Color Correction for Image-Based Modeling in the Large

Tianwei Shen, Jinglu Wang, Tian Fang*, Siyu Zhu, Long Quan

Department of Computer Science and Engineering,
Hong Kong University of Science and Technology
{tshenaa,jwangae,tianft,szhu,quan}@cse.ust.hk

Abstract. Current texture creation methods for image-based modeling suffer from color discontinuity issues due to drastically varying conditions of illumination, exposure and time during the image capturing process. This paper proposes a novel system that generates consistent textures for triangular meshes. The key to our system is a color correction framework for large-scale unordered image collections. We model the problem as a graph-structured optimization over the overlapping regions of image pairs. After reconstructing the mesh of the scene, we accurately calculate matched image regions by re-projecting images onto the mesh. Then the image collection is robustly adjusted using a non-linear least square solver over color histograms in an unsupervised fashion. Finally, a connectivity-preserving edge pruning method is introduced to accelerate the color correction process. This system is evaluated with crowdsourcing image collections containing medium-sized scenes and city-scale urban datasets. To the best of our knowledge, this system is the first consistent texturing system for image-based modeling that is capable of handling thousands of input images.

1 Introduction

The past few decades have witnessed the significant achievement of 3D reconstruction. With the help of unmanned aerial vehicles and mobile devices, we can recover the 3D structures of city-scale scenes from images with ease. Moreover, one of the advantages for image-based modeling is that the images can not only be used to recover the 3D information, but also to do texture mapping for surfaces so that the models are natural and photo-realistic. In the multi-view reconstruction setting, a set of images are captured so that they depict roughly the same object from different view points. As camera parameters, such as white balance and shutter speed, are re-calculated for each image, the same object would appear differently in images with different view angles. If the original images are used in texture mapping, the mesh may contain undesired visual artifacts, which harms the visual experience of image-based modeling.

* Tian Fang (tianft@cse.ust.hk) is the corresponding author.

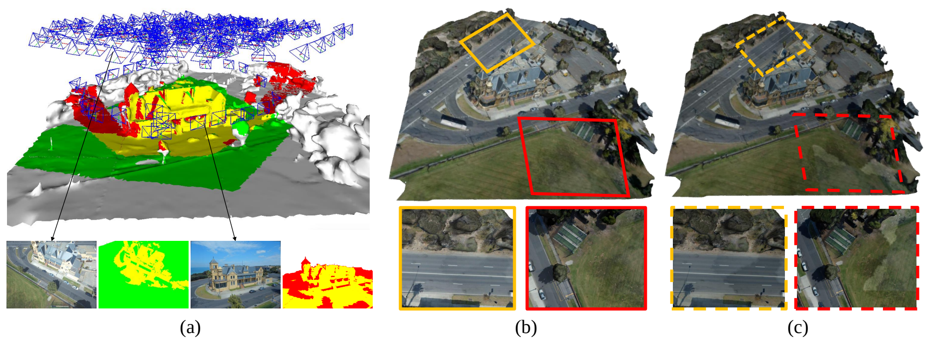

Fig. 1. Color correction for image-based modeling. (a) Color region correspondences generated by the reconstructed model. The top picture shows the registered cameras and the reconstructed mesh model. The colored regions are projected from a view pair at the bottom (using different colors to represent different views). The yellow region is the common region seen by the view pair, which provides the correspondence between the two views for color correction. (b) The resulting textured model using our color corrected images. (c) Textured model using the original images. It is evident that our method can generate much more harmonious and visually agreeable textured model.

Therefore, we need an efficient pre-processing step to fit the unordered image collection in a harmonious color tone. The objective is to use this set of corrected images to render a consistent and harmonious 3D model, in terms of fewer visual artifacts. Thus, we would like this color correction process to be based on the geometric relations and the 3D scenes within the images. As an existing technique, *color balancing* (or sometimes addressed as *color transfer* by the computer graphics community) deals with transferring the color palette of the source image to the target image, thus restricting the correction to an image pair. We would like to delve deeper by applying color correction on unordered image collections.

Color discontinuity also causes troubles for panoramic stitching and many works [1–3] have been proposed to address this issue. However, these techniques can not be directly applied to the 3D stitching problem due to the following reasons. First, panoramic stitches are usually conducted in a linear fashion. Once this linear order is decided, color adjustment for an image involves only the neighboring images, which is not the case for unordered image collections that form the graph structure with loops. Second, the images for panoramic stitching are often captured roughly around a single axis of rotation, thus the color discrepancy is less drastic, compared with multi-view stereo which involves more complicated camera motions, particularly for scenes containing non-Lambertian objects.

In this paper, we propose a method to address the color inconsistency problem in the image-based 3D reconstruction. Our method globally adjusts the color of original images based on histograms of color distributions in the overlapping regions. To get a precise overlapping region, we back-project the mesh to the images. Then the luminance component and the two chrominance components are separately balanced in a graph-structured optimization framework. We use the adjusted images for texturing and further processing.

Our method does not require a reference view, which differs from previous works such as [4]. There are generally two reasons that this property is preferred. First, we usually do not know the ground-truth reference view before doing the color adjustment. Second, if several images depict the same natural scene, we consider illumination variance in different images as noise and would like the color correction process to manifest the authentic color.

