Liu B, Xu K, Martin RR. Static scene illumination estimation from videos with applications. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 32(3): 430–442 May 2017. DOI 10.1007/s11390-017-1734-y

Static Scene Illumination Estimation from Videos with Applications

Bin Liu [1], Kun Xu [1,*], *Member, CCF, ACM, IEEE*, and Ralph R. Martin [2], *Member, ACM, IEEE*

[1] *Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China*
[2] *School of Computer Science and Informatics, Cardiff University, Cardiff CF24 3AA, U.K.*

E-mail: bin-liu13@mails.tsinghua.edu.cn; xukun@tsinghua.edu.cn; MartinRR@cardiff.ac.uk

Received December 18, 2016; revised April 6, 2017.

Abstract We present a system that automatically recovers scene geometry and illumination from a video, providing a basis for various applications. Previous image based illumination estimation methods require either user interaction or external information in the form of a database. We adopt structure-from-motion and multi-view stereo for initial scene reconstruction, and then estimate an environment map represented by spherical harmonics (as these perform better than other bases). We also demonstrate several video editing applications that exploit the recovered geometry and illumination, including object insertion (e.g., for augmented reality), shadow detection, and video relighting.

Keywords video processing, augmented reality, illumination recovery

1 Introduction

Video editing has become a popular research topic in computer graphics, due to the proliferation of handheld cameras and smart phones: video content is now very easy to generate. Many studies treat a video as a set of two-dimensional (2D) images, perhaps with temporal continuity. However, many video editing tasks require an understanding of the underlying scene in three-dimensional (3D) space. For example, a common requirement in film making, advertising and augmented reality is to add a virtual object to a video.

The difficulty of manipulating a video as a 3D scene comes from a lack of information. Explicit scene geometry, illumination and surface reflection characteristics are not recorded in videos. While current commercial video editing software such as Adobe After Effects provides many tools to help the user describe the scene geometry and illumination for subsequent editing, inputting such information is very tedious and unreliable. Even an experienced artist needs much effort to produce convincing editing results.

In particular, understanding the illumination is critical for creating proper shadows in the edited results. Any inconsistency in shadows can soon reveal the video to be a forgery[1]. Modeling scene illumination in a way which is compatible with its geometry and shadows is not easy, particularly when the illumination may not be simply a point light or directional light. Previous work has sought to estimate scene illumination with the aid of user interaction[2], by using a data driven approach[3], or by directly measuring it with a precise tool[4]. Here, we propose a system that can automatically infer the scene illumination for a given video without extra input. The scene structure is initially recovered using existing multi-view stereo methods. We then estimate shading intensity in the video by using intrinsic decomposition techniques to approximately obtain the distribution of illuminated and shadowed pixels in each frame. An illumination estimation module uses these shading maps and the scene geometry to compute an environment map represented in a spherical harmonic basis. The resulting estimated illumination and scene structure provide a basis for producing con-

Regular Paper
Special Section of CVM 2017
This work was supported by the National Natural Science Foundation of China (NSFC) and the Israel Science Foundation (ISF), Joint NSFC-ISF Research Program under Grant No. 61561146393, the National Natural Science Foundation of China under Grant No. 61521002, a research grant from the Beijing Higher Institution Engineering Research Center, and the Tsinghua-Tencent Joint Laboratory for Internet Innovation Technology.

*Corresponding Author

vincing results in a range of editing applications.

Our method works for a video of a static scene under constant (but possibly complex) illumination, taken by a moving camera. We require the input video to capture a sufficient range of views of the scene for geometric reconstruction. We assume scene surfaces to be made of Lambertian materials; we do not model specular reflection or inter-reflections between surfaces in the scene. Later we will see that, despite these assumptions, we can produce plausible results for real scenes containing real surface materials.

To summarize, our main contributions are as follows.

• We present an overall framework that combines geometric reconstruction, intrinsic decomposition, and an optimization-based approach to recover scene illumination.

• We demonstrate a set of applications that edit videos by exploiting scene geometry and illumination, producing convincing results for tasks such as object insertion[5] (e.g., for augmented reality), shadow detection and video relighting.

The remainder of this paper is organized as follows. In Section 2 we briefly review related work. Section 3 details our scene illumination estimation framework. Section 4 presents editing applications exploiting the recovered scene geometry and illumination. Section 5 presents some experiments and Section 6 draws some conclusions.

2 Related Work

In video editing, knowledge of illumination, as well as the geometric and reflectance properties of objects in the scene, is essential to achieving realistic results. Many approaches have been devised to recover this information from input images or videos. Intrinsic image decomposition algorithms aim to separate a single image into illumination and reflectance layers[6-9]. The illumination component can help determine the extent to which each pixel is in shadow, while the reflectance component reveals how the material there reflects incident light. Such a decomposition allows editing effects such as material replacement to be applied to a source image or video. Other intrinsic decomposition methods take image collections or videos of the same scene as input; matches between images add extra constraints on the relationships between reflectance values, permitting more accurate results[10-14].

However, approaches based on pixel-wise illumination and reflectance maps are not powerful enough to support more complex editing operations such as object insertion. To achieve plausible results, shadows and inter-reflections must be carefully computed, which requires an understanding of the geometry of the scene and the lighting configuration in 3D space. The problem of estimating illumination from images is called inverse lighting. Dong *et al.* proposed a method that can simultaneously recover spatially varying isotropic surface reflectance and the incident lighting given a video of a rotating object[15]. However, the geometry of the object must be known in advance, and the geometric model needs to be precisely registered to the image frames. Other methods recover the illumination distribution in a scene from shadows cast by objects of known shape[16-17]. Research has shown that humans are poor at distinguishing between different lighting conditions[18]; some work exploited this property to create fake illumination that is consistent with the scene image[19-20]. The results are physically wrong but visually plausible, which suffices to create realistic lighting effects.

The geometry of a scene in the image can be approximated by user interaction[2,21-24], inferred by scene understanding algorithms[25-26] or using depth sensors[27]. Many algorithms can directly estimate the geometry of the scene from a single image using very simple structures[28-30]. Structure-from-motion (SFM) is an image-based modeling technique that simultaneously estimates 3D scene structure, camera pose, and calibration parameters from a 2D image sequence[31]. It lies at the core of various applications in video processing such as depth inferencing[32-33], video stabilization[34], SLAM[35], and so on. For example, Snavely *et al.* developed a photo browser which takes unstructured collections of photos of tourist sites as input and computes the viewpoint of each photo as well as a sparse 3D point cloud of the scene[36], enabling the user to explore the photos in 3D space. Similarly, "Building Rome in a Day" showed how to reconstruct an entire city by processing an extremely large number of photos[37]. Fuhrmann *et al.*[38] provided an end-to-end image-based geometry reconstruction tool for object modeling; it takes photos of a scene as input and produces a textured surface as the result[38].

