Capture to Rendering Pipeline for Generating Dynamically Relightable Virtual Objects with Handheld RGB-D Cameras

Chih-Fan Chen
University of Southern California
chihfanc@usc.edu

Evan Suma Rosenberg
University of Minnesota
suma@umn.edu

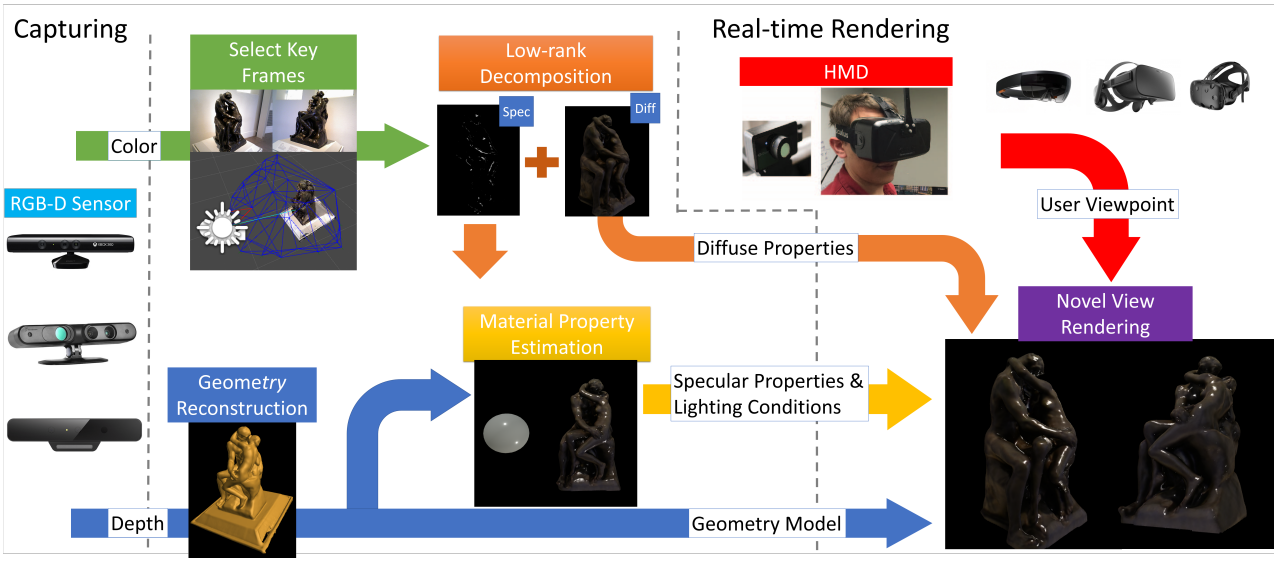


Figure 1: Overview of the virtual content creation pipeline. Color and depth image streams are captured from a single RGB-D camera, and the geometry is reconstructed from the depth information. Each selected frame is separated into diffuse and specular maps using low-rank decomposition, and the camera poses are optimized. The lighting conditions and surface reflectance parameters are computed from the specular maps. Finally, the texture of the dynamically relightable reconstructed object is rendered in real-time based on the user's current viewpoint.

ABSTRACT

We present a complete end-to-end pipeline for generating dynamically relightable virtual objects captured using a single handheld consumer-grade RGB-D camera. The proposed system plausibly replicates the geometry, texture, illumination, and surface reflectance properties of non-Lambertian objects, making them suitable for integration within virtual reality scenes that contain arbitrary illumination. First, the geometry of the target object is reconstructed from depth images captured using a handheld camera. To get nearly drift-free texture maps of the virtual object, a set of selected images from the original color stream is used for camera pose optimization. Our approach further separates these images into diffuse (view-independent) and specular (view-dependent) components using low-rank decomposition. The lighting conditions during capture and reflectance properties of the virtual object are subsequently estimated from the computed specular maps. By combining these parameters with the diffuse texture, the reconstructed model can then be rendered in real-time virtual reality scenes that plausibly replicate real world illumination at the point of capture. Furthermore, these objects can interact with arbitrary virtual lights that vary in direction, intensity, and color.