2 Related Work

2.1 2-View Color Transfer

Many previous works have addressed the problem of adjusting the color tone of a source image to a target image, commonly known as color transfer. These various color transfer techniques can be generally categorized using two criterions, namely *parametric* and *non-parametric* with regard to *global* and *local*. For large-scale unordered image collections, color transfer within an image pair serves as the basic building block which can be fitted in a large-scale optimization framework. To avoid computational burden for 3D reconstruction, a global model-based method is preferred as the basic unit in color correction of image collections. Global color correction ensures that the changes made to the local regions does not create visual artifacts to the textured model, while parametric methods facilitates the combination of color transfer units.

Reinhard et al. [5] first propose a popular color transfer method based on simple statistics of color distributions in images. To get rid of the undesired correlations between different channels in RBG, their method operates in the $l\alpha\beta$ color space, which has little correlation between different axes for many natural scenes [6]. Tian et al. [7] apply color correction to panoramic imaging, by matching the histograms of the overlapping region of two images. This method is still limited to an image pair thus should be categorized to 2-view methods. Recently, Hwang et al. [8] proposed a non-linear color transfer method based on matched correspondences. Each RGB color is adjusted to the target value by an affine transformation, using a process called moving least squares. Nguyen et al. [9] propose a new method which takes into consideration the scene illumination and color gamut. After running white-balancing on both the source and target images, they transform the luminance values of the target image using Xiao et al.'s gradient preserving matching technique [10]. For a detail comparison of several methods mentioned here, readers may refer to the quantitative evaluation work of Xu et al. [11].

2.2 Ordered N-View Color Correction

Nanda and Cutler's work [12] discusses several important issues for real-time panoramas generated by omnidirectional cameras, including auto brightness correction, auto white balance correction, vignetting correction, etc. Brown et al. [1]

propose the first automatic panoramic image stitching pipeline. To get a consistent color in the panoramic image, they simply tweak the gain (a linear factor) of intensity mean and apply multi-band blending [13] to further process visible seams. Xiong et al. [3] proposes a color correction method for sequences of overlapping images, which is in particular useful for panorama stitching. To deal with pixel overflow problem (pixels may be saturated during the color correction processing), they use YCbCr color space and separately apply gamma correction for the luminance component and linear correction for the chrominance components. However, this method is restricted to sequences of images and is not suitable in the large-scale unordered settings. The latter problem is much harder than tasks such as panorama stitching since there are much more constraints between pairs of images in a graphical style, and the color consistency is with respect to the whole 3D scenes, in different view points. Yamamoto and Oi's work [14] describe a color correction method for multi-view video embedded in an energy minimization scheme. This modeless non-parametric approach requires a manual decision on a reference view and a sequential order of neighboring cameras.

2.3 Other Approaches

To generate a mesh with a consistent color tone, other methods are also extensively used. Moulon et al. [4] propose a global method based on histogram quantiles of overlapping regions. This method uses VLD filter [15] to extend the corresponding region of an image pair. While in our method, we take into consideration the mesh to generate accurate corresponding regions, which particularly benefits the consistent texture mapping. Waechter et al. [16] propose a texturing system which corrects the mesh color on a per-triangle basis. Allène et al. [17] employ a multi-band texture blending technique to adjust the color of overlapping texture regions. However, texture blending has its own limitations and is not able to cope with the drastic changes in different lighting conditions, which has been come to realize by recent literature [11]. Though we do not perform multi-band texture blending explicitly, our approach is orthogonal to it and in practice we can combine these two approaches to deliver the best result.

Perhaps the most similar approach to our method is [18]. They also exploit shared color properties over an image collection. While in our work, we use a more precise way to generate correspondences through mesh re-projection. Our method uniquely combines image collection editing with consistent mesh texturing. Meanwhile, we are able to handle large-scale image collections containing thousands of images.

3 Our Method

3.1 Overview

Our method builds atop the 3D reconstruction pipeline, which is assumed readily available since our goal is to generate high-quality triangular meshes with

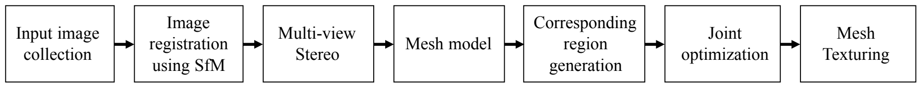

Fig. 2. The workflow of our method. Our method builds upon the 3D reconstruction pipeline, which consists of *Structure-from-Motion*, *dense reconstruction* and *surface reconstruction*. After *Joint optimization*, we can obtain the corrected and harmonious image collection. The color corrected image collection is further used for consistent texturing.

no visual seams. Here we give a brief overview of the state-of-the-art image-based modeling pipeline: given an unordered image collection $\mathcal{I} = \{I_i\}$ taken under different exposure and illumination conditions, camera poses and a sparse 3D point cloud is reconstructed using Structure-from-Motion(SfM) [19, 20]. The scene geometry is then reconstructed using state-of-the-art multi-view stereo techniques [21, 22]. The dense stereo points are further processed and triangulated to render a high-quality triangular mesh [23]. Here the image collection $\mathcal{I} = \{I_i\}$ is registered in the same coordinate as the mesh, with all the geometric relations such as camera poses $\mathcal{P} = \{P_i\}$ already known, thus enabling the interaction of original images and textures. The color correction method is based on the statistics of histograms thus it is robust to outliers in camera parameters and the mesh.

