the colorimetric properties differences between each image. A 3D-LUT with full resolution would imply $3 \times 256^3 \simeq 5.10^7$ variables involving extremely long computation times. Even a 3D-LUT with a standard resolution of 32^3 would result in $3 \times 32^3 \simeq 10^4$ variables that still can not be computed in a reasonable time delay.

In this paper, we introduce a substitution of the 3D-LUT by an orthogonal basis functions that can represent the initial 3D-LUT with very few variables. We present a non-linear minimization process that finds optimal values for these variables such the recovered 3D-LUT generates corrected images with similar colorimetric properties. In regard to the related works, our method does not require any precomputed disparity map, can handle non-linear corrections, does not consider each channel independently, generates a set of 3D-LUT and is fully compatible with SIFT or other point correspondences approaches.

This paper is organized as follows: In section 3, we introduce the Chebyshev polynomial basis. Section 5 describes the non-linear minimization process used for the colorimetric correction. Section 6 presents some optimizations to speedup the process and section 7 shows some results.

3 3D-LUT and Basis Functions

3.1 Basis Functions for 3D-LUT

The purpose of the basis function is to decrease the number of variables representing the 3D-LUT. The basis function should be orthogonal to ensure the unicity of the LUT representation. We selected Chebyshev polynomial basis for several reasons. Indeed, the first order polynomials have soft variations, hence higher order polynomials can be ignored without a significant loose on the 3D-LUT description. Moreover, polynomial basis functions can represent the identity function used for the initialization. Some other well known basis such as discrete cosine transform can not unless they use all the functions of the basis. Finally, each Chebyshev polynomials are alternatively odd and even such the first polynomials have a specific signification in term of colour processing, as presented in Table 1.

Table 1. Polynomial basis for 3D-LUT: a signification for the first degrees

Degree	Effect
0	colour offset
1	identy function
2	brightness/gain
3	contrast

3.2 Chebyshev Polynomial Basis

The Chebyshev polynomials are a sequence of orthogonal functions defined for $x \in [-1, 1]$ as:

$$T_n(x) = \frac{(x - \sqrt{x^2 - 1})^n + (x + \sqrt{x^2 - 1})^n}{2}$$

They can also be expressed recursively with:

$$\begin{cases} T_0(x) = 1 \\ T_1(x) = x \\ T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x) \end{cases}$$

Figure 1 depicts the first Chebyshev polynomials.

$T_0(x) = 1$
$T_1(x) = x$
$T_2(x) = 2x^2 - 1$
$T_3(x) = 4x^3 - 3x$
$T_4(x) = 8x^4 - 8x^2 + 1$
$T_5(x) = 16x^5 - 20x^3 + 5x$
$T_6(x) = 32x^6 - 48x^4 + 18x^2 - 1$
$T_7(x) = 64x^7 - 112x^5 + 56x^3 - 7x$
$T_8(x) = 128x^8 - 256x^6 + 160x^4 - 32x^2 + 1$
$T_9(x) = 256x^9 - 576x^7 + 432x^5 - 120x^3 + 9x$

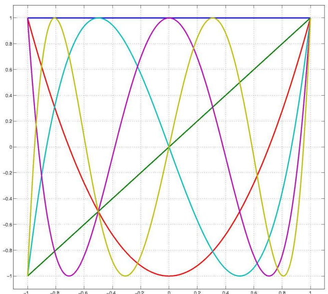

Fig. 1. Left: the first Chebyshev polynomials. Right: their graphical representation

3.3 3D-LUT Representation

Without loss of generality, we consider in the rest of this paper that the colour levels range from 0 to 1. We define a 3D-LUT f that transforms three input colours r, g and b into three output colours $(r', g', b')^\top = f(r, g, b)$ with the following formula:

$$\begin{pmatrix} r' \\ g' \\ b' \end{pmatrix} = \begin{pmatrix} \sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{k=0}^{n} \alpha_{i,j,k}^{r} T_i(r).T_j(g).T_k(b) \\ \sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{k=0}^{n} \alpha_{i,j,k}^{g} T_i(r).T_j(g).T_k(b) \\ \sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{k=0}^{n} \alpha_{i,j,k}^{b} T_i(r).T_j(g).T_k(b) \end{pmatrix} \tag{1}$$

where n is the higher polynomial degree used and $\alpha_{i,j,k}^{c}$ is the coefficient associated to the polynomial $T_i(x) \times T_j(y) \times T_k(z)$, for the output colour channel c. An illustration of a 2-dimension Chebyshev basis functions is depicted in Figure 2.