VRST '20, November 1–4, 2020, Virtual Event, Canada

ACM ISBN 978-1-4503-7619-8/20/11…$15.00
https://doi.org/10.1145/3385956.3418952

CCS CONCEPTS

• **Computing methodologies** → **Virtual reality**; **Reconstruction**; **Texturing**.

KEYWORDS

virtual reality, content creation, scanning, reconstruction

ACM Reference Format:
Chih-Fan Chen and Evan Suma Rosenberg. 2020. Capture to Rendering Pipeline for Generating Dynamically Relightable Virtual Objects with Handheld RGB-D Cameras. In *26th ACM Symposium on Virtual Reality Software and Technology (VRST '20), November 1–4, 2020, Virtual Event, Canada.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3385956.3418952

1 INTRODUCTION

With the recent proliferation of consumer head-mounted displays, research topics in virtual and mixed reality have been developing vigorously. In many applications, customized virtual content is one of the essential elements for building virtual worlds. Using ready-made virtual objects can potentially compromise the desire of developers or decrease the interests of users. However, manual creation of high-fidelity virtual content requires expert knowledge of 3D modeling, and is an often expensive and time consuming endeavor. A promising alternative is scanning physical objects and replicating their photorealistic appearance in the virtual environment. With the development of consumer-grade RGB-D sensors such as the Microsoft Kinect or Intel Realsense, grab-and-go 3D scanning is becoming increasingly accessible for general users. As a result, reconstructing the geometry of objects captured using depth sensors has been an extensive research topic in recent years (e.g., [16, 26, 40, 46]). However, replicating the photorealistic appearance of reconstructed virtual objects from an RGB-D sequence is still an open question. Existing methods (e.g., [2, 48]) averaged the colors from all captured images to compute the color of each vertex, which often results in lower fidelity textures. Other approaches (e.g., [7, 38]) have selected a single view per face to achieve higher color fidelity rendering for Lambertian models (i.e., models with diffusely reflecting surfaces). However, for non-Lambertian materials, texture maps with baked lighting would limit the potential viewing directions, which is not ideal for real-time virtual reality applications. Ideally, surface illumination (e.g, specular reflections) should dynamically change based on the user's viewpoint, and the absence of these dynamic visual cues can be especially noticeable in head-tracked virtual reality.

To overcome these limitations, view-dependent texture mapping (VDTM) techniques have been introduced with promising results [6, 11, 25]. In this approach, the object's appearance is dynamically rendered using a subset of captured images closest to the current virtual camera position. These methods can result in photorealistic quality because the surface illumination of the virtual model will change based on the user's viewpoint. However, the synthesized specular reflections are still problematic because the reflectance of an object is strongly related to its geometry, surface material properties, and environmental lighting conditions. The synthetic views created from interpolating between color images may have visual artifacts or inconsistencies with the geometry, especially for data captured with sparse camera trajectories. Moreover, these reconstructed objects would be incompatible with virtual environments that include dynamic lighting.

In this paper, we propose a capture-to-rendering content creation pipeline that estimates diffuse and specular reflectance of the original object and lighting conditions (see Figure 1). Given an RGB-D sequence, the geometry of the object is first reconstructed and several color frames are selected from the original stream. We introduce a novel texture separation method based on low-rank decomposition that simultaneously optimizes the camera poses of each frame and separates each color frame into a diffuse map and a specular map. Using the specular maps from many different viewing directions, the surface reflectance properties and lighting conditions can be estimated. At run-time, the optimized diffuse textures and reflectance parameters can then be used to synthesize the captured object's appearance from an arbitrary viewpoint. Our method can generate plausible results even for unseen views that were not present in the capture dataset. Moreover, the reconstructed virtual content can be readily integrated with virtual scenes and dynamically illuminated with lights of varying direction, intensity, and color. The major contributions of this paper include:

- A texture synthesis approach that jointly solves for diffuse and specular decompositions and optimized camera poses.
- Estimation of the light sources in the capture environment using the computed specular components. Optimized reflectance parameters are subsequently derived from the estimated lighting conditions.
- A rendering method that combines specular and diffuse components in real-time. Results show that the appearance of the original object can be plausibly replicated, including previously unseen views of the reconstructed models.
- A fully automatic virtual content creation pipeline that does not require expert knowledge or manual human effort. The reconstructed models are suitable for integration with industry-standard virtual environments.