For all image pairs with enough overlapping, we compute the common corresponding regions by back-projecting the mesh $\mathcal{M}$, which contains a set of triangular faces $\mathcal{F} = \{f_i\}$, to render two masks for a pair of overlapped images. Previous methods [5, 14, 18] without the access to geometry information usually generate this color correspondences using full frames or SIFT features. For mesh texturing, our method enlarges the corresponding region and well satisfies the purpose of texture editing. These corresponding regions impose constraints on the camera network in a graphical representation. These constraints form a joint optimization which aims to align the histograms of the common regions in an image pair. We use a parametric model for the color adjustment of each image and apply the color correction for each channel in YCbCr color space. The workflow of the system is shown in Figure 2. In the following sections, we describe different components of our system in details.

3.2 Color Region Correspondences

For each image I_i that is successfully registered into the scene geometry, there is a set of triangles $\mathcal{T}_i = \{f_1^{(i)}, f_2^{(i)}, \ldots, f_{n_i}^{(i)}\}$ that can be seen by this camera. Normally a face f_i is seen by several cameras, which is the origin of visual seems since the images used for texturing are captured under non-uniform conditions. Rather than tackling this problem in the texture space on the basis of per triangle, which indirectly solves the problem but is computationally intensive, we would like to pre-process the images to set them in a uniform color tone. We begin by re-projecting the mesh to cameras and obtain the visible triangle set $\mathcal{T}_i$. To determine the visibility of each triangle, we first render a depth map by re-projecting each triangle to the view. Each pixel value of the depth map saves the id of the nearest triangle seen by view. If a pair of images I_i and I_j share enough visibility overlapping, which is measured by the cardinality of $\mathcal{T}_i \cap \mathcal{T}_j$, the color of the common regions in the two images should be roughly the same. Suppose we denote the projection from 3D space to image I_i as Π_i, we can represent the common regions by a pair of 0-1 binary masks:

$$M_{ij} = \sum_{t \in \mathcal{T}_i \cap \mathcal{T}_j} \Pi_i(t), \quad M_{ji} = \sum_{t \in \mathcal{T}_i \cap \mathcal{T}_j} \Pi_j(t) \tag{1}$$

The two masks select the accurate corresponding regions which should possess the same color distribution. To robustly match the two regions without violating the smoothness of image, we propose to measure the discrepancy of the color histograms, denoted as $D(H(I_i * M_{ij}), H(I_j * M_{ji}))$. Here $H(I_i * M_{ij})$ is the histogram of the color distribution in image I_i selected by mask M_{ij}. To facilitate the implementation and make this method scalable to large-scale datasets, we adopt statistical measures of the color histogram, instead of matching image color pixel by pixel. Namely, we define $D(H(I_i * M_{ij}), H(I_j * M_{ji}))$ to be the sum of pixel values that correspond to the same quantiles in the cumulative distribution of color histograms.

3.3 Global Optimization of Color Distribution

We use a global transformation model to parameterize the adjustment of the color histogram. Particularly, we solve the following optimization problem

$$\begin{aligned} \underset{\{s_i\},\{o_i\}}{\text{minimize}} \quad & \sum_{i,j,k} \rho\Big(\frac{(s_i Q_{ij}^{(k)} + o_i) - (s_j Q_{ji}^{(k)} + o_j)}{s_i + s_j}\Big)^2 \\ \text{subject to} \quad & 1 - \delta_s \le s_i \le 1 + \delta_s, -\delta_o \le o_i \le \delta_o, \ \forall i. \end{aligned} \tag{2}$$

Where $Q_{ij}^{(k)}$ is the value of one color channel which corresponds to the k-th quantile of the cumulative distribution function for the overlapping region. Simply minimizing the numerator of the objective function, $(s_i Q_{ij}^{(k)} + o_i) - (s_j Q_{ji}^{(k)} + o_j)$, would result in a set of trivial solution in which the scale factors $\{s_i\}$ are all zeros and offset factors $\{o_i\}$ are unbounded. Therefore, we cancel out this shrinkage effect by normalizing the absolute error by the scale factors $s_i + s_j$ We also set lower bounds and upper bounds for scale and offset parameters, which ensures that the corrected color does not deviate too much from the original color. Equation 2 is a form of non-linear lest square problem with bounded constraints, where ρ is the loss function used to deal with outliers. Specifically, we use the Pseudo-Huber loss which writes as

$$\rho(x) = \delta^2(\sqrt{1 + (x/\delta)^2} - 1) \tag{3}$$

Equation 3(with $\delta = 1$) is a smooth approximation of Huber loss function which is extensively used in robust estimation. The smooth approximation version ensures the derivatives are continuous for all degrees. The optimization 2 can be solved using Levenberg-Marquardt algorithm [24, 25].

Image captured under the same scene usually possess stable chromatic characteristics, while differ in luminance. On the other hand, images are often stored in RGB space, which does not separate luminance component from chromatic ones. Moreover, there are correlations between different channels of RGB color space [8, 26]. Arbitrary modification to RGB channels independently would lead to out-of-gamut error. Hence we apply the color correction in YCbCr color space, in which the luminance component is separated out and the primary colors are possessed into perceptually meaningful information. The transformation from analog RGB to YCbCr can be expressed by a linear mapping:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.168736 & -0.331264 & 0.5 \\ 0.5 & -0.418688 & -0.081312 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix} \tag{4}$$