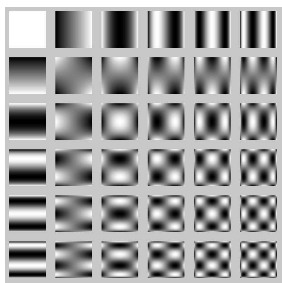

Fig. 2. A representation of the 2-dimensional Chebyshev basis with the 6 first levels, where each block $T_{ij}(x, y) = T_i(x) \times T_j(y)$. A coefficient $\alpha_{i,j}^{c}$ is associated to each block (in 3-dimension in our case) to represent a signal $g(x, y) = \sum_{i,j} \alpha_{i,j} T_{ij}(x, y)$.

The look-up table representation is defined by $3 \times n^3$ coefficients associated to the first n Chebyshev polynomials for the three channels r, g and b. A standard

3-dimension lookup table with n colour levels (i.e. 256 or 32) would involve $3 \times n^3$ variables. This representation also requires $3 \times n^3$ parameters, but the number n can be drastically diminished (e.g. $n = 7$). Moreover, the main response of the 3D-LUT is concentrated on the channel considered. Indeed, in (r, g, b), r is more significant than g and b to compute r' (respectively for g and b). Thus, it is possible to associate a higher accuracy for the channel considered rather than for the two other channels.

The conversion from the polynomial representation to the standard form is computed by applying equation (1) to all the lookup table elements.

4 Initialization

The 3D-LUT default initialization is the identity function, i.e. $(r, g, b)^\top = f(r, g, b)$. This configuration is obtained by using only the $L_1(x) = x$ polynomial for the channel related to the colour being processed. In term of coefficients, identity corresponds to:

$$\begin{cases} \alpha^r_{1,0,0} = \alpha^g_{0,1,0} = \alpha^b_{0,0,1} = 1 \\ \alpha^c_{i,j,k} = 0 \quad \text{otherwize} \end{cases} \tag{2}$$

It is also possible to convert an existing standard 3D-LUT to our model. Indeed, setting the initial function f with a good estimation of the expected lookup table will decrease the number of iterations required to reach convergence. Given a 1-dimensional lookup table $g(x)$ such that:

$$g(x) = \sum_{k=0}^{\infty} c_k L_k(x)$$

the coefficients c_k can be found by ([20], p.67):

$$c_k = \frac{4}{\pi} \int_{-1}^{1} \frac{g(x).T_k(x)}{\sqrt{1-x^2}} dx$$

except for c_0 that should be divided by 2.
The discreet form with n discretization steps is:

$$c_k = \frac{4}{\pi n} \sum_{i=1}^{n-1} \frac{g(2\frac{i}{n} - 1).T_k(2\frac{i}{n} - 1)}{\sqrt{1 - (2\frac{i}{n} - 1)^2}}$$

Actually, a much faster estimation of c_k is given by:

$$c_k \simeq \frac{2}{n+1} \sum_{i=0}^{n} g\Big(\cos\frac{\pi i}{n}\Big).\cos\frac{\pi k i}{n} \tag{3}$$

still with c_0 divided by 2.

To perform this stage with a 3-dimensional lookup table, the previous method should be repeated on the three dimensions.

5 Non-Linear Process

5.1 Image Descriptors

Let $\{I_{i=1...k}\}$ be a set of k images and the vector $\mathbf{x} = D(I)$ a representation of the colorimetric properties of an image I. The vector $\mathbf{x}_i$ is a concatenation of measures on the image I_i. The purpose of the minimization process is to find the 3D-LUT such the transformed images provide similar vectors $\mathbf{x}$. As a minimal setup, we propose the following measures:

- **image average color**: returns a value (in $[0, 1]$) for each r, g and b channel.
- **image saturation**: returns a single value (in $[0, 1]$) corresponding to the average saturation per pixel. A pixel saturation is computed as the variance of the r, g and b channels.
- **image contrast**: returns a single value (in $[0, 1]$) corresponding to the variance of the image 3D histogram.