2 RELATED WORK

A wide variety of techniques have been proposed to reproduce the visual appearance of physical objects in virtual environments. In several review papers [29, 34, 37, 49], they can be categorized as *image-based* and *model-based* methods based on the way unobserved viewpoints are represented and visually reproduced.

Image-Based Rendering. Light field rendering (LFR) [3, 5, 10, 21] is a method for synthesizing an unseen view without using the geometry of an object. Assuming the stored images are large enough to capture all the reflected lights coming from an object, LFR generates synthetic views by ray-tracing to find each pixel color from its image dataset. LFR can achieve photorealistic quality but requires a well-designed camera array or a programmable turntable. These specialized devices require expert knowledge and are not practical for most users. Furthermore, without a 3D model for rendering, it is not possible to edit or interact with other virtual content. View-dependent texture mapping is another image-based rendering method that can produce photorealistic results [11, 15, 25]. Given a set of selected images with known camera poses, these techniques blend the color maps from cameras closest to the user's current viewpoint and dynamically render the texture onto a 3D model at run-time. However, VDTM is very sensitive to errors such as missing data or inaccurate camera pose estimation. Dynamic Omnidirectional Texture Synthesis (DOTS) [6] generates synthetic views surrounding the reconstructed object, which are more robust to inconsistent camera trajectories and missing coverage during object capture. However, for non-Lambertian objects, interpolating from several images often cannot generate a correct appearance because specular reflections are the result of complex view-dependent interactions between the object's geometry, surface material properties, and environmental lighting conditions. Furthermore, real-time dynamic relighting with arbitrary virtual illumination is generally not possible with image-based rendering techniques.

Model-Based Rendering. Model-based methods reconstruct the geometry of the object from several images captured in different viewing directions and represent virtual appearance of surfaces using albedo, specular maps, normal maps, etc. However, computing average maps by blending all the observed images of a reconstructed object can result in lower visual fidelity (e.g., blurry textures). In previous work, color mapping optimization [48] was introduced to estimate the albedo of the model by maximizing the photometric agreement between multiple images. Several papers [2, 14, 17] generated optimized texture maps to improve the visual quality of the Lambertian surfaces. However, it is not ideal to represent the dynamic illumination effects of non-Lambertian surfaces using fixed texture maps. Dong et al, [13] proposed "appearance from motion" to model the reflectance from an image sequence or video. However, the physical object should be captured in a well-controlled environment. Wu et al. [43] attempted to estimate the surface reflectance in uncontrolled environments. However, this approach requires specialized IR capture devices and is therefore not practical for everyday use with consumer-grade cameras. Shi et al. [32] reviewed several methods for recovering the BRDF model of a target object under different lighting conditions. However, these methods require the target to be captured with known illumination; thus, they are not suitable for portable scanning in the wild. Other methods (e.g., [12, 23]) have used a single image to estimate the BRDF. However, additional constraints and assumptions are necessary to deal with uncertainty when only one observation is given.

Recently, several papers [19, 31, 42] have proposed methods that used RGB-D sensors to estimate material properties. To achieve good quality results, these methods had strict material constraints for captured objects. [19] assume the number of lighting should be known and in [42] and [31] they number of material should be given, which usually are not practical for random RGB-D sequence. The following two papers are most closely related to our work: Park et al. [27] used an IR sequence to estimate the specular map and then compute a diffuse map in real-time, and Wei et al. [39] used low-rank decomposition for highlight removal. Both methods assume that the camera poses obtained using KinectFusion [16] were sufficiently accurate; however, accumulated drifting errors can result in a failure to recover specular properties. Thus, these approaches only work for objects with short capture sequences.