Rendering virtual objects into real scenes has long been studied; Kronander *et al.* provided a survey[39]. The methods used solve the problems of illumination and geometry recovery in various ways. Debevec measures scene radiance and global illumination to support object insertion, using a mirrored ball to capture a

high-dynamic range lighting environment at the location of the inserted object[4]; a differential rendering approach is used to add shadows and reflections caused by the inserted object to the original scene. Karsch *et al.* devised an image composition system that renders synthetic objects into input photos. The approximate scene structure and area light sources are determined from user annotations[2]. This work was later extended to automatically infer depths and lighting using a data driven approach[3]. Depth values are transferred from existing annotations in a database of RGBD images. A light classifier is used to detect light sources in the image and image-based lights are used to represent illumination from outside the view frustum. Kholgade *et al.* modeled the objects in a photograph using stock 3D models from a public repository as guidance, representing the illumination in a von Mises-Fisher basis[40]. The user may apply complex manipulations to the objects in the image, including scaling, rotation, translation, copying-and-pasting, and non-rigid deformation. These methods work on a single image and require extra input (from user interaction, large image sets, or a 3D model database) to obtain geometric information which is essential for plausible illumination estimation. In contrast, our system works on video, and the geometry of the scene is automatically recovered without additional input.

Other work has also considered inserting objects into video. Zhang *et al.* estimated a depth map as discrete layers for each frame[41]. The object in another video clip is also treated as lying on a plane and is inserted between two layers of the source video; shadows are synthesized according to a user specified light direction. Our method differs from [41] in that we treat the scene and objects as 3D models, and create physically correct shadows using realistic rendering methods.

3 Pipeline

Our approach is applicable to videos of a static scene shot by a moving camera under (approximately) constant lighting. The pipeline of our system is illustrated in Fig.1. Given an input video, firstly the geometry of the scene, the camera pose and the camera parameters are recovered, using scene reconstruction methods. We also apply an intrinsic decomposition algorithm to each frame, obtaining shading and reflectance components. The illumination of the scene is then estimated based on the geometry and the shading. The recovered information is passed to downstream editing applications enabling such tasks as object insertion and video relighting.

3.1 Scene Reconstruction

Recovery of scene geometry and camera pose is the first step of our system. We directly use existing approaches without modification. But as they are the key to plausible results, we briefly outline how they are used in our 3-step process. Firstly, structure from motion (SFM) recovers a sparse point cloud of the scene geometry as well as the camera's extrinsic parameters (i.e., rotation and translation in world coordinates) and intrinsic parameters (i.e., focal length and principal

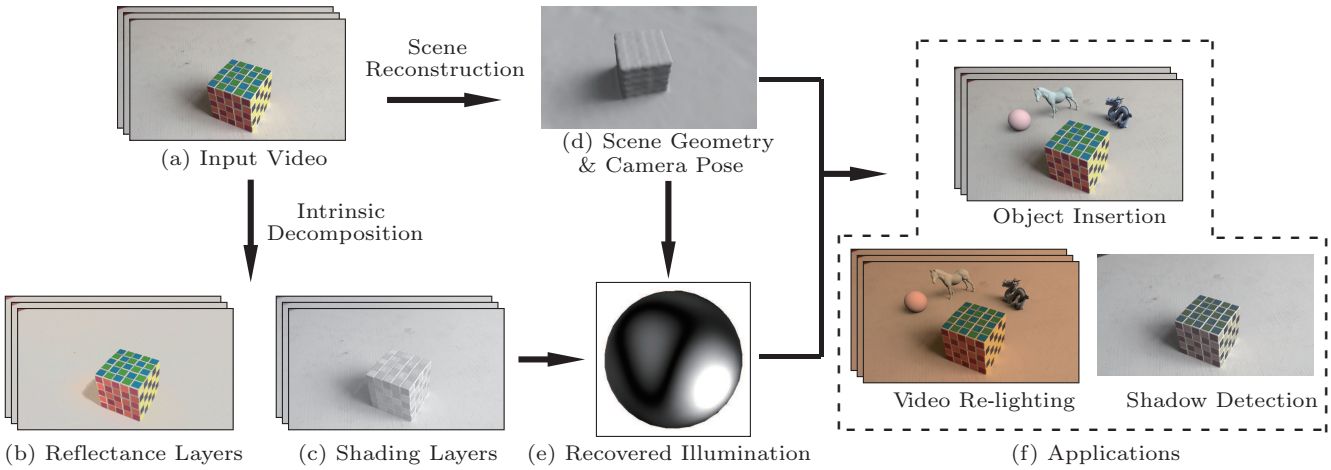
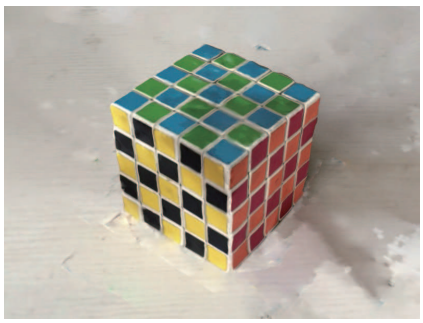


Fig.1. Pipeline. Given (a) an input video, for each frame, (b) the shading and (c) the reflectance are computed using intrinsic video decomposition. (d) The scene geometry and the camera pose are simultaneously recovered using scene reconstruction methods. (e) The illumination of the scene is then estimated from the shading maps and the scene geometry. The recovered information provides a basis for editing (f) applications such as object insertion, shadow detection, and video relighting.

point). Next, multi-view stereo computes a dense point cloud for the scene based on the SFM results. Finally, surface reconstruction estimates a collection of mesh surfaces that best explain the dense point cloud. Various tools are available that can perform these tasks, e.g., LS-ACTS[42], and MVE[38]. We use VisualSFM[43] and OpenMVS[44] for scene reconstruction. Various reconstructed scene point clouds, camera poses and mesh surfaces are shown in Fig.2.

(a)



(b)

Fig.2. Scene reconstruction results for a video of a Rubik's cube. (a) Sparse point cloud from SFM. (b) Final textured scene mesh produced by multi-view stereo and surface reconstruction.

Often the algorithm needs no assistance. Nevertheless, in cases with very complex scenes or insufficient views of some parts of the scene, it may fail to produce perfect surface meshes. We will discuss such cases in detail in Subsection 5.3, but we note here that all is not lost — the user can use 3D modeling software like MeshLab[45] to repair such problems before proceeding.

Once the scene geometry is ready, we use it as an input to our illumination estimation algorithm, as described next.

