Signal Processing: Image Communication 74 (2019) 231–241

Contents lists available at ScienceDirect

Signal Processing: Image Communication

journal homepage: www.elsevier.com/locate/image

Robust global and local color matching in stereoscopic omnidirectional content

Roman Dudek [a,*], Simone Croci [b], Aljosa Smolic [b], Sebastian Knorr [b,c]

[a] Universidad de Las Palmas de Gran Canaria, Spain
[b] V-SENSE, School of Computer Science and Statistics, Trinity College Dublin, The University of Dublin, Ireland
[c] Communication Systems Group, Technical University of Berlin, Germany

ARTICLE INFO

Keywords:
Virtual reality
360-video
Color matching
Binocular rivalry
Omnidirectional images
Stereoscopic 3D

ABSTRACT

Shooting a live-action immersive 360-degree experience, i.e. omnidirectional content (ODC) is a technological challenge as there are many technical limitations which need to be overcome, especially for capturing and post-processing in stereoscopic 3D (S3D). In this paper, we introduce a novel approach and entire system for stitching and color mismatch correction and detection in S3D omnidirectional content, which consists of three main modules: pre-processing, spherical color correction and color mismatch evaluation. The system and its individual modules are evaluated on two datasets, including a new dataset which will be publicly available with this paper. We show that our system outperforms the state of the art in color correction of S3D ODC and demonstrate that our spherical color correction module even further improves the results of the state of the art approaches.

1. Introduction

In contrast to traditional cinema, where the viewer perceives the world through a window i.e. the cinema screen, virtual reality (VR) allows a person to be present within the world by wearing a head-mounted display (HMD) [1]. One of the most popular formats to deliver VR content is 360-video, also called omnidirectional video (ODV), VR-video or cinematic VR. Shooting a live-action immersive 360-degree experience, i.e. omnidirectional content (ODC) is a technological challenge as there are many technical limitations which need to be overcome, especially for capturing and post-processing in stereoscopic 3D (S3D). 360-video is mostly captured with an omnidirectional multi-camera rig and stitched together in post-production [2–5]. In general, such limitations result in artifacts which cause visual discomfort when watching the content with an HMD. The artifacts or issues can be divided into three categories: binocular rivalry issues (e.g. color mismatch), conflicts of depth cues (e.g. vergence-accomodation conflicts) and artifacts which occur in both monocular and stereoscopic 360-degree content production (e.g. stitching artifacts) [6].

In this paper, we focus on binocular rivalry issues, in particular on color mismatch detection and correction in S3D omnidirectional content. Color mismatch in multi-camera systems is an inherent problem due to different camera and lens characteristics, different illumination and reflections resulting from different camera orientations, etc. Such color mismatches occur during the stitching and blending process of multiple views into a single monocular panorama, but also between the left and right view of a stereoscopic panorama, which often results in visual discomfort [7].

In this context, we introduce a novel approach and entire system for stitching and color mismatch correction and evaluation in S3D omnidirectional content. The system consists of three main modules: pre-processing, spherical color correction and color mismatch evaluation as shown in Fig. 1. During the pre-processing step, multiple camera views are geometrical aligned and stitched together to S3D omnidirectional images (ODIs). A global color matching (GCM) module is applied in order to reduce substantial color mismatches between the different camera views.

As global color matching cannot handle scene dependent issues like lens flares and polarization issues properly, as they mainly occur locally, we implemented a spherical color correction step consisting of a local color matcher with spherical adaption (LSCM). The local color matcher is based on optical flow to estimate pixel correspondences between the left and right stereoscopic views.

Finally, our patch-based color mismatch evaluation module [8], which is used for validation purposes, measures and visualizes color mismatch, which might still be present, between the left and right stereoscopic ODIs. While the entire color correction approach is applied in the RGB color space and uses an optical flow approach introduced in [9], the color mismatch evaluation module is applied in the Lab color

* Corresponding author.
E-mail addresses: roman.dudek101@alu.ulpgc.es (R. Dudek), crocis@scss.tcd.ie (S. Croci), smolica@scss.tcd.ie (A. Smolic), knorr@nue.tu-berlin.de (S. Knorr).

https://doi.org/10.1016/j.image.2019.02.013
Received 3 September 2018; Received in revised form 2 January 2019; Accepted 27 February 2019
Available online 5 March 2019

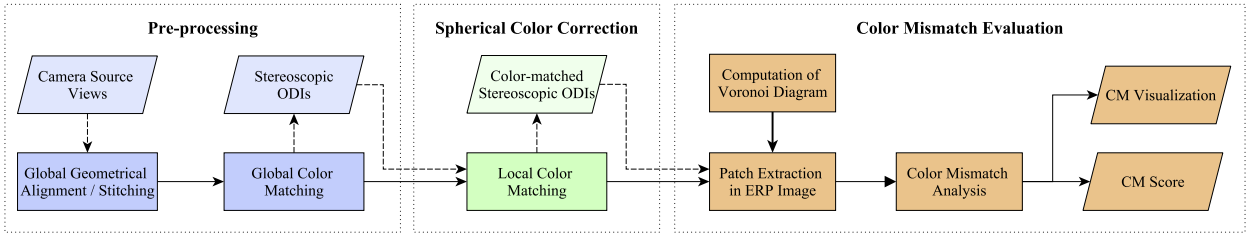


Fig. 1. Overall color mismatch correction system.

space and uses semi-global block matching as introduced in [10]. This allows a more objective and independent evaluation of still existing color discrepancies between the views.

Each of the modules works independently and could be used in different applications. For instance, the spherical color correction as well as the patch-based color evaluation module can be applied on already existing S3D omnidirectional datasets.

Then, we evaluate the system and individual modules on two datasets (a new dataset with six sequences captured with different multi-camera rigs and a dataset with 15 ODIs introduced in [11]) and compare it against the state of the art, namely the professional solutions Cara VR[1] and Google Jump[2] which is based on the work in [4], and the color correction approach recently introduced in [11]. We will show that our entire system outperforms all three methods and demonstrate that our individual spherical color correction module even further improves the results of the state of the art approaches. Furthermore, we will also show that our local color matcher without spherical adaption even outperforms the method introduced in [3] for rectlinear and cylindrical panoramas.

Finally, we make the new dataset consisting of six 360-degree sequences including the source views and stitched results using our approach and the state of the art approaches publicly available with this paper.

The contribution of this work can be summarized as follows:

- A novel and entire system for color correction and evaluation in S3D omnidirectional content.
- A novel and efficient global color matching approach for ODIs.
- A novel and efficient local color matching approach with spherical adaption.
- A comprehensive analysis of our proposed method against the state of the art.
- A new dataset with six 360-degree sequences.

The remainder of the paper is organized as follows. In Section 2, related work in artifact detection and color correction is reviewed. Then, in Section 3, we describe the proposed system for stitching, color correction and color mismatch evaluation. In Section 4, we evaluate the entire system and independent modules against the state of the art with a comprehensive amount of data. Finally, in Section 5, the paper concludes with a summary and discussion.

2. Related work

Over the recent years, binocular rivalry between stereoscopic images has been investigated in detail for traditional S3D content, e.g. for cinema screens [7,12] and 3D-TV [13,14], and more recently for omnidirectional content for HMDs [15,6]. A major objective for the assessment of S3D quality is to develop objective metrics for view asymmetries which are highly correlated to user scores of subjective quality tests.

In [16], the authors investigated with subjective tests how geometrical misalignments and color/luminance mismatch between the views have an influence on the viewer annoyance and compared the results with objective metrics. The authors of [17] proposed several objective metrics for luminance mismatch and evaluated their correlation with the results of subjective experiments. In [18], a method for detecting stereo camera distortions based on statistical models was presented in order to evaluate vertical misalignment, camera rotation, unsynchronized zooming, and color mismatch in S3D content.

A large variety of artifact detection methods, including a method for the detection of color mismatch, was introduced in [19]. The authors used the disparity to reconstruct one view from the other and compared the colors from the original and the reconstructed view based on the mean square error in the RGB color space. More recently, in [6,20], S3D quality assessment methods for stereoscopic ODIs were introduced which also focus on color mismatch detection.

In the computer vision and multi-view video processing communities, the initial efforts on solving color mismatches between multiple views used exposure compensation (or gain compensation) [21]. This approach adjusts the gain level of images to compensate for appearance differences caused by different exposure levels. However, this approach may fail in the case of local differences e.g. caused by lens flares or polarization.

The authors of [22] propose a simple method to compute 3D lookup tables with a non-linear process that minimizes the colorimetric properties of the source images. Wang et al. [23] proposed a robust algorithm to correct the color discrepancy between images, which neither requires a color calibration chart/object, nor explicitly compensates for the image as a whole. Instead, they correct the image region by region using local feature correspondences.