In this paper, we mainly focus on these three measures, however any other measures satisfying distances properties can be added in the vector $\mathbf{x}$.

5.2 Minimization

The non-linear process consists in finding the parameters $\alpha^c_{i,j,k}$ representing a 3D-LUT that transforms the input images $I_{i=1...k}$ such the $\mathbf{x}_{i=1...k}$ become similar. Thus, this process is equivalent to minimize the cost function $M(\{\mathbf{x}_{i=1...k}\})$:

$$M(\{\mathbf{x}_{i=1...k}\}) = \|\sigma(\mathbf{x}_i)_{i=1...k}\| \tag{4}$$

Where $\sigma(\mathbf{x}_i)$ denotes the variance of the vectors $\mathbf{x}_i$. This approach makes the corrected images to have their descriptors converging to an average value $\widehat{\mathbf{x}}$. Another possibility is to select a reference image I_r whose descriptor $\mathbf{x}_r$ will be considered as a target for the other images during the minimization process. The function $M_r(\{I_{i=1...k}\})$ becomes:

$$M_r(\{\mathbf{x}_{i=1...k}\}) = \sum_{\substack{i=1\\ i\neq r}}^{k} \Big(\mathbf{x}_i - \mathbf{x}_r\Big)^2 \tag{5}$$

5.3 Point Correspondences

The minimization process can be performed on the whole images but can also be restricted on a set of selected areas. In that situation, point correspondences can be found using usual techniques such SIFT [14] or SURF [21]. Applying the minimization on a restricted set of areas on the images presents some advantages about robustness. Indeed, if a colour appears only on an image but not on the others, this colour will not be selected and hence will not contribute to the colorimetric correction. However, the risk of this method is to limit the diversity of colours encountered in the areas and hence to decrease the accuracy of the colorimetric correction.

6 Optimizations

The minimization process is sometimes long to compute, hence we propose in the following sections some optimizations to speedup the process to reach convergences. None of these methods affect the accuracy of the final results.

6.1 Initialization

As described in section 4, the initial parameters $\alpha^c_{i,j,k}$ are setup with equation (2) such the resulting 3D-LUT represents the identity function. An alternative is to start the iterative process from a solution that is fast to compute and not too far from the expected solution. In practice, we first compute an histogram equalization for each r, g and b channels, resulting in three 1-dimensional LUT. The corresponding coefficients $\alpha^c_{i,j,k}$ are extracted from these LUTs using equation (3).

6.2 Histogram Domain

Most of the descriptors presented in section 5.1 require the computation of a 3D histogram and the remaining descriptors can be computed from these histograms. Hence, the successive 3D-LUT computed during the non-linear process are directly applied on the histograms rather than to apply them on the images and then extract the histograms. Moreover, the 3D histogram data is stored on a 1D array with size equal to the number of the different colours appearing in the image. Thus, in the worth case (all pixels have different colours), this array has the same size as the image. Since a LUT is a surjective function, the size of the array will never increase during the iterative process. Finally, avoiding to apply the lookup tables to the images makes the computation time independent from the images'resolution and hence makes possible to work on high resolution images.

6.3 Pyramidal 3D Histograms

Finally, we use a pyramidal method on the iterative process. During the first iterations, the 3D histograms are quantized to decrease their size of 80%, involving a fast but inaccurate convergence. The quantization effect is progressively decreased during the iterations such that the last iterations become slower but use the image data with full details. The effect of this pyramidal method is first to speed up the computation during the first iterations and second to speed up total convergence.

7 Tests and Results

We implemented our minimization method in C++, with Levenberg-Marquardt minimization algorithm as described in [22] (p. 600). We tested our method on a set of images with different colorimetric properties and geometrically rectified

Table 2. Computation time with and without optimizations presented in section 6. The image resolution for the RGB and non-optimized version are reduced to 400×266 to get acceptable computation time, whereas the optimized version run on 2300×1533 resolution images. Moreover, using only 3×7^3 parameters for the RGB-LUT leads to unsatisfactory results.

number of variables	standard RGB 3D-LUT	without optimization	pyramidal histogram	pyramidal histogram + initialization
$3 \times \mathbf{3}^3$	18 min	13 min	5 min	5 min
$3 \times \mathbf{5}^3$	230 min	124 min	58 min	54 min
$3 \times \mathbf{7}^3$	960 min	416 min	331 min	177 min

with [12]. Figure 4 depicts a result using the Chebyshev basis function with the degree 7, with a reference image as in equation (5). The resulting 3D-LUT for some images is shown in Figure 5 and clearly underline the non-independence between each channels.