3.2 Illumination Estimation

In the standard rendering equation proposed by Kajiya[46], the light L_r reflected from point $\boldsymbol{x}$ in direction $\boldsymbol{\omega}_r$ is given by:

$$L_r(\boldsymbol{x}, \boldsymbol{\omega}_r) = \int_{\Omega} f_r(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r) L_i(\boldsymbol{x}, \boldsymbol{\omega}_i)(\boldsymbol{\omega}_i \cdot \boldsymbol{n}) V(\boldsymbol{x}, \boldsymbol{\omega}_i) \mathrm{d}\boldsymbol{\omega}_i, \quad (1)$$

where L_i is the incident light arriving at $\boldsymbol{x}$ from direction $\boldsymbol{\omega}_i$, $\boldsymbol{n}$ is the surface normal at point $\boldsymbol{x}$, f_r is the bidirectional reflectance distribution function (BRDF) of the surface at $\boldsymbol{x}$, and V is the visibility term, indicating whether the light from other objects or light sources reaches $\boldsymbol{x}$ along $\boldsymbol{\omega}_i$. If we assume that the material is Lambertian, then f_r is only determined by the albedo of the surface at point $\boldsymbol{x}$. We assume that interreflections between the surfaces are negligible compared with the light received from the light sources, following other work in this area[3,40]. In this case (1) can be written:

$$L_r(\boldsymbol{x}) = f_r(\boldsymbol{x}) \int_{\Omega} L_i(\boldsymbol{x}, \boldsymbol{\omega}_i)(\boldsymbol{\omega}_i \cdot \boldsymbol{n}) V(\boldsymbol{x}, \boldsymbol{\omega}_i) \mathrm{d}\boldsymbol{\omega}_i. \quad (2)$$

Note that L_r no longer depends on $\boldsymbol{\omega}_r$: it does not appear on the right hand side of the equation. Setting f_r to 1 in (2) allows the received light at point $\boldsymbol{x}$ to be measured:

$$S(\boldsymbol{x}) = \int_{\Omega} L_i(\boldsymbol{x}, \boldsymbol{\omega}_i)(\boldsymbol{\omega}_i \cdot \boldsymbol{n}) V(\boldsymbol{x}, \boldsymbol{\omega}_i) \mathrm{d}\boldsymbol{\omega}_i. \quad (3)$$

Now, the shading layer of the video, which we denote as S_v, encodes the incident illumination at each point of the scene and can be estimated by applying an intrinsic decomposition algorithm[14] to it. The goal of illumination estimation is to determine L_i such that it produces a distribution of $S(\boldsymbol{x})$ as close as possible to S_v given the scene geometry and camera parameters. Thus, the following shading error should be minimized:

$$E_s = \sum_{t,p} \left(S(\boldsymbol{x}(t,p)) - S_v(t,p) \right)^2, \quad (4)$$

where t is the frame number, p is pixel position in the image plane, and $\boldsymbol{x}(t,p)$ un-projects pixel p to the scene surface in the camera configuration of frame t.

The illumination term L_i can take a variety of forms, such as point lighting, area lighting, directional lighting, spot lighting or an environment map, the last being well-suited to representing "natural" illumination[47]. An environment map is a distribution on a sphere showing the light intensity from each direction. It can be represented as a linear combination of basis functions, such as spherical harmonics[48], Haar wavelets[49] or von Mises-Fisher kernels[50]. We may write L_i as a weighted sum in some basis $\{l_k\}$:

$$L_i = \sum_k \alpha_k l_k. \quad (5)$$

Substituting (5) into (3) gives

$$\begin{aligned} S(\boldsymbol{x}) &= \sum_k \alpha_k \int_\Omega l_k(\boldsymbol{x}, \boldsymbol{\omega}_i)(\boldsymbol{\omega}_i \cdot \boldsymbol{n}) V(\boldsymbol{x}, \boldsymbol{\omega}_i) \mathrm{d}\boldsymbol{\omega}_i \\ &= \sum_k \alpha_k s_k(\boldsymbol{x}), \end{aligned} \tag{6}$$

where $s_k(\boldsymbol{x})$ is the individual contribution of basis l_k to the overall shading value $S(\boldsymbol{x})$. Substituting (6) into the objective function in (4) gives a quadratic form:

$$\begin{aligned} & E_s \\ &= \boldsymbol{\alpha}^{\mathrm{T}}(\sum_{t,p} \boldsymbol{s}_{t,p}\boldsymbol{s}_{t,p}^{\mathrm{T}})\boldsymbol{\alpha} - 2(\sum_{t,p} S_v(t,p)\boldsymbol{s}_{t,p}^{\mathrm{T}})\boldsymbol{\alpha} + \\ & \sum_{t,p} S_v(t,p)^2 \\ &= \boldsymbol{\alpha}^{\mathrm{T}} A \boldsymbol{\alpha} - 2\boldsymbol{b}^{\mathrm{T}}\boldsymbol{\alpha} + c, \end{aligned} \tag{7}$$

where $\boldsymbol{\alpha}$ and $\boldsymbol{s}_{t,p}$ are vectors whose k-th components are α_k and $s_k(\boldsymbol{x}(t,p))$ respectively. Having chosen a set of basis functions $\{l_k\}$, the weights of each component can be computed by minimizing the objective function in (7).

We use the spherical harmonic (SH) basis in our implementation for its efficiency in modeling low-frequency lighting[48]; 256 basis functions suffice in our application. Fig.3 illustrates the intensity distribution provided by some of these basis elements, and their shading effects; note that the coefficient vector $\boldsymbol{\alpha}$ of SH does not have to be non-negative as a single basis contains both positive and negative intensities. Matrix $\boldsymbol{A}$ in (7) is semi-positive definite, and thus the minimum of E_s is not unique if $\boldsymbol{A}$ is singular. To avoid this possibility, we add a regularization term E_r. The final objective function is:

$$E = E_s + E_r = \boldsymbol{\alpha}^{\mathrm{T}}(\boldsymbol{A} + \boldsymbol{\Lambda})\boldsymbol{\alpha} - 2\boldsymbol{b}^{\mathrm{T}}\boldsymbol{\alpha} + c, \tag{8}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix with positive entries, used to avoid singularities; it is set to an identity matrix in our implementation. The quadratic function E has a unique global minimum, which is the solution of the linear equation:

$$(\boldsymbol{A} + \boldsymbol{\Lambda})\boldsymbol{\alpha} = \boldsymbol{b}.$$

After obtaining the coefficients $\{\alpha_k\}$ for the basis functions, the environment map L_i can be generated from (5). Fig.4 illustrates the shading produced by such a computed environment map. The shadow cast by the Rubik's cube indicates that the light comes from the right hand side, as correctly estimated by our algorithm.

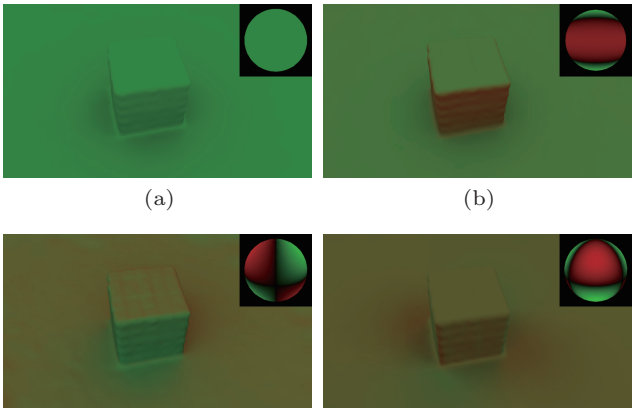


Fig.3. Sample spherical harmonic basis functions (indexed by l, m) and their shading effects. Both positive and negative intensities exist in a single SH basis, visualized here in green and red. (a) $l = 0, m = 0$. (b) $l = 2, m = 0$. (c) $l = 2, m = 1$. (d) $l = 3, m = 2$.