The optimization (2) is applied to three YCbCr channels independently. We have also observed that in natural scene images, the two chrominance components usually span a limited spectrum while the luminance Y channel is distributed over the whole range. Therefore, after the global color adjustment for the whole image collection, some images may be subjected to underflow or oversaturation. We remap the luminance range using a set of linear factors with respect to all the images. Suppose that $Q'^{(i)}_{\alpha}$ denotes the α-percentile of the luminance histogram of the i-th image, which may be above 255 or under 0, the 5-percentile and 95-percentile luminance values are mapped to the lower limit and the upper limit correspondingly.

$$\begin{aligned} 0 &= S_{\mathrm{g}} \max\nolimits_i \{Q'^{(i)}_{0.05}\} + O_g \\ 255 &= S_{\mathrm{g}} \max\nolimits_i \{Q'^{(i)}_{0.95}\} + O_g \end{aligned} \tag{5}$$

In the end the luminance value is adjusted by the combination of these two transformations

$$l' = S_g(s_i l + o_i) + O_g \tag{6}$$

The rescaling induced by S_g changes the absolute color value difference in a homogenous way, hence it does not affect the uniformness of the optimization result. The YCbCr value are then converted back to RGB color space, which completes the color correction in the image space.

3.4 Graph Compression

As the number of images increases, color balancing becomes a complicated optimization problem on a large-scale graph structure, which is hard to solve in a reasonable time. We apply two methods to tackle the efficiency issue. First, we use the input camera graph given by SfM, thus restricting the problem size to be the same as the complexity of the scene. Second, a connectivity-preserving edge pruning algorithm is applied on the scene graph to further speedup the optimization process.

Graph simplification by edge pruning The goal of graph simplification is two-folded: First, we would like to prune unimportant and redundant edge links so as to accelerate the color correction process while preserving the general structure of the scene. Second, we would like to rearrange the edge distribution such that the color tone does not lean toward the densely-connected and over-sampled regions.

To avoid defeating the purpose of accelerating color correction process, we design a simple connectivity-preserving edge pruning (CPEP) method similar to the spirit of [27]. The input is the scene graph $G = (\mathcal{V}, \mathcal{E})$ computed after the mask generation. Each element in $v \in \mathcal{V}$ corresponds to an image I_i in the unordered image collection $\mathcal{I}$. We define the overlapping ratio between a pair of images as $OR_{ij}(I_i) = \frac{|\{\Delta|\Delta \in I_i \cap \Delta \in I_j\}|}{|\{\Delta|\Delta \in I_i\}|}$ and $OR_{ij}(I_j) = \frac{|\{\Delta|\Delta \in I_i \cap \Delta \in I_j\}|}{|\{\Delta|\Delta \in I_j\}|}$, where $|\{\Delta|\Delta \in I_i\}|$ and $|\{\Delta|\Delta \in I_j\}|$, $|\{\Delta|\Delta \in I_i \cap \Delta \in I_j\}|$ represent respectively the number of mesh triangles seen by image I_i, by image I_j and by both I_i and I_j. Two nodes are connected if the edge weight e_{ij}, defined as the overlapping ratio $OR_{ij} = \sqrt{OR_{ij}(I_i) \cdot OR_{ij}(I_j)}$ between a pair of images, is greater than a threshold δ_{or}, for which we set as 0.25 throughout the experiments. This algorithm first sorts edges by weights and then simply prunes the weakest edges iteratively while ensuring the connectivity of the whole graph (Algorithm 1). It is assumed that the input scene graph is connected, otherwise we can first extract the largest connected component of G and then apply the graph simplification on the largest connected component.

In the extreme case, the simplest graph that preserves global connectivity is its spanning tree with $|V| - 1$ edges. A simplification ratio γ_{sim} controls the fraction of edges to be pruned. CPEP algorithm prunes $N_{rm} = \gamma_{\text{sim}}(|\mathcal{E}| - (|\mathcal{V}| - 1))$ edges. We set γ_{sim} to be 0.9, thus keeping one tenth of original edges to lower the complexity of the graph by an order of magnitude. The primary computational cost lies in testing whether the removal of an edge can disconnect the graph (lines 6-9 in Algorithm 1). This operation takes $O(|\mathcal{V}|)$ time since we can compute the number of connected components using depth-first-search on the graph with the testing edge removed. Together with the cost of sorting edges ($O(|\mathcal{E}| \log |\mathcal{E}|)$), the total computational complexity of the algorithm is $O(|\mathcal{E}| \log |\mathcal{E}| + N_{rm}|\mathcal{V}|) = O(|\mathcal{E}|(\log |\mathcal{E}| + |\mathcal{V}|))$.

4 Graph Streaming on Large-Scale Datasets

The computation of mask pairs can be conducted in either streaming or parallel fashions. In both cases, I/O is the major bottleneck since the back-projection operation involve reading image pairs and marking common regions. To speedup this process it is ideal to read a batch of images that covers a possibly large

Algorithm 1 Connectivity-Preserving Edge Pruning (CPEP)

```
Require: The connected undirected scene graph $G = (\mathcal{V}, \mathcal{E})$, simplification ratio $\gamma_{\mathrm{sim}}$
Ensure: A connected subgraph $G' = (\mathcal{V}, \mathcal{E}')$ of $G$
 1: Sort $\mathcal{E}$ by edge weights in ascending order
 2: $N_{\mathrm{rm}} \leftarrow \gamma_{\mathrm{sim}}(|\mathcal{E}| - (|\mathcal{V}| - 1))$
 3: $\mathcal{E}' \leftarrow \mathcal{E}$
 4: $i \leftarrow 0, j \leftarrow 0$
 5: while $i < N_{\mathrm{rm}}$ do
 6:     {test whether $e_j$ is a bridge}
 7:     $\mathcal{E}' \leftarrow \mathcal{E}' \setminus e_j$
 8:     Compute the number of connected components of $G'$
 9:     if $G'$ has more than one connected component then
10:         $\mathcal{E}' \leftarrow \mathcal{E}' \cup e_j$
11:     else
12:         $i \leftarrow i + 1$
13:     end if
14:     $j \leftarrow j + 1$
15: end while
        return $G' = (\mathcal{V}, \mathcal{E}')$
```

number of jobs that share the common image resources. Therefore, we design a batch processing paradigm to accelerate the mask pair computation.