The computational time is still long, Table 2 presents the computational time of our method, with and without the optimizations presented in section 6. As a comparison, our optimized method with 7 polynomials takes less than 3 hours to compute high resolution images when the direct RGB-3D-LUT computation

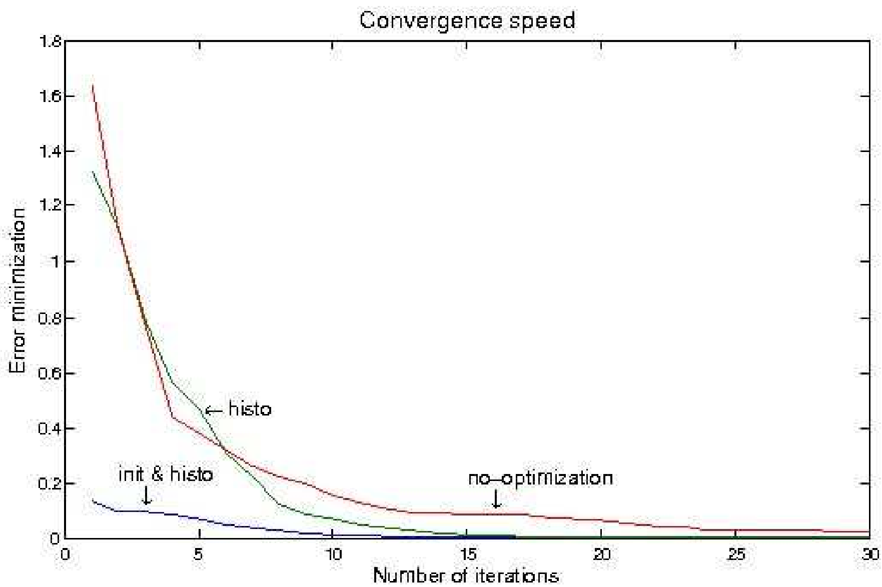

Fig. 3. Convergence speed with 3×7^3 polynomials for the methods using just the Chebyshev polynomial basis, the method with the 3D-histogram optimization and the method that also includes a 1D-LUT initialization

Fig. 4. Left: four input images with different colorimetric properties. Right: the corrected images with 7 polynomials using the top left image as a reference, with equation (5).

with 3×7^3 elements and low resolution images takes more than 16 hours, for very low quality results.

Figure 3 shows the minimization convergence speed comparison between our methods with or without the 3D histogram optimization and the initialization, for 3×7^3 polynomials. The use of the initialization drastically decreases the number of iterations required to reach convergence. The 3D-histogram optimization does not decrease the number of iterations, but reduces the computation time of an iteration. During the tests, we tried our method using L*a*b* colour space instead of rgb but we didn't noticed any changes in the results neither on the computation time.

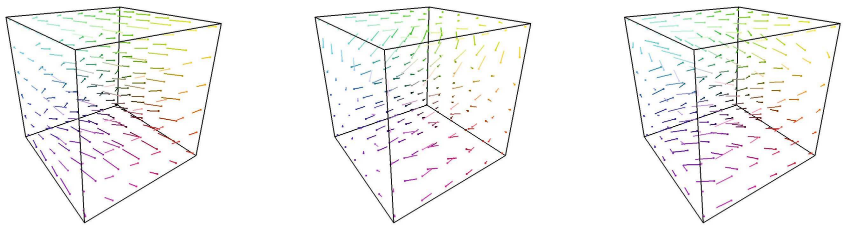

Fig. 5. 3D-LUT corresponding to the three corrected images of Figure 4

8 Conclusion

This paper presents a method to perform a colorimetric correction for a set of images captured with different cameras for stereoscopic purposes. The method produces a 3D lookup table that can be used in real-time on the GPU. This lookup table is represented by a basis function to reduce the number of required parameters. These parameters are computed by a non-linear method to minimize the difference of the colorimetric properties between the considered images.