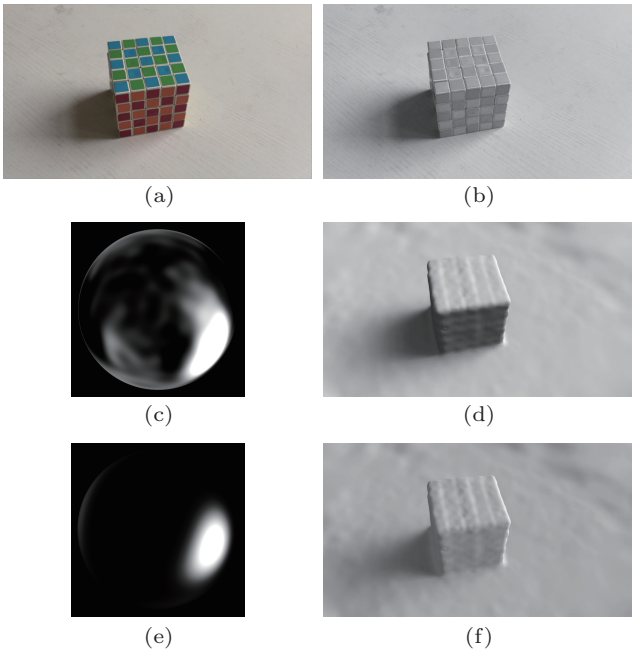


Fig.4. Illumination recovery. (a) Input video frame. (b) Shading layer from intrinsic decomposition. (c)(e) Environment maps estimated by our method and the one in [40] respectively. (d) (f) Rendered recovered scene geometry under the corresponding illumination.

4 Applications

4.1 Object Insertion

The purpose of recovering scene geometry and illumination from the video is to enable video editing with visually realistic results. One such application is to insert objects into the video. If we put a synthetic object into the scene, some pixels of the scene will be

occluded by the virtual object. The virtual object can also have an influence on unoccluded pixels. For example, the local area around the inserted object will darken, as the object casts shadows on it. This area is called a "local scene" by Debevec[4], who renders it with a differential approach which we now briefly review. Firstly, the virtual local scene is rendered with and without the synthetic object under the given illumination and camera parameters, giving images I_O and I_N. The difference between these two images reveals how the inserted object affects the scene. The change is added to the original video frame I_F to generate the composed frame I_Δ:

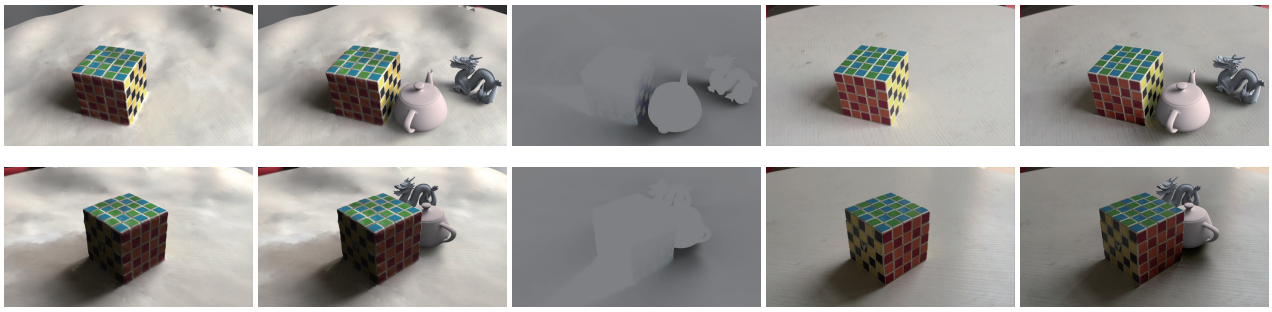
$$I_\Delta = I_F + (I_O - I_N). \tag{9}$$

Differential rendering requires the virtual object to be rendered in a virtual local scene with a locally similar color to that of the real scene. However, this does not always work well for video, as the intensity of the local scene may change from frame to frame, e.g., due to auto-exposure correction by the camera (see Fig.5), the intensity of the table changes. Adding the same difference to each frame can lead to inconsistent shadows. To overcome this problem, we compute the intensity in the composed frame using:

$$I_C = I_F \times (S_O/S_N), \tag{10}$$

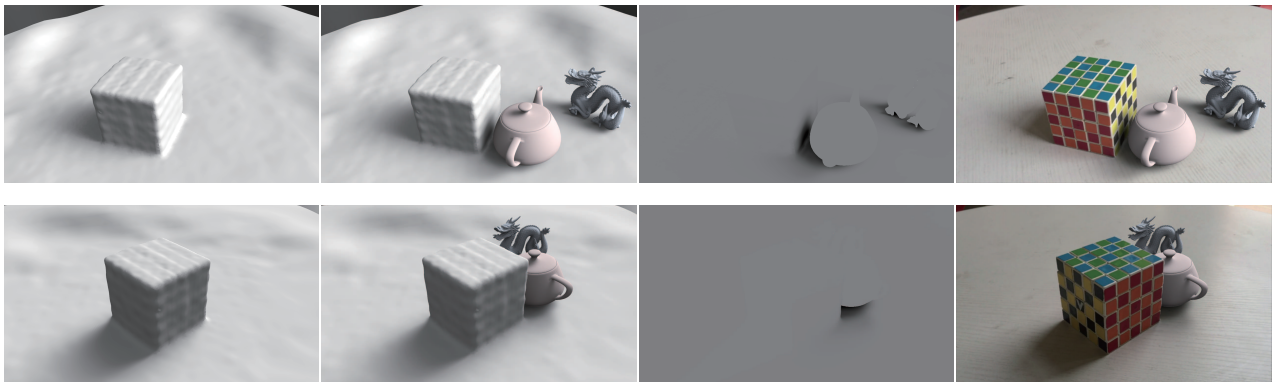
where S_O and S_N are the rendered textureless scenes with and without the virtual object respectively. Their ratio reveals how much pixel darkening is caused by inserting the virtual object (see Fig.6). In Subsection 5.1, we will give a qualitative and quantitative comparison between our proposed object insertion method and differential rendering.

If a region in the original scene is occluded by the inserted object, we should replace it by pixels from the



(a) (b) (c) (d) (e)

Fig.5. Differential rendering by Debevec[4]. (a) and (b) show the rendered result for the textured scene with and without inserted objects (i.e., I_N and I_O) respectively. (c) shows their difference $I_O - I_N$. This difference is added to the source frame I_F in (d), providing the composite results I_Δ in (e). Due to changes in intensity of the surface, inconsistencies may arise in shadows.