Dudek et al. [9] proposed a combination of a global and local color correction approach. While the global color correction is performed using a classic histogram based method for the entire image, the local color correction is a region-based approach using optical flow estimation. In [24], a method is proposed that combines global and local color information to correct color discrepancies between stereoscopic image pairs. The algorithm uses dense stereo matching and global color correction to initialize color values, and then improves the local color smoothness and global color consistency of the resulting image while maintaining the initial color as much as possible.

For large baseline multi-view video, Ye et al. [25] introduced a robust color correction method that enforces spatio-temporal color consistencies and gradient preservation by solving a global optimization problem. The authors of [26] proposed an effective color correction method for multi-view image stitching which first finds coherent content regions in inter-image overlaps, where reliable color correspondences are extracted, and then parameterizes a color remapping curve as transform model, and expresses the constraints of color consistency, contrast and gradient in an uniform energy function.

The image processing and computer graphics communities were developing similar color manipulation methods, called *color transfer* techniques. These methods transfer the color feel from a palette image to a target image, and assume that the content of the images is different. The earliest work in this area was by Reinhard et al. [27], who proposed transforming the mean and standard deviation of each color channel in

[1] https://www.foundry.com/products/cara-vr.
[2] https://vr.google.com/jump/.

the target image to match that of the palette image. Since then, more complex techniques have been used to model the color distributions of the images more accurately, including histograms and Gaussian Mixture Models [28,29]. While global color transfer functions are often used, including affine, radial basis and optimal transport functions [30–32], local techniques have also been proposed to allow for more flexibility in the recoloring [33,34]. Recently, Grogan and Dahyot [35,36] proposed a color transfer technique that could also be enhanced to take into account color correspondences between the target and palette images, ensuring the method could be used to color correct images of the same scene. They showed that this method performed as well as other state of the art color corrections techniques, with the advantage of being more robust to correspondence outliers. In Croci et al. [11], the authors extended this work and demonstrated that it can be applied to also color correct stereoscopic ODIs. Besides a comparison of our approach with current professional solutions like Cara VR and Google Jump [4] for video sequences, we will also compare our local color matching approach against the approach introduced in [11] using the same dataset. Finally, we compared our LCM method, i.e. without spherical extension, against the method described in Zhang and Liu [3].

Our entire proposed system is most related to the work in [4] with the addition of applying local color correction to the finally stitched and globally color corrected ODVs.

3. Proposed method

Our overall approach consists of the following main modules: global geometric alignment and stitching, global color matching (GCM), local spherical color matching (LSCM), and independent color mismatch evaluation. The entire system is illustrated in Fig. 1 and will be described in detail in the following subsections.

3.1. Pre-processing

3.1.1. Global geometrical alignment and stitching

As the whole process of geometrical alignment, stitching and blending is beyond the scope of the paper but an important pre-processing step, we describe this process briefly. Stitching, or combining of partial views into one spherical image [37,38,2,4,39,5], requires a precise calibration of the camera model. In general, each camera of an omnidirectional camera rig is modeled by a set of parameters: intrinsic parameters (sensor resolution, sensor center, focal length, lens distortion and field of view) and extrinsic parameters (focal point, camera orientation). Methods to calculate these parameters consist of finding point correspondences between the different views, a global numerical solver method and an outlier removal step to handle false point correspondences. A good overview can be found in [38].

Finding initial point correspondences between the camera views, however, can be very challenging without any knowledge about the used camera rig. Thus, rig manufacturers often provide so-called rig templates, which are nominal models expressing the rig geometry, and which are used as basis for an iterative refinement to match the mechanical variations of the actual rig unit.

After calibration, the partial views can be mapped onto a spherical surface to produce an omnidirectional image (ODI). For overlapping areas between the views, multiple possible pixel values are available. A simple approach would be to select the pixel value corresponding to the nearest view, i.e. the view with the smallest angular distance between its center and the pixel location in the overlapping area. Thus, the ODI consists of a mosaic of similar sized adjacent patches, each representing one camera view as illustrated in Fig. 2.

However, even with a perfectly calibrated camera model, it is impossible to map the 3-dimensional scene correctly onto a spherical surface due to the parallax between the camera views which do not share the same center of projection [40]. Hence, artifacts will occur at the stitch lines.

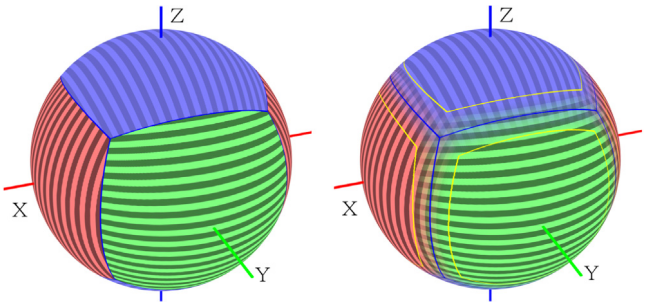


Fig. 2. Example of stitching for a 6 camera cube rig design. Stitched camera views in a sphere without blending (left) and with blending (right). The region between the yellow lines is the blending area.

To reduce the stitching artifacts, pixel blending within the overlapping areas is commonly applied as illustrated in Fig. 2, where the area between the yellow lines is mapped and blended from different camera views. Cross-blending of pixel values from different views, however, may result in ghosting, i.e. double images in highly textured regions [41]. Also, other artifacts like seams within objects might occur if such objects cross these stitching regions due to imperfect scene segmentation [42].

To reduce artifacts like double edges or visible seams within objects, we use a local image alignment, which is based on optical flow estimation [9], and correct the parallax along stitch lines. Additionally, we apply blending to further reduce the visibility of the stitches caused by color mismatches between different views.

3.1.2. Global color matching

We found that the global mismatch between the individual cameras is owed principally to each camera separately adjusting its “gain” and “color temperature”. Even if many rigs support a joint regulation of these settings, they are frequently left intentionally to drift separately between cameras: Adjusting all cameras for the worst case, like when one camera is pointing to the sun, would result in quite poor settings for the camera pointing to a dark floor, for example.

Both gain and color temperature are basically scale factors applied to the camera image: Gain is a linear scale of the signal, while temperature setting basically balances the scale of the red component versus the blue component to compensate the type of lighting (e.g. sunny versus cloudy, fluorescent versus tungsten) of the scene. As a consequence, the pixel values corresponding to the same point of the scene but captured by different cameras of the rig, can differ substantially. However, knowing that the main reason is that of each camera applying a scale factor to all its pixels, we can reasonably assume that the relation between pixel values for the same point of the scene will be that of a scale.

In our proposed GCM method, to minimize color mismatches between the camera views, we try to find such a set of scale factors $S_i = (S_i^r, S_i^g, S_i^b)^T$, where $\{r, g, b\}$ represents the color channels for each camera C_i of the rig with $i = 1, \ldots, m$. Before deriving the color matching process, however, we need to take into account how the pixel values are encoded, so that pixel values can be correctly compared and eventually matched. The image pixel values produced by the cameras are mostly gamma encoded in logarithmic scale in order to reduce the volume of the data. In such a case, we first need to convert all of the camera source images I_i, from their gamma encoded pixel values, into linear light encoded images L_i as follows:

$$L_i = E^{-1}(I_i), \tag{1}$$

where $E^{-1}(I_i)$ is the inverse luminance transfer function applied to image I_i.

A large amount of luminance transfer functions $E(L)$ or “gamma curves” exists as almost every camera brand defines its own functions.

As an example, a commonly used transfer function is the "ITU-R Recommendation BT.709 transfer function" [43] defined as:

$$E(L) = \begin{cases} 4.500L & \text{for } L < 0.018 \\ 1.009L^{0.45} - 0.099 & \text{for } L \geq 0.018 \end{cases} \quad (2)$$

where L is the linear light encoded image and normalized to $\{0, 1\}$.

Our proposed GCM method needs to take into account the use of equidistant cameras with wide-angle lenses, which are commonly used in current omnidirectional camera rigs. We implemented an iterative method where we match one camera versus all the other cameras by mapping their images into a common plane. To avoid the complexity of working in an omnidirectional, spherical image space, we utilize the fact that, while each camera covers a segment of a sphere, the image captured by each camera sensor is a planar, 2-dimensional image with limited non-linearity. Basically, we split the sphere into a set of planar, overlapping patches equivalent to the sensor spaces and dimensions. For each camera C_i and its image L_i, in its sensor space with $i = 1, \ldots, m$ as shown in Figs. 3(a) and 3(c), we create a combined spherical stitched image of all cameras C_j with $j = 1, \ldots, m$ and $j \neq i$ as shown in Fig. 3(b). We then re-project the result as image L'_i into Cartesian coordinates of camera view C_i as illustrated in Fig. 3(d): Note that this Fig. 3(d) shows how the other five camera views overlap with the current camera view. In our example, only four of the other cameras can be seen, as the fifth camera faces in the opposite direction and has no overlap at all.

$$L'_i = \Omega_{\forall j \neq i}(L_j, C_j) \quad (3)$$

The $\Omega()$ function represents the entire spherical stitching and re-projection operation, combining all the camera views except camera view C_i. We use a simplified, fast stitching operation $\Omega()$ without use of registration, as the resulting L'_i will be used only for iterative color matching of image L_i.