Since each mask computation job depends on a pair of images, these jobs and down-sampled image resources constitute a bipartite graph which we called *Job-Resource-Depend-Graph (JRDG)*. For each batch iteration, maximum number of jobs and their dependent resources are loaded up to the memory limit. The order of loaded jobs can be randomly sequential or based on heuristics from JRDG. For example, we can load the images in the decreasing order of the degree of resource nodes. The pairwise mask computation is conducted after a batch loading and the memory for image resources is released for the next iteration. The histogram correspondence data for one iteration is streaming to the hard disk in order to hold as many images as possible in the memory.

When the problem scale becomes large, removing unimportant edges by checking connectivity is cumbersome and computationally expensive. To strike the balance between efficiency and perfect consistency, we take the simplest of Algorithm 1 for large-scale datasets (>10k images), in which γ_{sim} equals 1. The benefit is that we can just extract the maximum spanning tree of the scene graph without going through every weak edges, which greatly accelerates the graph simplification process. To solve large-scale color optimization in a reasonable time, we also made the approximation by using the conjugate gradients solver on the normal equations involved in the Levenberg-Marquardt algorithm. This results in an inexact step variant [28] of the Levenberg-Marquardt algorithm.

Table 1. Computation time for different stages. Statistics shown from left to right: dataset name, number of images, number of mask pairs, number of mask pairs after graph simplification, time of computing the correspondence masks, time of optimization on the simplified graph, time of optimization on the full graph, number of iterations for loading the image collection (section 4). Small-scale experiments (less than 1000 images) were running on a multi-core PC with Intel(R) Core(TM) i7-4770K processors and 32GB main memory, while large large-scale experiments were running on a server with Intel(R) Xeon(R) E5-2630 v3 CPU and 128GB main memory.

Dataset	#images	#mask pairs	#sim pairs	mask computation time (s)	graph sim opt time (s)	graph full opt time (s)	#batch
tunnel	370	14523	1755	95.9	48.7	372.0	1
hotel	237	11320	1345	102.4	20.8	159.5	1
monasterio	110	927	190	11.9	0.9	3.2	1
castle	200	9243	1104	173.4	27.0	196.1	1
city*	36480	4655k	36479	6.4 hours	25	N/A	2

5 Experiments

5.1 Implementation Details

In this section, we describe the implementation details as well as specific model parameters. We focus on the details of the color correction engine and omit techniques involving 3D reconstruction. For the later part readers may refer to a series of literatures and open-source implementations, such as [29, 30] for Structure-from-motion, [31, 32] for dense reconstruction and TexRecon [16] for texturing.

We solve equation 2 using Ceres solver [33], which implements Levenberg-Marquardt algorithm [24, 25] to solve the non-linear least-square problem with bounded constraints. We choose a set of different bounded constraints for the Y channel and for Cb, Cr channels, since the luminance component usually span the whole value range while the two chrominance components are concentrated on a narrow scope. We set $\delta_s = 0.4, \delta_o = 30$ for the luminance channel and $\delta_s = 0.2, \delta_o = 5$ for the chrominance channels. The number of quantiles k is fixed to be 10 across all the experiments.

5.2 Comparison Results

Small-scale benchmark dataset We first demonstrate the color correction performance of our method on a synthetic dataset. To show that the proposed method can work well under drastic color tone and illumination variations, we deliberately apply color balance adjustments for the *Herz* dataset, which is consist of 8 images and widely is used in multi-view stereo benchmarks [34]. The first six images in *Herz* are each set to the extreme of one color balance axis. The color tones of them are each dominated by red, yellow, blue, green, cyan, magenta. The seventh image is modified by changing the color contrast and the last image is fixed unchanged. We show original images, modified images and the color correction results in Figure 3. We compare our method with Moulon et al. [4], with the unchanged image as the reference view. The performance on this small-scale dataset is comparable though their method needs a pre-specified reference