(a) (b) (c) (d)

Fig.6. Compositing method. (a) and (b) show the original and edited scenes (i.e., S_N and S_O) respectively. The decrease in received light caused by object insertion at each point is measured by S_O/S_N in (c). Each pixel in the source frame is darkened according to this proportion, generating the composited result I_C in (d).

synthetic object. We determine such regions by comparing the depths of scene pixels and object pixels in the camera coordinate system; the pixel in I_O is copied to I_C if the inserted object is nearer to the viewer. Note that other approaches such as the one in [2] lack scene geometry, requiring the occluded region to be manually determined by the user. Synthetic frames showing examples of object insertion can be seen in Fig.7.

Some frames of the input video can be blurred due to camera motion. To produce more convincing results, the synthetic object should also be blurred as determined by the camera movement. When compositing frame t, we gather several samples in the time domain, at times $t' = t - \Delta t$ with $\Delta t \in [0, 1]$. The final composite frame $I_{C,t}$ is obtained by averaging the individually composited images at sample times t':

$$I_{C,t} = \frac{\sum_{t'} (t - t') \left(M_{t'} I_{O,t'} + (1 - M_{t'}) I_{F,t} (S_{O,t'} / S_{N,t'})\right)}{\sum_{t'} (t - t')},$$

where $I_{O,t'}$, $S_{O,t'}$ and $S_{N,t'}$ have similar meanings to I_O, S_O and S_N in (9) and (10), rendered at time t'. The camera pose at t' is computed by linearly interpolating the rotation and translation of the camera between frames t and $t - 1$. $I_{F,t}$ is the t-th frame of the input video. $M_{t'}$ is a binary mask with value 1 where the inserted object occludes the original scene at time t' and 0 elsewhere. Fig.8 shows some composited frames to which a bench has been added, with and without the use of motion blur. With motion blur, we perceive more coherent movement of the synthetic object and the background scene, strengthening the realism of the result.

4.2 Shadow Detection

Shadow detection and removal are challenging computer vision tasks[51]. If the light source is blocked by various objects, the shadows cast can undesirably darken other shapes. If we can compute the extent to which the light is blocked, we can remove the shadow. This is possible if scene geometry and illumination are available. To determine how much the light is blocked at each point, we need to compute the amount of light it receives when there is no occlusion, by ignoring the visibility term V in (3):

(a)

(b)

Fig.7. Object insertion example. (a) Garden. Synthetic objects are bench, dragon and ram. (b) Gallery. Synthetic objects are Buddha, Lucy and chair.

$$S'(\boldsymbol{x}) = \int_{\Omega} L_i(\boldsymbol{x}, \boldsymbol{\omega}_i)(\boldsymbol{\omega}_i \cdot \boldsymbol{n})\, \mathrm{d}\boldsymbol{\omega}_i.$$

As $S(\boldsymbol{x})$ and $S'(\boldsymbol{x})$ measure the light that reaches $\boldsymbol{x}$ with and without occlusion respectively, $S(\boldsymbol{x})/S'(\boldsymbol{x})$ indicates the degree to which the light is unobstructed at $\boldsymbol{x}$ (see Fig.9). In this example, there are two places which appear darker: 1) the front of the Rubik's cube, as it is facing away from the light sources, and 2) the table to the left of the Rubik's cube, where light is blocked by the cube. Only the latter constitutes a shadow. Using the shading layer from the intrinsic decomposition alone, we cannot distinguish these two cases. However, we can distinguish shadows of the kind in the second case by computing $S(\boldsymbol{x})/S'(\boldsymbol{x})$.

(a) (b)

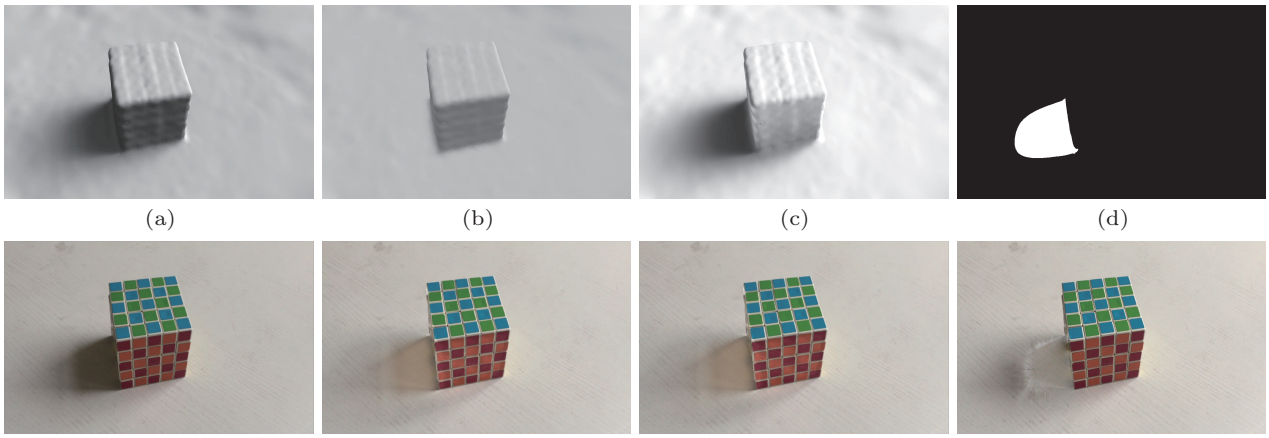
Fig.8. Synthesised frames with an added bench. (a) Without motion blur. (b) With motion blur.

To perform shadow removal, we divide each pixel $I(\boldsymbol{x})$ by the value of $S(\boldsymbol{x})/S'(\boldsymbol{x})$ to compensate for shadowing. Doing so directly is insufficient, as this value may be inaccurate and some areas may be overcorrected while other areas are still too dark. To overcome this problem, we threshold $S(\boldsymbol{x})/S'(\boldsymbol{x})$ to generate a shadow mask and employ local illumination changes[52] in the masked region to adjust its appearance, as shown in Fig.9. Although being imperfect, our shadow removal results are comparable to those generated by state-of-the-art automatic[53] and interactive[54] algorithms. An alternative would be to use the shadow mask to remove part of the original image, and to apply image completion methods or related techniques to replace the removed shadowed region. Such approaches remain for future work.

4.3 Video Relighting

Movie studios often set up the lighting carefully before filming as everyday environments typically do not provide appropriate lighting. However, not all film makers have access to the substantial resources required, and even with careful planning, things may not always turn out as desired. Thus, an alternative is to improve videos after shooting by using software to synthetically relight them. Relighting is still a challenging problem, studied in various works[55-56]. As our approach recovers scene geometry and illumination, we can relight video by editing the estimated environment map. For example, we can change the color temperature of the light, change the environment map or add additional light sources.

Fig.10 shows an example of relighting a scene. Given the rendered scene geometry with original and new environment maps, the relit frame can be computed from (10).