The process behind the $\Omega()$ function starts with an empty ODI and iteratively adds the individual re-projected camera views, calculates the distances to the camera centers for any point covered by more than one camera view, and chooses the nearest one, i.e. performs an image mosaicing to map patches into the image plane similar to [44], but adapted to the spherical space. To avoid hard cuts between the patches, cross-blending is used at the edges of the patches, where the distance to the nearest cameras is similar.

Finally, we compare and match the pixels of image L_i with the combined image L'_i representing all the other cameras views. Based on the comparison result, we modify the scale factors of camera C_i.

Depending on the degree of the overlap between the camera views C_j, image L'_i consists of valid pixels and unfilled pixels, usually resulting in a hole in the center of the sensor space of the current camera C_i as illustrated in Fig. 3(d). On the other side, image I_i has valid pixels only inside a circular area given by the lens view angle, commonly smaller than the whole sensors area as shown in Fig. 3(c). Hence, we can create a mask W_i as intersection of pixels valid in both L_i and L'_i as depicted in Fig. 3(e):

$$W_i = \begin{cases} 1 & \text{where both } L_i \text{ and } L'_i \text{ are defined} \\ 0 & \text{otherwise} \end{cases}. \quad (4)$$

We then compare the pixel values of L_i and L'_i only inside this mask W_i, i.e. pixel values which are defined in both images.

Assuming that the relation between the pixel values for the same scene point captured by different cameras is basically that of scale, we need to compute the average scale factor, i.e. geometric mean, in the overlap area between cameras. However, we substituted the geometric mean by a computationally more convenient arithmetic mean using the equivalency $\prod x = e^{\sum \log(x)}$.

As the gain is often expressed in logarithmic units of "stops", where one 'stop" unit is defined as doubling the light amount, we substitute the linear scale factors S_i by "stop" unit factors $G_i = \log_2(S_i)$. This

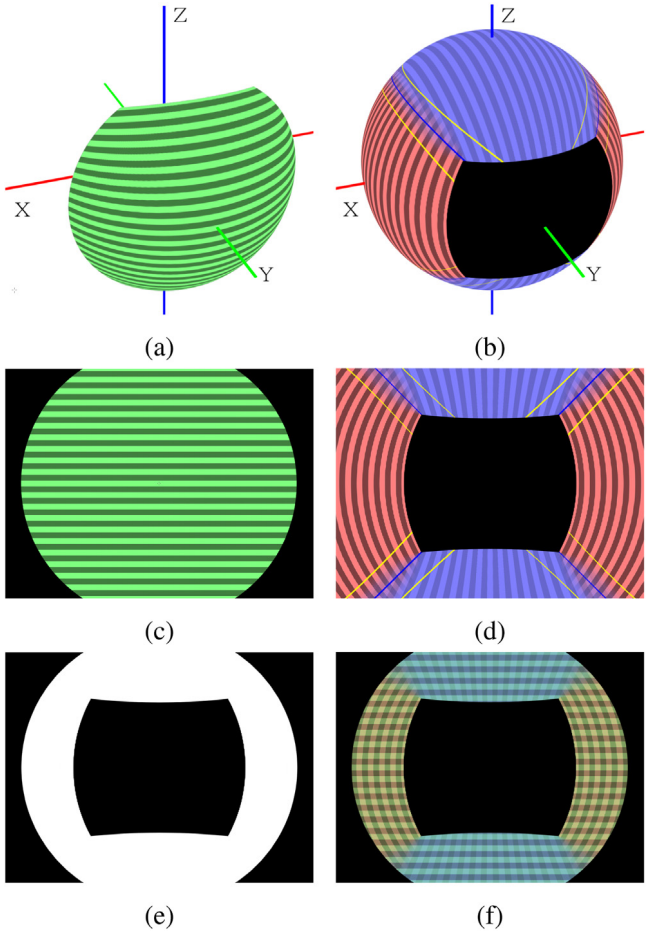


Fig. 3. Visualization of GCM in spherical coordinates and in camera space of current camera C_i: (a) Image of current camera C_i mapped onto a sphere, (b) all images of cameras $C_{j\neq i}$ mapped onto a sphere, (c) image L_i from camera C_i in sensor space, (d) image L'_i in sensor space of camera C_i, (e) overlap mask W_i between L_i and L'_i, (f) L_i and L'_i in mask W_i.

enables us to perform the whole calculation in logarithmic scale. The average scale $\Delta G_{i(k)}$ is then calculated as follows:

$$\Delta G_{i(k)} = \sum_{\forall xy} W_i \cdot \log_2 \frac{L_i \cdot 2^{G_{i(k)}}}{L'_{i(k)}}, \quad (5)$$

where k is the iteration step, i.e. $\Delta G_{i(k)}$ is updated after each iteration. Notice that the image $L'_{i(k)}$ is created by stitching all images but L_i while applying their corresponding gains $G_{i(k)}$, so it will change for each iteration k. Using the Newton method, we calculate the corrected gain $G_{i(k+1)}$ for the camera C_i with:

$$G_{i(k+1)} = G_{i(k)} + \alpha \cdot \frac{\Delta G_{i(k)}}{\sum_{\forall xy} W_i}, \quad (6)$$

where α is the step size of the Newton's method, and the weighted sum $\Delta G_{i(k)}$ is normalized by the sum of all weights W_i.

We did tests using optical flow based registration for the L_i match as well as part of the $\Omega()$ function (Eq. (3)). Therefore, we measured the average difference between the GCM values obtained with and without registration using the dataset described in Table 2. The results are presented in Table 1. We found out that the resulting GCM values differ only marginal between both cases. This can be explained with Eq. (5) which can be restated as two independent sums (see Eq. (7)), making it easier to understand that it is basically a simple comparison of the overall brightness ratio across the overlap area of the camera

Table 1
Comparison of the standard deviation of GCM G_i values with and without registration. Notice that the average G_i should be close to zero as the GCM corrections are designed to be relative to their average. We calculated the standard deviation of the differences for each camera G_i.

ODV	With registration	No registration	Diff
1	0.03131	0.03414	0.00162
2	0.22216	0.22400	0.00159
3	0.54748	0.54547	0.00130
4	0.42168	0.42681	0.00324
5	0.51241	0.51897	0.00325
6	0.14870	0.16200	0.00496
Avg.	0.21031	0.20867	0.00141

Table 2
Dataset of used ODVs.

ODV	Camera type	Camera res.	Geometry	ODV res.
1	Insta360 Pro	3200 × 2400	6 × 1	4096 × 2048
2	iZugar Z6X3DC	2704 × 2028	3 × 2	4096 × 2048
3	Google Odyssey	2704 × 2028	16 × 1	5760 × 2880
4	Google Odyssey	1920 × 1440	16 × 1	5760 × 2880
5	Google Odyssey	2704 × 2028	16 × 1	5760 × 2880
6	Vuze	1600 × 1088	4 × 2	4096 × 2048

Algorithm 1: GCM

Objective: To minimize the gain mismatch ΔG_i between the views.
Input: Cameras C_i, images I_i with $i = 1, ..., m$, camera parameters.
Initialization: $k = 0$, $G_{i(k)} = (0, 0, 0)^T$
while $\Delta G_{i(k)} > q$ **do**
  for *(i = 1 to m)* **do**
    Create image $L'_{i(k)}$ using gains $G_{i(k)}$;
    Calculate $\Delta G_{i(k)}$ (Eq. (5));
    Calculate $G_{i(k+1)}$ (Eq. (6));
  end
  Increment k by 1;
end
Create globally color matched images M_i applying the gains $G_{i(k)}$ to images I_i (Eq. (8));
Output: The globally color matched images M_i.

image and the combined remaining cameras in logarithmic scale:

$$\Delta G_{i(k)} = G_{i(k)} \sum_{\forall xy} W_i \cdot \log_2 (L_i) - \sum_{\forall xy} W_i \cdot \log_2 (L'_{i(k)}). \quad (7)$$

We then iteratively repeat the global color matching for each camera as described until a given degree of convergence $q < 0.0039$ with $q = \Delta G_{i(k)} - \Delta G_{i(k-1)}$. We have chosen as reference one quantization step of the source images (1/256 of the encoding range). As a final result of GCM, we create globally color matched images M_i by applying the gains $G_{i(k)}$ to all images L_i with $i = 1, \dots, m$ and transferring the linear encoded and color matched images back into the original logarithmic scale as follows:

$$M_i = E\left(L_i \cdot 2^{G_i}\right). \quad (8)$$

The speed of convergence can vary depending on the topology of the rig and the lighting conditions of the scene. However the process is computationally not complex and converges in short time. The whole algorithm is summarized in Algorithm 1.