In order to speedup the minimization process, we consider a compact form of the 3D histogram of the images rather than the images by them-self. This technique makes the process much faster and independent of the images resolution. The minimization process can start from the identity 3D lookup table or from any lookup table. Our tests show that using a fast 1D lookup table as an approximation of the results makes a very suitable starting point for our minimization process and produces a very fast convergence.

References

1. Lambooij, M.T.M., IJsselsteijn, W.A., Heynderickx, I.: Visual discomfort in stereoscopic displays: a review. Stereoscopic Displays and Virtual Reality Systems XIV 6490 (2007)
2. Smolic, A., Müller, K., Stefanoski, N., Ostermann, J., Gotchev, A., Akar, G.B., Triantafyllidis, G.A., Koz, A.: Coding algorithms for 3dtv - a survey. IEEE Trans. Circuits Syst. Video Techn. 17, 1606–1621 (2007)
3. Nozick, V., Saito, H.: Online Multiple View Computation for Autostereoscopic Display. In: Mery, D., Rueda, L. (eds.) PSIVT 2007. LNCS, vol. 4872, pp. 399–412. Springer, Heidelberg (2007)
4. Joshi, N., Wilburn, B., Vaish, V., Levoy, M., Horowitz, M.: Automatic color calibration for large camera arrays. Technical report, UCSD CSE Tech Report CS2005-0821 (2005)
5. Reinhard, E., Ashikhmin, M., Gooch, B., Shirley, P.: Color transfer between images. IEEE Comput. Graph. Appl. 21, 34–41 (2001)
6. Papadakis, N., Edoardo, P., Caselles, V.: A variational model for histogram transfer of color images. IEEE Transactions on Image Processing 20, 1682–1695 (2011)
7. Morovic, J., Sun, P.: Accurate 3d image color histogram transformation. Pattern Recognition Letters 24, 1725–1735 (2003)

8. Neumann, L., Neumann, A.: Color style transfer techniques using hue, lightness and saturation histogram matching. In: Proceedings of Computational Aesthetics in Graphics, Visualization and Imaging, pp. 111–122 (2005)
9. Pitié, F., Kokaram, A.C., Dahyot, R.: Automated colour grading using colour distribution transfer. Computer Vision and Image Understanding 107, 123–137 (2007)
10. Abadpour, A., Kasaei, S.: An efficient pca-based color transfer method. J. Visual Communication and Image Representation 18, 15–34 (2007)
11. Adabpour, A., Kasaei, S.: A fast and efficient fuzzy color transfer method. In: Proceedings of the IEEE Symposium on Signal Processing and Information Technology, pp. 491–494 (2004)
12. Nozick, V.: Multiple view image rectification. In: Proc. of IEEE-ISAS 2011, International Symposium on Access Spaces, Yokohama, Japan, pp. 277–282 (2011)
13. Yamamoto, K., Yendo, T., Fujii, T., Tanimoto, M., Suter, D.: Colour correction for multiple-camera system by using correspondences. The Journal of the Institute of Image Information and Television Engineers 61, 213–222 (2007)
14. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vision 60, 91–110 (2004)
15. Yamamoto, K., Oi, R.: Color correction for multi-view video using energy minimization of view networks. International Journal of Automation and Computing 5, 234–245 (2008)
16. Tehrani, M.P., Ishikawa, A., Sakazawa, S., Koike, A.: Iterative colour correction of multicamera systems using corresponding feature points. Journal of Visual Communication and Image Representation 21, 377–391 (2010)
17. Shao, F., Peng, Z., Yang, Y.: Color correction for multi-view video based on background segmentation and dominant color extraction. WSEAS Transactions on Computers 7, 1838–1847 (2008)
18. Shao, F., Jiang, G.Y., Yu, M., Ho, H.S.: Fast color correction for multi-view video by modeling spatio-temporal variation. Journal of Visual Communication and Image Representation 21, 392–403 (2010)
19. Shao, F., Jiang, G., Yu, M., Chen, K.: A content-adaptive multi-view video color correction algorithm. In: IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 1, pp. 969–972 (2007)
20. Gil, A., Segura, J., Temme, N.M.: Numerical Methods for Special Functions, 1st edn. Society for Industrial and Applied Mathematics, Philadelphia (2007)
21. Evans, C.: Notes on the opensurf library. Technical Report CSTR-09-001, University of Bristol (2009)
22. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision. 2nd edn. Cambridge University Press (2004) ISBN: 0521540518