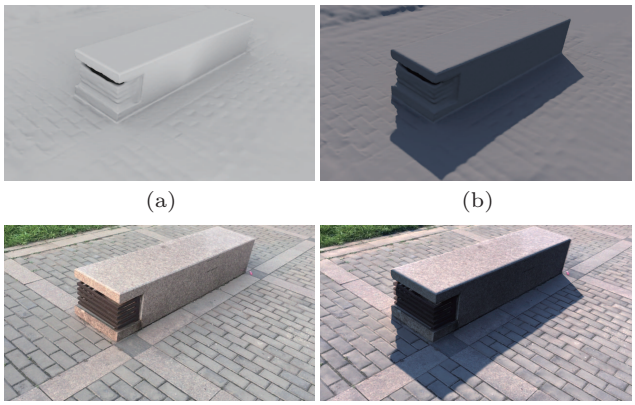
(a) (b) (c) (d)

(e) (f) (g) (h)

Fig.9. Shadow detection and removal. The shadow mask used to guide shadow removal is obtained by thresholding $S(\boldsymbol{x})/S'(\boldsymbol{x})$. (a) $S(\boldsymbol{x})$. (b) $S'(\boldsymbol{x})$. (c) $S(\boldsymbol{x})/S'(\boldsymbol{x})$. (d) Shadow mask. (e) Source frame. (f) Our shadow removal result. (g) Result of [53]. (h) Result of [54].

5 Experiments and Discussions

In this section we first evaluate our method and compare its results with those of other leading methods, and finally discuss its limitations.



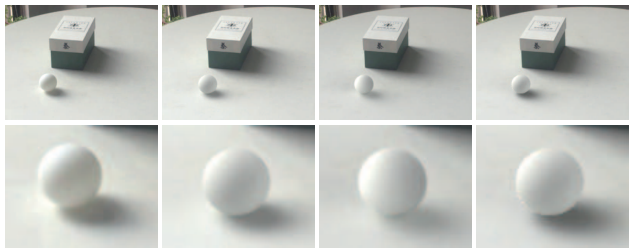
(a) (b) (c) (d)

Fig.10. Video relighting. (a) Scene geometry rendered under recovered illumination. (b) New environment map. The change of received light at each point is calculated and is applied to (c) the input video frame, generating (d) a relit frame.

5.1 Evaluation

To evaluate our illumination recovery method, we compare its results with those produced by the method of [40]. The latter uses von Mises-Fisher (vMF) kernels as basis functions to represent the environment map in (5); sparseness and non-negativity constraints are used when solving for the objective function in (7). As can be seen in Fig.4, the resulting environment map models light as mainly coming from a few directions, which is not representative of the real world. A comparison of the rendered scene geometry shows that our method is superior: the shapes of shadows in our result are a better match to those in the input video frame.

To verify the correctness of our illumination estimation algorithm, we conducted the following experiment. We first took a picture I_N of a scene with a camera mounted on a tripod. We then put a sphere in the scene and took another picture I_O as ground truth at the same camera pose. We also took several pictures of the scene from several viewing directions to enable us to recover geometry. We then estimated the illumination using the pipeline in Section 3, using SH and vMF bases respectively. We then inserted a virtual sphere into the original scene using the method in Subsection 4.1, producing I_C. A comparison of the ground truth image containing a real sphere, I_O, and the augmented image, I_C, is provided in Fig.11 for both sets of basis functions. The environment map using vMF generates much less realistic highlights and shadows than the one using SH. The estimated illumination using SH provides a plausible environment map, and the augmented result is comparable to the ground truth.



(a) (b) (c) (d)

Fig.11. Comparison of augmented images to ground truth. (a) Ground truth image using a real sphere. (b) (c) Augmented reality images with added spheres, using environment maps based on spherical harmonic and von Mises-Fisher bases respectively. (d) Augmented reality images using differential rendering. The bottom row shows the close-ups of the spheres in the upper images.

We measured the similarity between the augmented image and the ground truth using the peak signal-to-noise ratio (PSNR), defined in terms of the mean squared error (MSE) between two images:

$$PSNR(A, B) = -10 \log_{10} (MSE(A, B)),$$

for pixel values in the range [0,1]. We repeated the aforementioned experiment using five different scenes. Table 1 gives the results; higher PSNR values mean that the two images are more similar. Images generated using SH are closer to the ground truth than those using vMF, confirming that SH is more suitable for representing the environment map.

Table 1. PSNR Comparisons Between Ground Truth G and Augmented Images Using Our Illumination Estimation and Object Insertion Method, the Method of [40] and the One from [4]

Scene	Ours	[40]	[4]
1	24.1	20.5	21.7
2	26.4	22.5	27.0
3	23.2	22.6	19.2
4	26.4	23.9	26.1
5	28.0	23.2	27.8

We also compared our object insertion method with differential rendering[4] (see Subsection 4.1) in a similar way. The results are given in Fig.11 and Table 1. In most cases, our object insertion method achieves a higher PSNR value than differential rendering.

5.2 Discussion of Results

We have implemented our illumination estimation algorithm in C++. On a PC with an Intel Core i7 4.0 GHz CPU, it takes around 90 seconds to minimize (8) to find the weights for a 256 basis SH representation. It takes 190 seconds to render all 256 shading results for a single frame. Thus in practice we do not use the shading layers of all frames for computation, and instead we uniformly sample some frames.

In the supplemental material, we present four results. In case 1 (the "cube"), we inserted five virtual objects around a Rubik's cube, including a tea pot, a dragon, a Buddha, a bunny and a spaceship. In the output video, the shadows cast by the objects on the table indicate the estimated environment map is plausible. Also the shadows on the Rubik's cube cast by the teapot and the spaceship are correctly computed.

In case 2 (the "garden"), we inserted a bench, a lion sculpture and a ram sculpture. In this scene, light comes evenly from all directions and the dark area on the ground under the bench shows the effect of ambient occlusion. Although the geometry of the scene is not precisely recovered (in the last few frames, the body of the ram is wrongly occluded by the ground), the artefacts are not readily noticeable.

In case 3 (the "cuboid stone"), the environment map is also nearly constant; we have added a directional light which causes a shadow on the ground.

In case 4 (the "gallery"), we inserted a Lucy statue, a Buddha and a wooden chair into the scene. We assume the scene contains Lambertian surfaces, but the material of the virtual objects can be arbitrary; the materials of the Lucy and the Buddha here are glass and aluminium respectively.

Fig.12 shows the shading layers for the examples in Figs.7 and 10. While they are not entirely accurate, causing inaccurate environment maps, the rendered results remain plausible. Since there are no strong shadows in the original and synthetic scenes, it is hard for viewers to detect any inconsistencies in the shadows.

Fig.12. Shading layers for the examples in Fig.7 and Fig.10.

5.3 Limitations

Our method may be less successful in certain situations. Firstly, illumination recovery does not work correctly when significant shadows are cast by objects lying outside the video frame, as in such cases, the illumination cannot be modeled by a single environment map[57] (see Fig.13).

(a) (b)

Fig.13. Our method cannot model complex illumination which cannot be represented by an environment map, e.g., shadows cast by objects outside the video frame. (a) Hard shadows are cast by a tree outside the frame. (b) Our method fails to recover the illumination, so hard shadows are not produced on the bunny added to the scene.