Usually, GCM is only performed with one frame and then applied for the entire shot. However, if the illumination of the scene changes considerably over time, the cameras auto-adjustment may gradually follow the new lighting conditions, requiring to apply GCM at certain time intervals. For intermediate frames of these intervals, parameter interpolation can then be applied.

Typically our GCM process converges sufficiently after 5 to 15 iterations for the first frame. When GCM is also applied at time intervals, we can use the previous converged GCM parameters as starting point for the GCM at new frame position, reducing the number of iterations for this new position. Finally, after GCM is applied to all source camera images, the ODIs are stitched together as described in Section 3.1.1.

Our GCM method uses a single 3-component gain per image as a model. Much more complex global matching methods could be applied. However, there are important sources of color imbalance, like e.g. lens flare and polarization issues, that are inherently local, and which cannot be addressed by global color matching alone. Thus, we further concentrated on local color matching to further improve the results instead of further refining GCM.

3.2. Local spherical color matching

In order to deal with local color mismatches and scene dependent anomalies like lens flare and polarized light, we introduce a novel local spherical color matching (LSCM) approach.

When polarized light is entering the lens in very shallow angles, especially near the fringe of the fisheye lenses, it gets reflected or refracted in dependence of its entrance polarization. In a simple test using a consumer omnidirectional camera, the pixel values near the fringe changed up to 33% for the same light source but rotated light polarization. Most panorama stitching approaches propagate these local color mismatches into the stereoscopic ODIs, resulting in binocular rivalry, where the same point of the scene has brightness, color or sharpness differences between the left and right eye views. Thus, our goal is to reduce these local color imbalances between stereoscopic ODI pairs V^L and V^R for both ODIs stitched and globally color matched as outlined in the previous subsections as well as ODIs stitched by state of art approaches, which will be evaluated in Section 4.

Without loss of generality, we always modify the left view V^L to match its colors to the right view V^R in the remainder of this section.

3.2.1. Local color matching

We first describe our local color matching (LCM) process for common, rectilinear stereoscopic images by utilizing optical flow and color transfer from one eye view to the other. A naïve approach would be to directly match the colors of each corresponding pixel from the optical flow estimation. However, this would produce severe artifacts due to the unavoidable imperfections in the optical flow field, with an example of artifacts shown in Fig. 10(c). Therefore, our method contains an additional filtering stage to apply color transfer locally in order to not introduce artifacts resulting from incorrect pixel correspondences.

We first estimate the optical flow $\mathbf{h} = (\mathbf{u}, \mathbf{v})^T$ between the image of the left view V^L and the image of the corresponding right view V^R as proposed in [45]. Then, we warp the right view to the left view using $\mathbf{h}$, resulting in V^R_{warped}, and compute the difference between V^L and V^R_{warped}. The resulting differences are then filtered with a Gaussian filter kernel as follows:

$$C = G_\sigma \circ (V^L - V^R_{warped}), \quad (9)$$

where C is a smooth, blurred image representing the overall local difference between left and right view. The standard deviation σ of the Gaussian filter controls the smoothness of the correction, balancing between the naïve pixel-wise color correction ($\sigma = 0$) and a kind of region-based color correction ($\sigma > 0$) by taking the influence of neighboring pixels into account in order to overcome the inaccuracy of the optical flow.

Finally, we color correct V^L by subtracting the difference C at each pixel:

$$V'^L = V^L - C. \quad (10)$$

We found out that for $\sigma = 0.5$ the introduction of additional artifacts resulting from the errors in the optical flow estimation is marginal or not visible, while the color matching between the views is substantially improved (see evaluation results in Section 4).

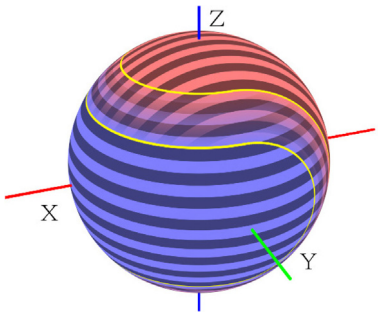


Fig. 4. Visualization of two overlapping patches on the sphere. The area between the two yellow lines is the overlapping area.

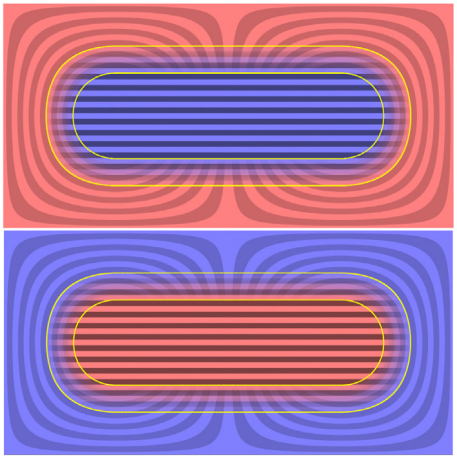


Fig. 5. Projection of two overlapping patches into the ERP format: patch P_1 centered in the ERP image (top), patch P_2 centered in the ERP image after rotation of the sphere (bottom).

3.2.2. Adaptation of LCM to spherical images

The local color correction as described in the previous subsection could be applied directly in the equirectangular format of the ODIs. However, equirectangular images are strongly non-linear near the poles. Furthermore, the continuity within a sphere is lost due to cutting and unrolling the spherical image, which would also result in missing pixel correspondences near the left and right borders in the equirectangular projection (ERP) format that may cross the cut line. Obviously the estimation of the optical flow in these extreme regions would fail. In order to circumvent these limitations, we split the spherical image into multiple patches and deal with these as if they were planar images.

We found that our color matching method is relatively insensitive to non-linearity caused by mapping a spherical patch into a plane, and that it is sufficient to split the sphere into two elongate patches P_1 and P_2 equal in their shape and size as shown in Fig. 4 and highlighted in blue and red. We then map both patches into the ERP format by rotating the sphere 90° along the forward X axis and 180° around the vertical Z axis as shown in Fig. 5. This ensures that each patch has minor distortions as it is centered around the equator.

The rotation can be expressed as a simple transformation matrix:

$$\mathbf{p}_2 = S^{-1}(\mathbf{R} \cdot S(\mathbf{p}_1)) \qquad \mathbf{R} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \tag{11}$$

where $\mathbf{p}_2$ is the new location of point $\mathbf{p}_1 = (x, y)^T$ after the sphere rotation, $S()$ represents the transformation from ERP to spherical projection and $S^{-1}()$ is the inverse transformation, respectively.

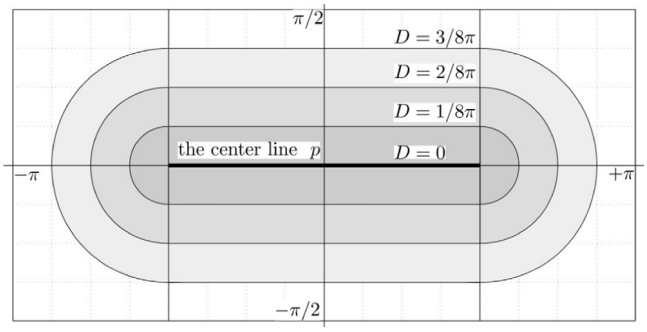


Fig. 6. Distance $D(\mathbf{p})$ from the central line defined by points $\mathbf{p}$.

When estimating the optical flow separately for each patch, slightly different values will result for scene points located around the edges of the patches. To mask any possible discontinuity when mapping back the processed patches into spherical coordinates, we widen their areas and linearly blend these overlapping patches across the edge zone as shown delimited by the yellow lines in Figs. 4 and 5.

We define the blending mask B for a patch by a Voronoi style diagram, where each cell is formed by points nearer to its center than to the center of other cells. Instead of a single point, however, we define the cell center as a line segment defined by the points $\{\mathbf{p} \mid \mathbf{p}_x \in [-\pi/2, +\pi/2],\ \mathbf{p}_y = 0\}$ with $\mathbf{p}_x$ and $\mathbf{p}_y$ as the coordinates of the points in the ERP format, as shown in Fig. 6. The distance $D(\mathbf{p})$ of a point from the line segment is calculated as follows:

$$D(\mathbf{p}) = \begin{cases} \sqrt{(|\mathbf{p}_x| - \pi/2)^2 + \mathbf{p}_y^2} & \text{if } |\mathbf{p}_x| \geq \pi/2 \\ |\mathbf{p}_y| & \text{otherwise.} \end{cases} \tag{12}$$

If $d_1(\mathbf{p}) = D(\mathbf{p})$ is the distance measure for the mapping case of patch P_1 into for the ERP format, then d_2 can be calculated for the mapping case of patch P_2 with the following equation by transforming the points $\mathbf{p}$ into spherical coordinates, apply the rotation and transforming them back into the ERP format according to Eq. (11):

$$d_2(\mathbf{p}) = D(S^{-1}(\mathbf{R} \cdot S(\mathbf{p}))) \tag{13}$$

Finally, the blending mask $B(\mathbf{p})$ for both patches in the ERP format is defined as

$$B(\mathbf{p}) = \begin{cases} 0 & \text{if } d(\mathbf{p}) < 0 \\ d(\mathbf{p}) & \text{if } d(\mathbf{p}) \in [0, 1] \\ 1 & \text{if } d(\mathbf{p}) > 1 \end{cases} \tag{14}$$

with

$$d(\mathbf{p}) = \frac{d_1(\mathbf{p}) - d_2(\mathbf{p})}{w} + \frac{1}{2}, \tag{15}$$

where w defines the width of the transition region of the blending mask with an opacity $0 < \alpha < 1$. For $w = \pi$, the transition region would have the maximum width, going from the center of the image to its borders. In our computation, we chose a value of $w = \pi/4$, as we found that it produces very smooth transition while it keeps the transition region relatively small.