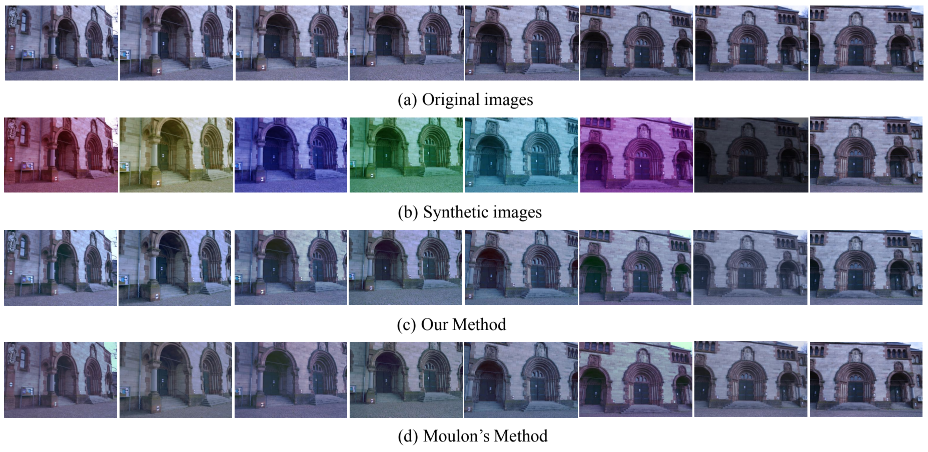

Fig. 3. Color correction result for a synthetic dataset with intense color changes. (a) the original image sequence; (b) the synthetic image sequence, modified by tuning color balance in Adobe Photoshop; (c) the result of our automatic color correction pipeline; (d) Moulon's method [4] using the unchanged image as the reference view.

view. When applying on medium-sized datasets containing hundreds of images (less than 300), however, it took hours for [4] to converge because of its L-∞ formulation, while our method took less than three minutes.

Texturing performance While the drastic color change in the above case is rarely seen in the real world, we further evaluate our method on several natural scene datasets, namely *tunnel*, *monasterio*, *castle*, and *hotel*. To the best of our knowledge, there is no previous work that demonstrates the result of color correction for an image collection on the level of mesh texturing, thus we only compare our consistent texturing results with the commonly-used texturing pipeline. Namely, the texturing pipeline can be viewed as a triangle labelling process that considers fragment quality and color discrepancy between bordering images in a MRF-based energy function [17, 35]. No texture blending technique is used for all the experiments, since we would like to evaluate the consistency of the corrected image collection.

Figure 4 illustrates the work flow of our method on the *tunnel* dataset. This dataset contains 370 images taken under drastic illumination changes. The input images are first robustly registered into a common coordinate frame using SfM. Then the multi-view stereo techniques reconstruct the high-quality surface model. If we use original images to texture the model, it greatly harms the realism of the scene and yields annoying seams. Although texture blending and photo-consistency checking may partially solve this problem, these techniques are often computationally expensive and perform poorly if the input datasets are not balanced in the image space in the first place. Our method uses the corrected images for texturing and achieves a smooth and uniform color tone

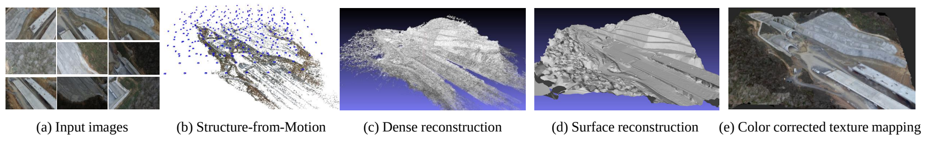
(a) Input images
(b) Structure-from-Motion
(c) Dense reconstruction
(d) Surface reconstruction
(e) Color corrected texture mapping

Fig. 4. The work flow of our method shown by the *tunnel* dataset.

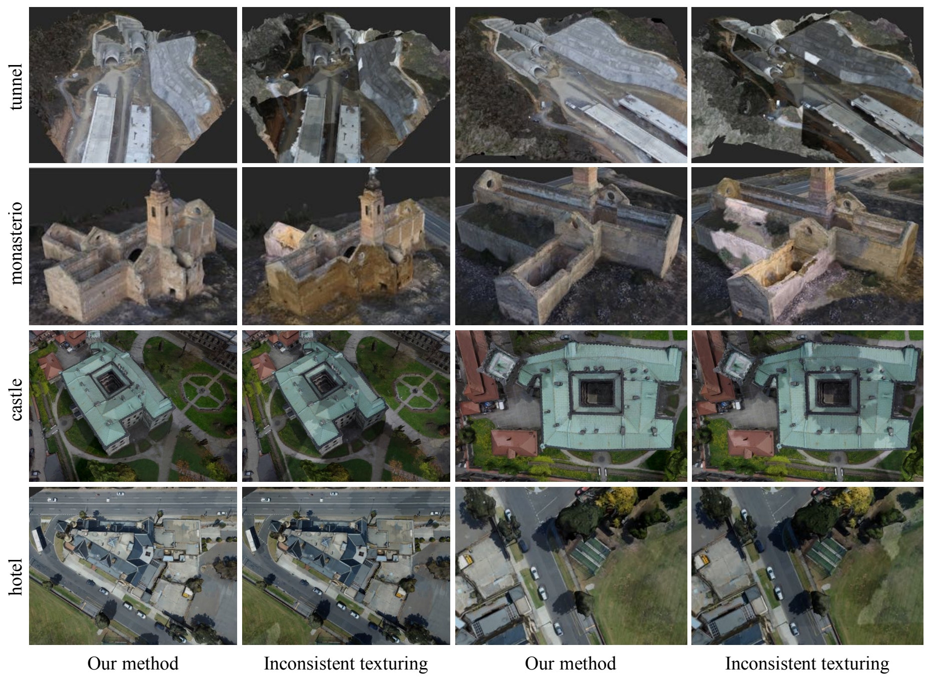

Fig. 5. Comparison of our consistent texturing and inconsistent texturing for small datasets. From top to bottom: *tunnel*, *monasterio*, *castle*, *hotel*.

without visual artifacts. The color correction process finishes in less than three minutes for small-sized datasets in the aforementioned settings. The computation time of different stages is showed in Table 1. The graph simplification operation greatly accelerates the optimization solver without introducing much overhead.