Deep Learning Approaches. Recently, deep learning has resulted in significant breakthroughs for traditional computer vision research in areas such as object recognition, event detection, scene understanding, etc. Several papers have also applied deep learning to the 3D reconstruction problem. Hou et al. [18] and Avetisyan [1] have aimed to improve reconstruction from RGB-D data. They primarily focused on generating 3D geometry, but not textures. Li et al. [22] recovered the shape and reflectance properties from an image, and Meka et al. [24] estimated the material properties. However, these papers focused on reconstruction from a single image, which only has partial view of the target object, and is therefore not ideal for VR applications that allow objects to be viewed from any potential direction. Xu et al. [44] recovered the texture of objects from multiple images. However, this approach required capture using a light stage, which is not allow for the possibility of portable scanning. This approach also reported a processing time of 2 seconds to render an unseen view, which is much too slow for real-time rendering in virtual reality. For image-based rendering, neural networks have also been used to convert a single RGB-D input image into a 3D photo [33], and deferred neural rendering was recently proposed to synthesize images using learned feature maps that are trained during the scene capture process [36].

In this paper, we present an automated end-to-end pipeline for reconstructing the geometry, surface materials, and illumination of objects captured in unconstrained conditions using a single handheld RGB-D camera. Our approach also simultaneously optimizes the camera trajectories, making it practical for objects that require long scanning sequences. This pipeline was not designed to compute a physically accurate bidirectional reflectance distribution function; instead, we focus specifically on generating visually plausible results for efficient real-time rendering in virtual reality. This work, therefore, occupies a unique niche in the literature.

3 OVERVIEW

The color appearance of virtual objects is often represented as a linear combination of two components: diffuse and specular reflection. Diffuse reflection only depends on the light sources and material properties, and is therefore view-independent. In contrast, specular reflectance is highly viewpoint dependent, and incorrect reflections will become especially noticeable during smooth motion. Thus, specularity is particularly important for high-fidelity rendering in virtual reality applications, because the user's viewpoint will be continuously updated using head tracking. A naive method would directly use the selected frames G from the original color-stream to render the object with photorealistic textures, and the reflectance can be accurately replicated when the viewing direction is perfectly aligned with one of the camera poses. However, for objects captured using handheld consumer-grade RGB-D cameras, the trajectories are often sparse and unstructured, especially when performed by non-expert users. Therefore, rendering an unseen view (i.e., the viewing direction cannot be found in the original stream) by interpolating between the closest frames often leads to poor results, especially when specular reflections are present. Thus, it is necessary to identify the specularity of a target 3D object from a given RGB-D sequence. With the estimated reflectance, a reconstructed virtual object can be better modeled under different lighting conditions and environments. To achieve this goal, we need to separate the original images into diffuse and specular maps.

An overview of the system pipeline is shown in Figure 1. The object's geometry is first reconstructed from the depth stream captured using a consumer-grade RGB-D camera. A set of keyframes are selected from the entire color stream, and the camera poses of those frames and the diffuse/specular maps are optimized from the low-rank objective function. Lighting conditions and surface material reflectance properties are estimated from the specular components. At run-time, the user viewpoint provided by a head-tracked virtual reality display is used to generate the view-dependent appearance of the reconstructed 3D model in real-time.

3.1 Object Capture and Reconstruction

Multiple capture and reconstruction methods could be used to obtain a 3D model with a triangular mesh representation. Similar to

other papers[2, 6, 17, 48] that focus on the texture mapping for objects captured using handheld RGB-D cameras, we use KinectFusion [16] as the first step in reconstructing the 3D model M from the captured sequence of depth images. The camera trajectory is also roughly estimated for use later in the texture generation process.

Using all the frames I from the input video sequence to synthesize a global texture is inefficient, and some frames may negatively impact the results due to artifacts from the camera's motion (e.g., blurriness). Instead, temporal [2, 48] and spatial [6, 17, 31] keyframe selections have been proposed. These methods use either the distribution of time or space to select a subset of representative frames from the original color sequence I. We chose spatial keyframe selection to maximize the variation in viewing angles of the 3D model. The selected n keyframes, $G = \{g_1, g_2, \cdots, g_n\}$, and the initial estimated camera poses, $T_g = \{t_{g_1}, t_{g_2}, \cdots, t_{g_n}\}$, are inputs for the texture generation process. The blurriness [9] of each frame is computed to sort all the images and select the frames from the lowest to the highest blurriness which satisfy the following constraint: $||p_i - p_j||_2^2 > d \quad \forall i, j \in G$, where p is the camera position and d is the minimum distance between any selected images. The number of keyframes, n, varies because it depends on the trajectory of the original video sequence.