Applying the color correction to the left image V_1^L and its rotated version V_2^L according to Eq. (10) by using the blend mask B, the final color corrected image V'^L can be computed with:

$$V'^L = (V_1^L - C) \cdot B + (V_2^L - C) \cdot (1 - B). \tag{16}$$

The whole process of local color matching for spherical images (LSCM) is outlined in Algorithm 2.

Algorithm 2: LSCM

Objective: To locally match colors between stereoscopic ODIs.

Input: Stereoscopic pair of ODIs in ERP format, standard deviation σ of Gaussian filter

i. Map left and right spherical ODIs into ERP format resulting in images V_x^L and V_x^R with $x \in \{1,2\}$ ($x = 2$ in the case of rotated images).

ii. For all pixels $p \in \{V_x^L, V_x^R\}$ estimate the optical flow **h** between the stereoscopic pair.

iii. Compute the color difference C according to Eq. (9).

iv. Correct V_x^L according to Eq. (10).

v. Rotate original left and right spherical ODIs according to Eq. (11) and repeat steps i. to iv.

vi. Undo rotation for $V_2'^L$.

vii. Compute the blending mask B according to Eq. (14).

viii. Create locally color matched stereopair $\{V'^L, V^R\}$ by using the blending mask B according to Eq. (16).

Output: The locally color matched left image V'^L.

(a) (b)

Fig. 7. Voronoi patch extraction in (a) spherical and (b) ERP format.

3.3. Color mismatch evaluation

In order to evaluate the results of the proposed color correction steps presented in the previous subsections, we implemented a simplified version of the method proposed by Croci et al. [8], obtained by discarding the saliency. The analysis consists of three main components: the spherical patch extraction, the color mismatch analysis between corresponding patches in the left and right stereoscopic view using color statistics, and the visualization and computation of color mismatch scores.

For the extraction of approximately equally sized patches from the ODI, first a spherical Voronoi diagram [46] is computed from evenly distributed points on the sphere. Each cell of the Voronoi diagram corresponds to a patch. Fig. 7 shows the spherical Voronoi diagram computed from 30 evenly distributed points on the sphere, and its projection into the ERP format overlaid with an input image. In the next step, each spherical patch of the ODI is mapped into the ERP format, and the pixel colors of each patch are obtained by sampling the ODI in ERP format using bilinear interpolation.

Once the patches are extracted, the disparity maps for each patch are estimated using the Semi-Global Block Matching approach described in [10] which delivers good results at reasonable computational costs. Since the disparity estimation can be noisy and inaccurate, we apply a consistency check for the disparity maps, and only disparity values which are consistent are used for further computations. If d_{L2R} and d_{R2L} are the disparity maps from left to right view, and from right to left view, then the disparity at pixel (x, y) in d_{L2R} is valid if

$$|d_{L2R}(x,y) + d_{R2L}(x - d_{L2R}(x,y), y)| \leq \delta, \tag{17}$$

where δ is a predefined threshold.

From the consistent pixels of each patch two color statistics are extracted: the mean and standard deviation of the color channels in the Lab color space similar to [27]. Assuming that μ_L and μ_R are the mean color vectors of the left and right view of the patch i, and that σ_L and σ_R are the standard deviation color vectors of the same patch, then the color mismatch score of the patch i is defined by the following equation:

$$CMS_i = \sqrt{\|\mu_L - \mu_R\|^2 + \lambda \|\sigma_L - \sigma_R\|^2} \tag{18}$$

where λ is a tuning parameter that was set to one for the generation of the results. The patch scores can also be visualized using the jet color map as shown in Fig. 8(b).

In addition to the patch scores, we also compute a global score CMS_{global} by simply averaging the patch scores:

$$CMS_{global} = \frac{\sum_{i=1}^{N} CMS_i}{N}. \tag{19}$$

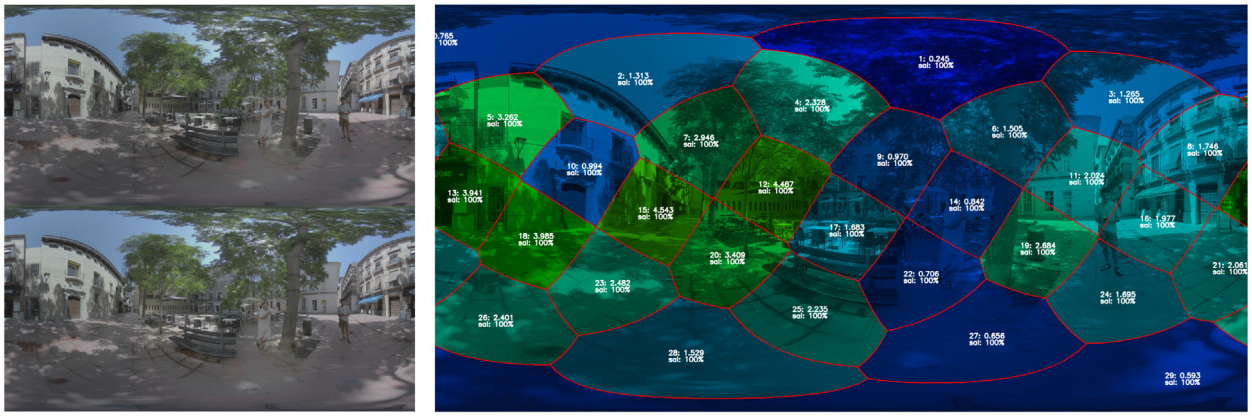
4. Evaluation

The evaluation is performed in three parts. First we evaluate the modules and the influence of different parameter settings with respect to quality and robustness in Section 4.1. In Section 4.2, we compare our entire system against current state of the art post-production tools (i.e. Foundry's Cara VR and Google Jump as introduced by Anderson et al. in [4]) which are used for professional productions using 6 video sequences for which all source views were available. Furthermore, we compare our LSCM method against the method described in Croci et al. [11] using their dataset consisting or 15 stereoscopic ODIs. Additionally, we compare our LCM method, i.e. without spherical extension, against the method described in Zhang and Liu [3] using their dataset consisting of 21 planar scenes (stitched from 2 views) and one 360-degree cylindrical panorama (stitched from 16 views). Finally, we evaluate the performance of our modules with a benchmark test on the video dataset in Section 4.3.

4.1. Evaluation of modules and parameter settings

In order to demonstrate and evaluate the performance of the different modules, we selected a stereoscopic ODI shown in Fig. 8 characterized by local color mismatch in parts of the image. We first applied our GCM approach to match the source views globally. Then, we applied the LSCM approach with different smoothness parameters, i.e. different standard deviations for the Gaussian filter kernel. For validation purposes, the color matched images were then analyzed with the independent color mismatch analysis method presented in Section 3.3.

The averages of the color mismatch patch scores are illustrated in Fig. 9. In this Figure, we can see how GCM removes most of the color mismatch, and how LSCM is able to reduce the color mismatch even further. It is worth noticing, that the smoothness parameter influences these results. By reducing this parameter, the color mismatch improves, but on the other hand, artifacts are introduced due to incorrect matches in the optical flow estimation. Fig. 10(c) shows some close-ups from the color corrected results. As can be seen in this Figure, the artifacts related to the optical flow are present only for $\sigma = 0$. We found out that for $\sigma = 0.5$ the introduction of additional artifacts resulting from the errors in the optical flow estimation is marginal or not visible, while the color matching between the views is substantially improved. Thus, in all further evaluations, we set $\sigma = 0.5$.



(a) (b)

Fig. 8. Stereoscopic ODI used for the evaluation of the proposed method: (a) Input and (b) visualization of color mismatch (green colored patches indicate color mismatch).