Secondly, scene reconstruction may inadequately model the surface mesh if the scene is not viewed from a large enough range of viewing directions. In turn, this can lead to incorrect occlusion relations between the scene and any inserted object, causing artefacts. For example, in Fig.14, some of the flowers on the left are wrongly occluded by the synthetic lion because of inaccurately reconstructed scene geometry.

(a) (b) (c)

Fig.14. Failure of scene reconstruction leading to incorrect occlusion. (a) Original frame. (b) Rendered scene geometry with synthetic object. (c) Composed result. Some flowers are missing due to incorrect occlusion relationships.

Thirdly, the geometry of textureless surfaces, for example a white wall, may not be recovered due to a lack of trackable feature points. As Fig.15 shows, part of the wall is missing in the result reconstructed by the automatic routine in Subsection 3.1. This can be overcome with user assistance — the user can refine the scene mesh using 3D modeling software, allowing the overall

approach to still produce convincing results. A further example can be found in Fig.8.

(a) (b) (c) (d)

Fig.15. Failure of scene reconstruction in smooth regions. (a) Textureless surfaces cannot be correctly reconstructed in (b) the scene mesh. This means that when (c) inserting objects, shadows cast on the surface cannot be computed correctly. (d) The problem is solved by manually editing the scene mesh before inserting objects.

Fourthly, SFM may be unable to compute the camera pose for some frames, if they contain an insufficient number of matched feature points. In case 4 of the supplemental material, starting from the 10th second, several frames are missing for this reason.

Finally, our system has other limitations. We assume scene surfaces are made of Lambertian materials, which is only approximately valid for certain real world materials, and quite invalid for others. Also, the illumination is assumed to be constant in each frame, which may not be true of a partially cloudy day, in long-period time-lapse videos or in circumstances with moving light sources.

6 Conclusions

We presented a framework that automatically recovers scene geometry and illumination from a video. Our system reconstructs scene geometry using structure-from-motion, with spherical harmonic basis functions used to approximate the environment map. The recovered illumination and geometry are useful for subsequent applications such as object insertion, shadow detection, and video relighting. While our system has certain limitations, there are also many circumstances in which it works well enough to be useful.

In future, we intend to consider cases involving varying illumination, changing in either time or position. A more sophisticated compositing algorithm would also help to generate more realistic results by simulating complex surface materials.

References

[1] Kee E, O'brien J F, Farid H. Exposing photo manipulation from shading and shadows. *ACM Transactions on Graphics*, 2014, 33(5): 165:1-165:21.

[2] Karsch K, Hedau V, Forsyth D, Hoiem D. Rendering synthetic objects into legacy photographs. *ACM Transactions on Graphics*, 2011, 30(6): 157:1-157:12.

[3] Karsch K, Sunkavalli K, Hadap S, Carr N, Jin H, Fonte R, Sittig M, Forsyth D. Automatic scene inference for 3D object compositing. *ACM Transactions on Graphics*, 2014, 33(3): 32:1-32:15.

[4] Debevec P. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Proc. the 25th Annual Conference on Computer Graphics and Interactive Techniques* (*SIGGRAPH*), Jul. 1998, pp.189-198.

[5] Chen X, Xu W W, Yeung S K, Zhou K. View-aware image object compositing and synthesis from multiple sources. *Journal of Computer Science and Technology*, 2016, 31(3): 463-478.

[6] Bell S, Bala K, Snavely N. Intrinsic images in the wild. *ACM Transactions on Graphics*, 2014, 33(4): 159:1-159:12.

[7] Shen J, Yang X, Jia Y, Li X. Intrinsic images using optimization. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Jun. 2011, pp.3481-3487.

[8] Bousseau A, Paris S, Durand F. User-assisted intrinsic images. *ACM Transactions on Graphics*, 2009, 28(5): 130:1-130:10.

[9] Bi S, Han X, Yu Y. An L1 image transform for edge-preserving smoothing and scene-level intrinsic decomposition. *ACM Transactions on Graphics*, 2015, 34(4): 78:1-78:12.

[10] Laffont P Y, Bousseau A, Drettakis G. Rich intrinsic image decomposition of outdoor scenes from multiple views. *IEEE Transactions on Visualization and Computer Graphics*, 2013, 19(2): 210-224.

[11] Laffont P Y, Bazin J C. Intrinsic decomposition of image sequences from local temporal variations. In *Proc. IEEE International Conference on Computer Vision* (*ICCV*), Dec. 2015, pp.433-441.

[12] Bonneel N, Sunkavalli K, Tompkin J, Sun D, Paris S, Pfister H. Interactive intrinsic video editing. *ACM Transactions on Graphics*, 2014, 33(6): 197:1-197:10.

[13] Kong N, Gehler P V, Black M J. Intrinsic video. In *Proc. the 13th European Conference* (*ECCV*), Sept. 2014, pp.360-375.

[14] Ye G, Garces E, Liu Y, Dai Q, Gutierrez D. Intrinsic video and applications. *ACM Transactions on Graphics*, 2014, 33(4): 80:1-80:11.

[15] Dong Y, Chen G, Peers P, Zhang J, Tong X. Appearance-from-motion: Recovering spatially varying surface reflectance under unknown lighting. *ACM Transactions on Graphics*, 2014, 33(6): 193:1-193:12.

[16] Sato I, Sato Y, Ikeuchi K. Illumination from shadows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2003, 25(3): 290-300.

[17] Panagopoulos A, Samaras D, Paragios N. Robust shadow and illumination estimation using a mixture model. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Jun. 2009, pp.651-658.

[18] Ramanarayanan G, Ferwerda J, Walter B, Bala K. Visual equivalence: Towards a new standard for image fidelity. *ACM Transactions on Graphics* (*TOG*), 2007, 26(3): 76:1-76:11.

[19] Khan E A, Reinhard E, Fleming R W, Bülthoff H H. Image-based material editing. *ACM Transactions on Graphics* (*TOG*), 2006, 25(3): 654-663.

[20] Lalonde J F, Efros A A. Synthesizing environment maps from a single image. Technical Report CMURITR-10-24, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 2010.

[21] Chen T, Zhu Z, Shamir A, Hu S M, Cohen-Or D. 3-sweep: Extracting editable objects from a single photo. *ACM Transactions on Graphics*, 2013, 32(6): 195:1-195:10.

[22] Cao Y P, Ju T, Fu Z, Hu S M. Interactive image-guided modeling of extruded shapes. *Computer Graphics Forum*, 2014, 33(7): 101-110.

[23] Zheng Y, Chen X, Cheng M M, Zhou K, Hu S M, Mitra N J. Interactive images: Cuboid proxies for smart image manipulation. *ACM Transactions on Graphics*, 2012, 31(4): 99:1-99:11.

[24] Wu J, Rosin P L, Sun X, Martin R R. Improving shape from shading with interactive tabu search. *Journal of Computer Science and Technology*, 2016, 31(3): 450-462.

[25] Gupta A, Satkin S, Efros A A, Hebert M. From 3D scene geometry to human workspace. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Jun. 2011, pp.1961-1968.