The second row of Figure 5 shows the comparison result of the *monasterio* dataset which contains 110 images captured in different days and different hours. The visual seams disappear when the corrected images are used for texturing. Other comparison results for small-scale datasets are also shown in Figure 5.

We demonstrate the performance of our method on large-scale image collections using the *city* dataset, which contains 36480 aerial images under illumination variations and occlusions. Figure 6 shows results of our method on the large-scale *city* dataset. The main computation is spent on the mask generation because of the heavy image I/O burden. Much simplification (spanning tree,

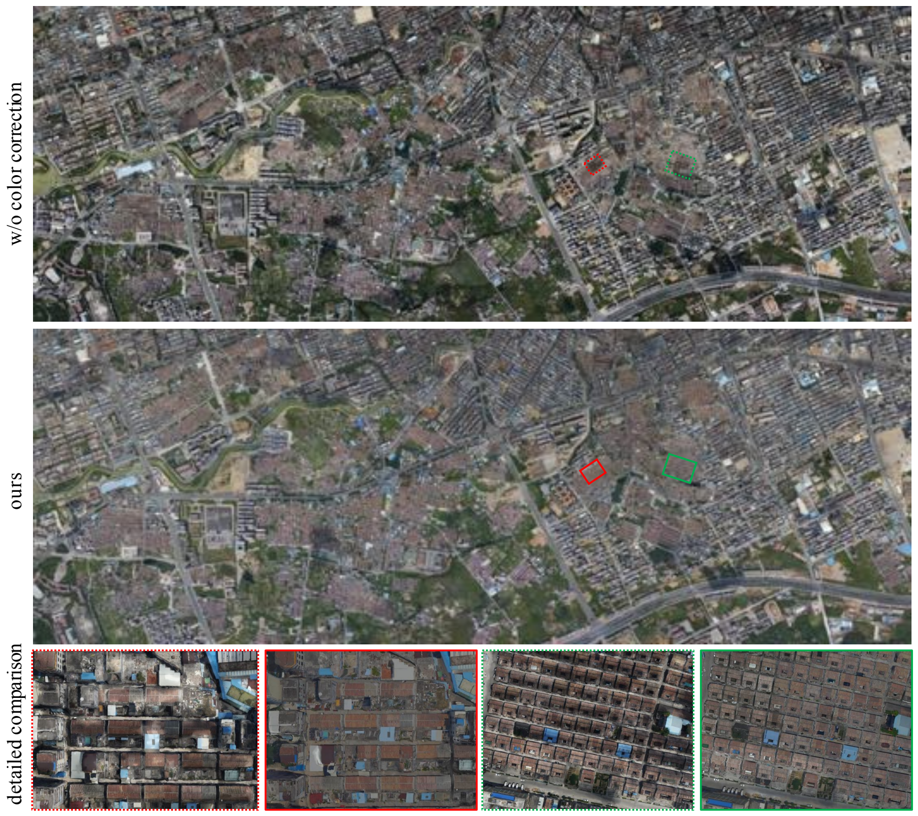

Fig. 6. Large-scale color correction on urban dataset. From top to bottom: normal texturing without color correction; refined texturing using color corrected images; two pairs of detailed texture comparison.

inexact step) is employed on the solver side to make the computation feasible, as described in Section 4. Though a less accurate solver is applied on this large-scale problem, the addition of color correction improved the texture uniformness significantly. For more high-resolution comparison results, readers may refer to the supplementary materials.

6 Conclusions and Discussions

In this paper, we present a global method to harmonize the color of an image collection using the scene geometry. Since this method relies on the mesh re-projection onto the original images, it is particularly useful for generating high-quality textured meshes without visual seams. In addition, our method works smoothly on large-scale datasets and imposes low computational burden for the 3D reconstruction pipeline. Moreover, our method can also be used to elevate the user experience for the display and exploration of large-scale image collections, such as the Photosynth system [29].

We have implicitly assumed that the intrinsic color of the overlapping region is view-independent, therefore for scenes with non-Lambertian objects the accuracy of our method will be affected. In fact, the modeling of non-Lambertian objects poses challenges for other aspects of multi-view reconstruction as well. Avenues for future work include incorporating other techniques such as highlight removal [36] to tackle the issues with reflective objects.

Acknowledgement. The authors would like to thank all the anonymous reviewers for their constructive feedbacks. This work is supported by Hong Kong RGC 16208614, T22-603/15N, Hong Kong ITC PSKL12EG02, and China 973 program, 2012CB316300.