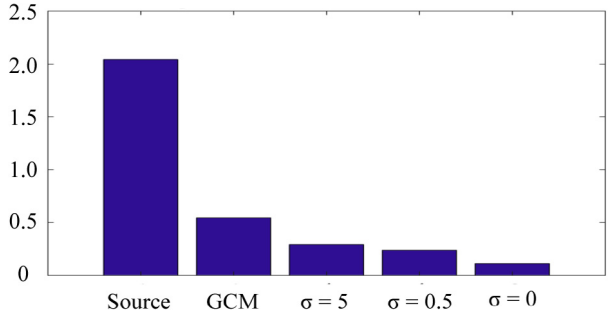
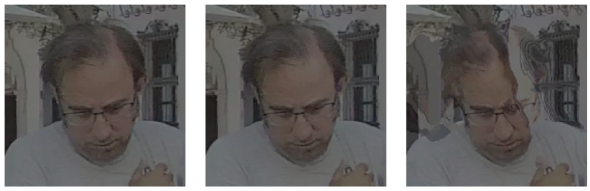


Fig. 9. Average of the color mismatch patch scores obtained with different parameters.



(a) $\sigma = 5.0$ (b) $\sigma = 0.5$ (c) $\sigma = 0.0$

Fig. 10. Visualization of optical flow artifacts with varying smoothness parameters.

4.2. Comparison with state of the art

In order to compare our entire system against the state of the art post-production tools which currently dominate the market, i.e. The Foundry's Cara VR (a proprietary algorithm based on unpublished work, but considered to be the industry reference) and Google Jump which is based on the paper of Anderson et al. [4], we first created a dataset (see Table 2) of stereoscopic ODVs from a variety of commercially available stereoscopic omnidirectional cameras, where both unstitched and stitched data were available. As we aimed for a real world application, we collected six 100 frames long samples captured with static and moving cameras to explore the stability of our system over time. ODVs 1, 3, 4 and 5 were captured with rigs using a regularly spaced ring of single cameras, while ODVs 2 and 6 were captured with rigs using parallel stereo camera pairs.

Table 3 contains the average color mismatch scores and standard deviations for all 100 frames for each ODV using 6 different methods. The results of the best method are highlighted in bold. As ODVs 1, 2 and 6 were not captured with a Google rig, we could not apply the Google's stitching method with this data. The table shows that our stitching approach with GCM and LSCM provides the best results for most of the ODVs. For ODV 1 and ODV 3, the combination of Cara VR + LSCM and Google + LSCM, respectively, provides the best color matching results.

Fig. 11 shows the color mismatch scores for all frames over time. The individual curves show a high stability over time after LSCM is applied. Only ODV 6 (see Fig. 10(i)) shows some variation over time which is especially noticeable if LSCM is not applied. This was caused by a moving camera rig which makes the stitching and global color matching of the source views more challenging. However, the LSCM not only reduces the color matching scores substantially, the scores are also becoming stable over time.

Table 4 shows the comparison between our LSCM method and Croci et al.'s method [11] based on a dataset of 15 ODIs from [11]. The ODIs of this dataset were captured with the following cameras: Vuze VR, Omnicam-360, Panocam POD 3D, and Jumpgate rig.

Croci et al.'s method in average reduces the overall color mismatch score to 23.03%, i.e. by a factor of 4.34, compared to the source, with the maximum reduction (11.25% compared to the source, i.e. a factor of 8.89) for ODI 1. On the other hand, our method reduces the overall color mismatch score to 17.24% on average, i.e. by a factor of 5.80, compared to the source, with the maximum reduction (9.02% compared to the source, i.e. a factor of 11.09) for ODI 1, respectively. The table shows that LSCM outperforms the approach of Croci et al. for all 15 ODIs under test. Finally, we compared our LCM method, i.e. without spherical extension, against the method described in Zhang and Liu [3] using their dataset consisting of 21 rectlinear scenes (stitched from 2 views) and one 360-degree cylindrical panorama (stitched from 16 views). This data set is not ideal as the stitched images are far from spherical ODIs. However, in order to demonstrate that our approach also outperforms the state of the art in color correction even for rectlinear and cylindrical panoramas, we applied our LCM method to the dataset provided in [3] and measured the color mismatch scores accordingly. The results are shown in Table 5.

Adding our method on top of Zhang's and Liu's results reduces the overall color mismatch score to 17.16% on average, i.e. by a factor of 5.8265 compared to the source, with the maximum reduction by a factor of 19.6 for scene 3, and with the least reduction by a factor of 1.49 for scene 20.

4.3. Performance evaluation

As our approach is intended to be integrated into a professional application, we run a performance evaluation on an HP Z8 workstation with the following specifications:

- CPU: 2x Intel Xeon E5-2687 W v3 @ 3.10 GHz

Table 3
Mean and standard deviation of the color mismatch scores for all frames in the sequence.

Method	ODV 1	ODV 2	ODV 3	ODV 4	ODV 5	ODV 6
Cara VR	0.2581 (± 0.0049)	2.0458 (± 0.0175)	1.2922 (± 0.0153)	1.9316 (± 0.0081)	2.0775 (± 0.0199)	1.6288 (± 0.4166)
Cara VR+LSCM	**0.1013** (± **0.0021**)	0.2611 (± 0.0048)	0.1652 (± 0.0079)	0.1958 (± 0.0065)	0.1847 (± 0.0083)	0.4486 (± 0.0350)
Google [4]	–	–	0.4003 (± 0.0023)	0.7523 (± 0.0122)	0.9823 (± 0.0375)	–
Google [4]+LSCM	–	–	**0.0990** (± **0.0043**)	0.1509 (± 0.0059)	0.2352 (± 0.0105)	–
Ours+GCM	0.4825 (± 0.0045)	0.8264 (± 0.0094)	0.4050 (± 0.0157)	0.7052 (± 0.0159)	0.6566 (± 0.0236)	1.2328 (± 0.1911)
Ours+GCM+LSCM	0.1411 (± 0.0037)	**0.1843** (± **0.0035**)	0.1331 (± 0.0052)	**0.1299** (± **0.0069**)	**0.1095** (± **0.0041**)	**0.3856** (± **0.0194**)

Table 4
Comparison of color mismatch scores between Croci et al. [11] and our LSCM method.

Method	ODI 1	ODI 2	ODI 3	ODI 4	ODI 5	ODI 6	ODI 7	ODI 8	ODI 9	ODI 10	ODI 11	ODI 12	ODI 13	ODI 14	ODI 15
Source	7.4216	4.5972	1.3804	1.5043	3.0799	3.3265	2.1033	4.5528	2.6655	2.3739	4.2286	2.2239	2.5471	2.0127	3.3212
Croci et al. [11]	0.8347	0.7242	0.5343	0.6442	0.8884	0.8864	0.5492	0.7724	0.6697	0.6211	0.9884	0.8680	0.4765	0.6663	0.7803
LSCM	**0.6693**	**0.5514**	**0.5285**	**0.3944**	**0.5224**	**0.7335**	**0.2955**	**0.5385**	**0.4760**	**0.5698**	**0.9258**	**0.7549**	**0.3467**	**0.2995**	**0.5531**

Table 5
Comparison of color mismatch scores between Zhang and Liu [3] and our LCM method.

Method	Scene 1	Scene 2	Scene 3	Scene 4	Scene 5	Scene 6	Scene 7	Scene 8	Scene 9	Scene 10	Scene 11
Zhang	1.1734	1.4560	1.2230	0.8471	1.3194	1.0212	1.6214	1.0016	1.3983	1.0532	0.8474
Zhang+LCM	**0.1361**	**0.1150**	**0.0624**	**0.0915**	**0.2680**	**0.1915**	**0.3666**	**0.2331**	**0.2271**	**0.2717**	**0.1104**
Method	Scene 12	Scene 13	Scene 14	Scene 15	Scene 16	Scene 17	Scene 18	Scene 19	Scene 20	Scene 21	Panorama
Zhang	1.7260	0.7060	1.0976	1.3273	0.8953	1.9105	2.3900	1.8384	1.8370	1.8121	1.2877
Zhang+LCM	**0.2642**	**0.4258**	**0.1773**	**0.1771**	**0.1503**	**0.4134**	**1.2939**	**1.1379**	**1.2325**	**1.0417**	**0.5820**