[26] Jung C, Kim C. Real-time estimation of 3D scene geometry from a single image. *Pattern Recognition*, 2012, 45(9): 3256-3269.

[27] Zhu Z, Martin R R, Pepperell R, Burleigh A. 3D modeling and motion parallax for improved videoconferencing. *Computational Visual Media*, 2016, 2(2): 131-142.

[28] Horry Y, Anjyo K, Arai K. Tour into the picture: Using a spidery mesh interface to make animation from a single image. In *Proc. the 24th Annual Conference on Computer Graphics and Interactive Techniques* (*SIGGRAPH*), Aug. 1997, pp.225-232.

[29] Saxena A, Sun M, Ng A Y. Make3D: Learning 3D scene structure from a single still image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009, 31(5): 824-840.

[30] Hoiem D, Efros A A, Hebert M. Automatic photo popup. *ACM transactions on graphics*, 2005, 24(3): 577-584.

[31] Longuet-Higgins H C. A computer algorithm for reconstructing a scene from two projections. In *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms*, Fischler M A, Firschein O (eds.), Morgan Kanfmann Publishers Inc., 1987, pp.61-62.

[32] Zhang G, Jia J, Wong T T, Bao H. Recovering consistent video depth maps via bundle optimization. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Jun. 2008, pp.1-8.

[33] Jiang H, Zhang G, Wang H, Bao H. Spatio-temporal video segmentation of static scenes and its applications. *IEEE Transactions on Multimedia*, 2015, 17(1): 3-15.

[34] Kopf J, Cohen M F, Szeliski R. First-person hyperlapse videos. *ACM Transactions on Graphics*, 2014, 33(4): 78:1-78:10.

[35] Engel J, Schöps T, Cremers D. LSD-slam: Large-scale direct monocular slam. In *Proc. the 13th ECCV*, Sept. 2014, pp.834-849.

[36] Snavely N, Seitz S M, Szeliski R. Photo tourism: Exploring photo collections in 3D. *ACM Transactions on Graphics*, 2006, 25(3): 835-846.

[37] Agarwal S, Snavely N, Simon I, Seitz S M, Szeliski R. Building rome in a day. In *Proc. IEEE International Conference on Computer Vision* (*ICCV*), Sept. 2009, pp.72-79.

[38] Fuhrmann S, Langguth F, Goesele M. MVE — A multi-view reconstruction environment. In *Proc. the Eurographics Workshop on Graphics and Cultural Heritage* (*GCH*), Oct. 2014, pp.11-18.

[39] Kronander J, Banterle F, Gardner A, Miandji E, Unger J. Photorealistic rendering of mixed reality scenes. *Computer Graphics Forum*, 2015, 34(2): 643-665.

[40] Kholgade N, Simon T, Efros A A, Sheikh Y. 3D object manipulation in a single photograph using stock 3D models. *ACM Transactions on Graphics*, 2014, 33(4): 127:1-127:12.

[41] Zhang G, Dong Z, Jia J, Wan L, Wong T T, Bao H. Refilming with depth-inferred videos. *IEEE Transactions on Visualization and Computer Graphics*, 2009, 15(5): 828-840.

[42] Zhang G, Dong Z, Jia J, Wong T T, Bao H. Efficient non-consecutive feature tracking for structure-from-motion. In *Proc. the 11th ECCV*, Sept. 2011, pp.422-435.

[43] Wu C. Towards linear-time incremental structure from motion. In *Proc. International Conference on 3D Vision*, Jun. 2013, pp.127-134.

[44] OpenMVS: Open multi-view stereo reconstruction library. http://cdcseacave.github.io/openMVS/, Mar. 2017.

[45] Cignoni P, Corsini M, Ranzuglia G. MeshLab: An open-source 3D mesh processing system. *ERCIM News*, 2008, 2008 (73).

[46] Kajiya J T. The rendering equation. In *Proc. the 13th Annual Conference on Computer Graphics and Interactive Techniques* (*SIGGRAPH*), Aug. 1986, pp.143-150.

[47] Jakob W. Mitsuba renderer. http://www.mitsuba-renderer.org, Mar. 2017.

[48] Ramamoorthi R, Hanrahan P. On the relationship between radiance and irradiance: Determining the illumination from images of a convex Lambertian object. *Journal of the Optical Society of America A*, 2001, 18(10): 2448-2459.

[49] Ng R, Ramamoorthi R, Hanrahan P. All-frequency shadows using non-linear wavelet lighting approximation. *ACM Transactions on Graphics*, 2003, 22(3): 376-381.

[50] Hara K, Nishino K, Ikeuchi K. Multiple light sources and reflectance property estimation based on a mixture of spherical distributions. In *Proc. IEEE International Conference on Computer Vision* (*ICCV*), Oct. 2005, pp.1627-1634.

[51] Russell M, Zou J J, Fang G. An evaluation of moving shadow detection techniques. *Computational Visual Media*, 2016, 2(3): 195-217.

[52] P'erez P, Gangnet M, Blake A. Poisson image editing. *ACM Transactions on Graphics*, 2003, 22(3): 313-318.

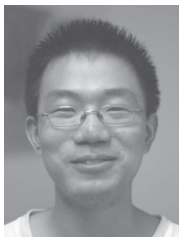
[53] Guo R, Dai Q, Hoiem D. Paired regions for shadow detection and removal. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013, 35(12): 2956-2967.

[54] Gong H, Cosker D. Interactive shadow removal and ground truth for variable scene categories. In *Proc. the British Machine Vision Conference* (*BMVC*), Sept. 2014.

[55] Chen X, Jin X, Zhao Q, Wu H. Artistic illumination transfer for portraits. *Computer Graphics Forum*, 2012, 31(4): 1425-1434.

[56] Chen X, Wu H, Jin X, Zhao Q. Face illumination manipulation using a single reference image by adaptive layer decomposition. *IEEE Transactions on Image Processing*, 2013, 22(11): 4249-4259.

[57] Xing G, Zhou X, Peng Q, Liu Y, Qin X. Lighting simulation of augmented outdoor scene based on a legacy photograph. *Computer Graphics Forum*, 2013, 32(7): 101-110.



Bin Liu is a Ph.D. student in the Department of Computer Science and Technology, Tsinghua University, Beijing. He received his Bachelor's degree in computer science from the same university in 2013. His research interests include image and video editing.

Kun Xu is an associate professor in the Department of Computer Science and Technology, Tsinghua University, Beijing. He received his Bachelor's and Ph.D. degrees in computer science from the same university in 2005 and 2009, respectively. His research interests include realistic rendering, and image and video editing.

Ralph R. Martin obtained his Ph.D. degree in engineering from Cambridge University, U.K., in 1983. He has been a professor at Cardiff University, Cardiff, since 2000. He is Associate Editor-in-Chief of Computational Visual Media, and on the editorial boards of several other journals, including Computer Aided Design, Computer Aided Geometric Design, Computers & Graphics, and Geometric Models.