References

1. Brown, M., Lowe, D.G.: Automatic panoramic image stitching using invariant features. International journal of computer vision (2007) 59–73
2. Eden, A., Uyttendaele, M., Szeliski, R.: Seamless image stitching of scenes with large motions and exposure differences. In: Computer Vision and Pattern Recognition (CVPR). (2006) 2498–2505
3. Xiong, Y., Pulli, K.: Color matching of image sequences with combined gamma and linear corrections. In: International Conference on ACM Multimedia. (2010) 261–270
4. Moulon, P., Duisit, B., Monasse, P.: Global multiple-view color consistency. In: Conference on Visual Media Production (CVMP). (2013)
5. Reinhard, E., Ashikhmin, M., Gooch, B., Shirley, P.: Color transfer between images. IEEE Computer graphics and applications (2001) 34–41
6. Ruderman, D.L., Cronin, T.W., Chiao, C.C.: Statistics of cone responses to natural images: Implications for visual coding. JOSA A (1998) 2036–2045
7. Tian, G.Y., Gledhill, D., Taylor, D., Clarke, D.: Colour correction for panoramic imaging. In: International Conference on Information Visualisation. (2002) 483–488
8. Hwang, Y., Lee, J.Y., Kweon, I.S., Kim, S.J.: Color transfer using probabilistic moving least squares. In: Computer Vision and Pattern Recognition (CVPR). (2014) 3342–3349
9. Nguyen, R., Kim, S., Brown, M.: Illuminant aware gamut-based color transfer. In: Computer Graphics Forum. Number 7 (2014) 319–328
10. Xiao, X., Ma, L.: Gradient-preserving color transfer. In: Computer Graphics Forum. Number 7 (2009) 1879–1886
11. Xu, W., Mulligan, J.: Performance evaluation of color correction approaches for automatic multi-view image and video stitching. In: Computer Vision and Pattern Recognition (CVPR). (2010) 263–270
12. Nanda, H., Cutler, R.: Practical calibrations for a real-time digital omnidirectional camera. CVPR Technical Sketch (2001)
13. Burt, P.J., Adelson, E.H.: A multiresolution spline with application to image mosaics. ACM Transactions on Graphics (TOG) (1983) 217–236

14. Yamamoto, K., Oi, R.: Color correction for multi-view video using energy minimization of view networks. International Journal of Automation and Computing (2008) 234–245
15. Liu, Z., Marlet, R.: Virtual line descriptor and semi-local matching method for reliable feature correspondence. In: British Machine Vision Conference (BMVC). (2012)
16. Waechter, M., Moehrle, N., Goesele, M.: Let there be color! large-scale texturing of 3d reconstructions. In: European Conference on Computer Vision (ECCV). Springer (2014) 836–850
17. Allène, C., Pons, J.P., Keriven, R.: Seamless image-based texture atlases using multi-band blending. In: International Conference on Pattern Recognition (ICPR). (2008) 1–4
18. HaCohen, Y., Shechtman, E., Goldman, D.B., Lischinski, D.: Optimizing color consistency in photo collections. ACM Transactions on Graphics (TOG) (2013) 38
19. Agarwal, S., Snavely, N., Simon, I., Seitz, S.M., Szeliski, R.: Building rome in a day. In: International Conference on Computer Vision (ICCV). (2009) 72–79
20. Shen, T., Zhu, S., Fang, T., Zhang, R., Long, Q.: Graph-based consistent matching for structure-from-motion. In: European Conference on Computer Vision (ECCV). (2016)
21. Goesele, M., Snavely, N., Curless, B., Hoppe, H., Seitz, S.M.: Multi-view stereo for community photo collections. In: International Conference on Computer Vision (ICCV). (2007) 1–8
22. Furukawa, Y., Curless, B., Seitz, S.M., Szeliski, R.: Towards internet-scale multi-view stereo. In: Computer Vision and Pattern Recognition (CVPR). (2010) 1434–1441
23. Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: Eurographics symposium on Geometry processing. (2006)
24. Levenberg, K.: A method for the solution of certain non–linear problems in least squares. (1944)
25. Marquardt, D.W.: An algorithm for least-squares estimation of nonlinear parameters. Journal of the society for Industrial and Applied Mathematics (1963) 431–441
26. Zhang, M., Georganas, N.D.: Fast color correction using principal regions mapping in different color spaces. Real-Time Imaging (2004) 23–30
27. Zhou, F., Mahler, S., Toivonen, H.: Simplification of networks by edge pruning. In: Bisociative Knowledge Discovery. Springer (2012) 179–198
28. Wright, S., Holt, J.N.: An inexact levenberg-marquardt method for large sparse nonlinear least squres. The Journal of the Australian Mathematical Society. Series B. Applied Mathematics (1985) 387–403
29. Snavely, N., Seitz, S.M., Szeliski, R.: Modeling the world from internet photo collections. International Journal of Computer Vision (2008) 189–210
30. Moulon, P., Monasse, P., Marlet, R.: Global fusion of relative motions for robust, accurate and scalable structure from motion. In: International Conference on Computer Vision (ICCV). (2013) 3248–3255
31. Furukawa, Y., Ponce, J.: Accurate, dense, and robust multiview stereopsis. Pattern Analysis and Machine Intelligence (PAMI) (2010) 1362–1376
32. Lhuillier, M., Quan, L.: A quasi-dense approach to surface reconstruction from uncalibrated images. Pattern Analysis and Machine Intelligence (PAMI) (2005) 418–433
33. Agarwal, S., Mierle, K., Others: Ceres solver. (http://ceres-solver.org)

34. Strecha, C., von Hansen, W., Gool, L.V., Fua, P., Thoennessen, U.: On benchmarking camera calibration and multi-view stereo for high resolution imagery. In: Computer Vision and Pattern Recognition (CVPR). (2008) 1–8
35. Lempitsky, V., Ivanov, D.: Seamless mosaicing of image-based texture maps. In: Computer Vision and Pattern Recognition (CVPR). (2007) 1–6
36. Tan, P., Lin, S., Quan, L., Shum, H.Y.: Highlight removal by illumination-constrained inpainting. In: International Conference on Computer Vision (ICCV). (2003) 164–169