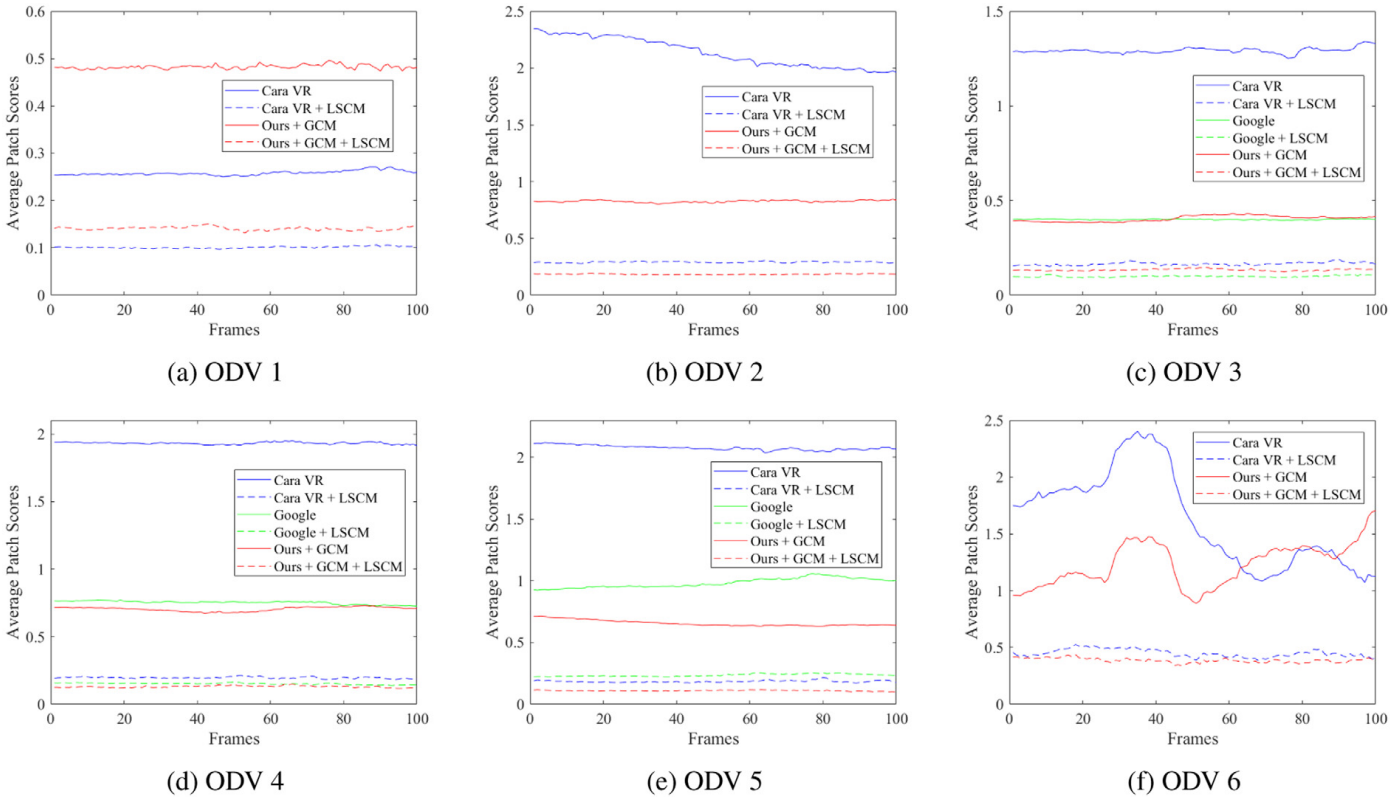


Fig. 11. Color mismatch scores for all frames of the ODVs.

- RAM: 64 GB
- GPU: NVidia Quadro P4000 with 8 GB RAM

The evaluation results for the individual modules and the entire color correction system are detailed in Table 6. The average runtime for processing a frame with stitching, GCM and LSCM is less than 1 s, whereas a large part is actually spent on the optical flow estimation.

In order to compare the processing time with Google Jump [4], we processed the slowest ODV 5 in 8 K resolution. Here, our pipeline excluding LSCM has a processing time of 953 ms per frame whereas Google Jump needs 60 s per frame on a single machine with similar color mismatch scores (see Table 3 and Fig. 11). The entire pipeline (stitching + GCM + LSCM) has a processing time of only 2.3 s per frame with much lower color mismatch scores compared to Google

Table 6
Performance evaluation of individual modules and entire color correction system (GCM includes also the stitching process).

Module/ ODV	GCM (per frame)	LSCM (per frame)	GCM + LSCM (per frame)
ODV 1	198 ms	268 ms	496 ms
ODV 2	106 ms	297 ms	398 ms
ODV 3	769 ms	645 ms	1311 ms
ODV 4	542 ms	631 ms	1195 ms
ODV 5	755 ms	643 ms	1410 ms
ODV 6	64 ms	260 ms	317 ms
Average	406 ms	457 ms	855 ms

Jump. Thus, we could demonstrate that our pipeline is quite suitable for professional post-production tasks in real-world applications.

5. Conclusion

In this paper, we introduced a novel approach and entire system for stitching and color mismatch correction and detection of S3D omnidirectional content. First, a global color matching (GCM) module was applied to video sequences where the source camera views were available in order to reduce substantial color mismatches between the different camera views.

As global color matching cannot handle scene dependent issues like lens flares and polarization issues properly, as they mainly occur locally, we introduced a spherical color correction step consisting of a local color matcher with spherical adaption (LSCM).

Then, we evaluated the system and individual modules with a patch-based color mismatch detection module [8] on three datasets (a new dataset with six sequences captured with different multi-camera rigs, a dataset with 15 ODIs introduced in [11] and a dataset with 21 rectlinear scenes and one cylindrical panorama introduced in [3]) and compared it against the state of the art. Our results showed that our color correction approaches outperformed the professional software solutions for ODVs, the method introduced by Croci et al. [11] for ODIs and the method introduced by Zhang and Liu [3] for rectlinear and cylindrical panoramas. We further demonstrated, that our LSCM method is even able to improve the results of ODVs stitched with professional tools, which currently represent the state of the art.

The proposed method was developed with the goal to integrate it into a professional workflow, where both speed and quality are of high importance. Depending on the amount of cameras and resolution of the ODVs, a processing time of less than 1 s per frame could be achieved while improving the quality significantly compared to the state of the art.

Finally, the new dataset consisting of six 360-degree sequences including the source views and stitched results using our approach and the state of the art approaches is publicly available with this paper.

Acknowledgment

Aljosa Smolic is under following grant: Science Foundation Ireland (SFI) under the Grant Number 115/RP/2776.

References

[1] S. Smith, T. Marsh, D. Duke, P. Wright, Drowning in immersion, in: Proceedings of UK-VRSIG, vol. 98, Citeseer, 1998, pp. 1–9.

[2] C. Richardt, Y. Pritch, H. Zimmer, A. Sorkine-Hornung, Megastereo: Constructing high-resolution stereo panoramas, in: Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, USA United States, 2013, pp. 1256–1263, http://dx.doi.org/10.1109/CVPR.2013.166, this work is covered by United States Patent US9398215.

[3] F. Zhang, F. Liu, Casual stereoscopic panorama stitching, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2015, pp. 2002–2010, http://dx.doi.org/10.1109/CVPR.2015.7298811.

[4] R. Anderson, D. Gallup, J.T. Barron, J. Kontkanen, N. Snavely, C. Hernández, S. Agarwal, S.M. Seitz, Jump: Virtual reality video, ACM Trans. Graph. 35 (6) (2016) 198:1–198:13, http://dx.doi.org/10.1145/2980179.2980257, URL http://doi.acm.org/10.1145/2980179.2980257.

[5] C. Schroers, J.-C. Bazin, A. Sorkine-Hornung, An omnistereoscopic video pipeline for capture and display of real-world VR, ACM Trans. Graph. 37 (3) (2018) 37:1–37:13, http://dx.doi.org/10.1145/3225150, URL http://doi.acm.org/10.1145/3225150.

[6] S. Knorr, S. Croci, A. Smolic, A modular scheme for artifact detection in stereoscopic omni-directional images, in: Proceedings of the Irish Machine Vision and Image Processing Conference, 2017.

[7] S. Knorr, K. Ide, M. Kunter, T. Sikora, The avoidance of visual discomfort and basic rules for producing "good 3D" pictures, SMPTE Motion Imaging J. 121 (7) (2012) 72–79, http://dx.doi.org/10.5594/j18236.

[8] S. Croci, S. Knorr, L. Goldmann, A. Smolic, A framework for quality control in cinematic VR based on Voronoi patches and saliency, in: 2017 International Conference on 3D Immersion, IC3D, 2017, pp. 1–8.

[9] R. Dudek, C. Cuenca-Hernández, F. Quintana-Domínguez, Stereoscopic rectification brought to practical: A method for real film production environments, in: R. Moreno-Díaz, F. Pichler, A. Quesada-Arencibia (Eds.), Computer Aided Systems Theory – EUROCAST 2017, Springer International Publishing, Cham, 2018, pp. 157–165.

[10] H. Hirschmuller, Stereo processing by semiglobal matching and mutual information, IEEE Trans. Pattern Anal. Mach. Intell. 30 (2) (2008) 328–341, http://dx.doi.org/10.1109/TPAMI.2007.1166.

[11] S. Croci, M. Grogan, S. Knorr, A. Smolic, Colour correction for stereoscopic omnidirectional images, in: Irish Machine Vision and Image Processing Conference, IMVIP 2018, 2018.

[12] D. Vatolin, A. Bokov, M. Erofeev, V. Napadovsky, Trends in S3D-movie quality evaluated on 105 films using 10 metrics, in: Proceedings of the IS&T/SPIE Electronic Imaging: Stereoscopic Displays and Applications XXVII, vol. 2016, (5) 2016, pp. 1–10.

[13] M. Lambooij, W. IJsselsteijn, M. Fortuin, I. Heynderickx, Visual discomfort and visual fatigue of stereoscopic displays: A review, J. Imaging Sci. Technol. 53 (3) (2009) 30201, http://dx.doi.org/10.2352/J.ImagingSci.Technol.2009.53.3.030201.

[14] T. Shibata, J. Kim, D.M. Hoffman, M.S. Banks, The zone of comfort: Predicting visual discomfort with stereo displays, J. Vis. 11 (8) (2011) 1–29, http://dx.doi.org/10.1167/11.8.11.Introduction, arXiv:NIHMS150003.

[15] G. Kramida, A. Varshney, Resolving the vergence-accommodation conflict in head mounted displays, IEEE Trans. Vis. Comput. Graphics 22 (7) (2015) 1–16, http://dx.doi.org/10.1109/TVCG.2015.2473855.

[16] D. Khaustova, J. Fournier, E. Wyckens, O. Le Meur, An objective method for 3D quality prediction using visual annoyance and acceptability level, in: Proceedings of the IS&T/SPIE Electronic Imaging: Stereoscopic Displays and Applications XXVI, vol. 9391, 2015, http://dx.doi.org/10.1117/12.2076949.

[17] J. Chen, J. Zhou, J. Sun, A. Bovik, Binocular mismatch induced by luminance discrepancies on stereoscopic images, in: IEEE Int. Conf. Multimed. Expo, ICME, 2014, pp. 1–6, http://dx.doi.org/10.1109/ICME.2014.6890127.

[18] Q. Dong, T. Zhou, Z. Guo, J. Xiao, A stereo camera distortion detecting method for 3DTV video quality assessment, in: IEEE Asia-Pacific Signal and Inform. Process. Assoc. Annu. Summit and Conf, 2013, pp. 1–4, http://dx.doi.org/10.1109/APSIPA.2013.6694209.

[19] A. Voronov, D. Vatolin, D. Sumin, V. Napadovsky, A. Borisov, Methodology for stereoscopic motion-picture quality assessment, in: Proc. of the SPIE, Stereoscopic Displays and Appl. XXIV, vol. 8648, 2013, http://dx.doi.org/10.1117/12.2008485.

[20] S. Croci, S. Knorr, L. Goldmann, A. Smolic, A framework for quality control in cinematic VR based on Voronoi patches and saliency, in: IEEE Int. Conf. 3D Immersion, IC3D, Brussels, Belgium, 2017.

[21] W. Xu, J. Mulligan, Performance evaluation of color correction approaches for automatic multi-view image and video stitching, in: IEEE Conf. on Comput. Vision and Pattern Recognition, CVPR, 2010, pp. 263–270, http://dx.doi.org/10.1109/CVPR.2010.5540202.

[22] C. Mouffranc, V. Nozick, Colorimetric correction for stereoscopic camera arrays, in: Comput. Vision - ACCV 2012 Workshops, 2013, pp. 206–217.

[23] Q. Wang, P. Yan, Y. Yuan, X. Li, Robust color correction in stereo vision, in: Int. Conf. Image Processing, ICIP, no. 2, 2011, pp. 965–968, http://dx.doi.org/10.1109/ICIP.2011.6116722.

[24] X. Zheng, N. Yuzhen, J. Chen, Y. Chen, Color correction for stereoscopic image based on matching and optimization, in: IEEE Int. Conf. 3D Immersion, IC3D, Brussels, 2017, http://dx.doi.org/10.1109/IC3D.2017.8251900.

[25] S. Ye, S.P. Lu, A. Munteanu, Color correction for large-baseline multiview video, Elsevier Signal Process.: Image Commun. 53 (January) (2017) 40–50, http://dx.doi.org/10.1016/j.image.2017.01.004.

[26] M. Xia, J. Yao, R. Xie, M. Zhang, Color consistency correction based on remapping optimization for image stitching, in: IEEE Int. Conf. Comput. Vision, ICCV, 2017.

[27] E. Reinhard, M. Ashikhmin, B. Gooch, P. Shirley, Color transfer between images, IEEE Comput. Graph. Appl. 21 (5) (2001) 34–41, http://dx.doi.org/10.1109/38.946629.

[28] F. Pitie, A. Kokaram, R. Dahyot, N-dimensional probability density function transfer and its application to color transfer, in: IEEE Int. Conf. Comput. Vision, ICCV, vol. 2, 2005, pp. 1434–1439, http://dx.doi.org/10.1109/ICCV.2005.166.

[29] Y.-W. Tai, J. Jia, C.-K. Tang, Local color transfer via probabilistic segmentation by expectation-maximization, in: IEEE Conf. on Comput. Vision and Pattern Recognition (CVPR), vol. 1, 2005, pp. 747–754 vol. 1, http://dx.doi.org/10.1109/CVPR.2005.215.

[30] F. Pitié, A. Kokaram, The linear Monge–Kantorovitch linear colour mapping for example-based colour transfer, in: Visual Media Prod., 2007. IETCVMP. 4th European Conf. on, 2007, pp. 1–9.

[31] M. Grogan, R. Dahyot, A. Smolic, User interaction for image recolouring using L2, in: Proc. of the 14th European Conf. on Visual Media Prod, CVMP 2017, ACM, New York, NY, USA, 2017, pp. 6:1–6:10, http://dx.doi.org/10.1145/3150165.3150171.

[32] N. Bonneel, G. Peyré, M. Cuturi, Wasserstein barycentric coordinates: Histogram regression using optimal transport, ACM Trans. Graph. 35 (4) (2016) 71:1–71:10, http://dx.doi.org/10.1145/2897824.2925918.

[33] B. Wang, Y. Yu, T.-T. Wong, C. Chen, Y.-Q. Xu, Data-driven image color theme enhancement, ACM Trans. Graph. 29 (6) (2010) 146:1–146:10, http://dx.doi.org/10.1145/1882261.1866172.

[34] Y. Shih, S. Paris, F. Durand, W.T. Freeman, Data-driven hallucination of different times of day from a single outdoor photo, ACM Trans. Graph. 32 (6) (2013) 200:1–200:11, http://dx.doi.org/10.1145/2508363.2508419.

[35] M. Grogan, R. Dahyot, Robust registration of gaussian mixtures for colour transfer, CoRR abs/1705.06091 (2017) arXiv:1705.06091.

[36] M. Grogan, M. Prasad, R. Dahyot, L2 registration for colour transfer, in: 23rd European Signal Process. Conf., EUSIPCO, 2015, pp. 1–5, http://dx.doi.org/10.1109/EUSIPCO.2015.7362799.

[37] R. Szeliski, H.-Y. Shum, Creating full view panoramic image mosaics and environment maps, in: Computer Graphics, Proceedings of Siggraph '97, 2000.

[38] M. Brown, D.G. Lowe, Automatic panoramic image stitching using invariant features, Int. J. Comput. Vis. 74 (1) (2007) 59–73, http://dx.doi.org/10.1007/s11263-006-0002-3.

[39] A.Z. Jamaluddin, C. Jiang, O. Morel, R. Seulin, D. Fofi, 3D reconstruction from specialized wide field of view camera system using unified spherical model, in: S. Battiato, G. Gallo, R. Schettini, F. Stanco (Eds.), Image Analysis and Processing - ICIAP 2017, Springer International Publishing, Cham, 2017, pp. 495–506.

[40] R. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge University Press, 2003.

[41] M.P. Mike Krainin, Google AI Blog, Seamless Google Street View Panoramas, 2017, URL https://ai.googleblog.com/2017/11/seamless-google-street-view-panoramas.html.

[42] A. Zomet, A. Levin, S. Peleg, Y. Weiss, Seamless image stitching by minimizing false edges, IEEE Trans. Image Process. 15 (2006) 969–977.

[43] ITU, Recommendation ITU-R BT.709-5: Parameter Values for the HDTV Standards for Production and International Programme Exchange, International Telecommunication Union, 2002.

[44] E. Santellani, E. Maset, Fusiello, Seamless image mosaicking via senchronization, in: ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences, 2018.

[45] R. Dudek, C. Cuenca, F. Quintana, An application of optical flow: slow motion effect on streaming image sequences, in: R. Moreno Díaz, F. Pichler, A. Quesada Arencibia (Eds.), Computer Aided Systems Theory – EUROCAST 2007, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 654–659.

[46] F. Aurenhammer, Voronoi diagrams - A survey of a fundamental data structure, ACM Comput. Surv. 23 (3) (1991) 345–405, http://dx.doi.org/10.1145/116873.116880.