3.2 Reflection Model Estimation

The colors observed in eyes or images are a combination of both illumination and the object color. In order to approximate these light reflections and surface properties, several empirical models have been proposed to describe the appearance changes. In this paper, we use the well-known Phong reflection model [28]. We use the following equation to compute the illumination of each surface point I_p by combining I_d and I_s:

$$\begin{aligned} I_p &= I_d + I_s \\ I_p &= \sum_{j \in J} k_d (L_j \cdot N) i_j + \sum_{j \in J} k_s (R_j \cdot V)^{\alpha} i_j \end{aligned} \tag{1}$$

where k_d and k_s are the reflection constants for the diffuse and specular components, J is the set of all lights, i_j is the intensity of the $j-th$ light source, L_j and R_j are the corresponding incident and reflection directions from the surface, N is the surface normal, V is the direction toward the viewer, and α is its shininess parameter.

However, separating a single image into two images is an ill-posed problem. Moreover, the unknown camera poses of each image make this problem more challenging. In Section 4, the proposed method simultaneously locates the optimized the camera positions and extracts the diffuse map I_d (i.e., independent with view-point) from several observations from different keyframes. The residual value is treated as the specular components I_s and is handled by minimizing the error function in Section 5.

4 DIFFUSE AND SPECULAR MAP SEPARATION

In previous work, Wright et al. [41] applied robust principal component analysis to decompose a set of pixel-wise registered images into specular and diffuse components. However, in our case, the images are captured from a handheld camera in uncontrolled conditions. Each individual image is unable to cover all directions of the target object, and the reconstruction must be robust to missing data, camera pose drift, and motion blur. To solve this more challenging problem, we apply the low-rank matrix recovery and solve jointly for specular and diffuse decompositions, camera poses, and vertex colors. We do this by alternating between optimizing for the specular and diffuse components and for the camera poses and vertex colors.

4.1 Camera Pose Optimization

As described in Section 3, the reconstructed model M, selected n images G, and the initial camera pose estimation T_G is obtained. However, the raw camera poses are not accurate enough for mapping the texture onto the model correctly. Therefore, Zhou et al. [48] proposed a color mapping optimization to get the optimized camera poses T_G and the vertex color $C(v)$ of the reconstructed model M. The objective function is defined as follows:

$$E(C, T_G) = \sum_{g \in G} \sum_{v \in M} (C(v) - \Gamma(v, g, t_g))^2 \tag{2}$$

where Γ retrieves the color by projecting vertex v to image g using the estimated pose t_g. The objective function iteratively solves $C(v)$ and T_G to find the optimal solution.

4.2 Low-Rank Decomposition

The appearance of an object from a certain viewpoint can be divided into two components: isotropic (i.e., diffuse maps) and anisotropic (i.e., the specular map). According to the Lambertian reflectance model, the diffuse textures obey Lambert's cosine law and do not change when viewed from different directions. Assuming the selected images K are well aligned and the illumination environment is fixed, the diffuse layers are strongly correlated. In Figure 2, with known camera poses, projecting all color images of selected frames into a certain viewpoint, the synthetic images can be vectorized into an image matrix A. Our objective is to separate A into a diffuse matrix D and a specular matrix S. The rank of the diffuse matrix D is considered low, since the observed intensity of a Lambertian surface is the same regardless of the observer's angle of view. On the other hand, the specular matrix S contains only the highlights and forms a sparse matrix. We use low-rank matrix recovery [4, 41] to separate each frame into a diffuse and specular map. Low-rank decomposition minimizes the nuclear norm of $||D||_*$ while reducing the 1-norm $||S||_1$:

$$\begin{aligned} min_{D,S} \quad & \|D\|_* + \lambda \|S\|_1 \\ s.t. \quad & A = D + S \end{aligned} \tag{3}$$

By further refining Eq. 3, we can obtain the diffuse map and the specular map of each selected frame.

4.3 Texture Separation and Pose Estimation

In this pipeline, we aim to simultaneously estimate the camera poses and separate the frames. For each image $g_i \in G$, it can be divided into diffuse image d_i and specular image s_i using low-rank decomposition. The diffuse component d_i is used to refine camera pose by projecting all the vertices v of the model M onto image d_i using the estimated transformation matrix T_i. Note that vertices that

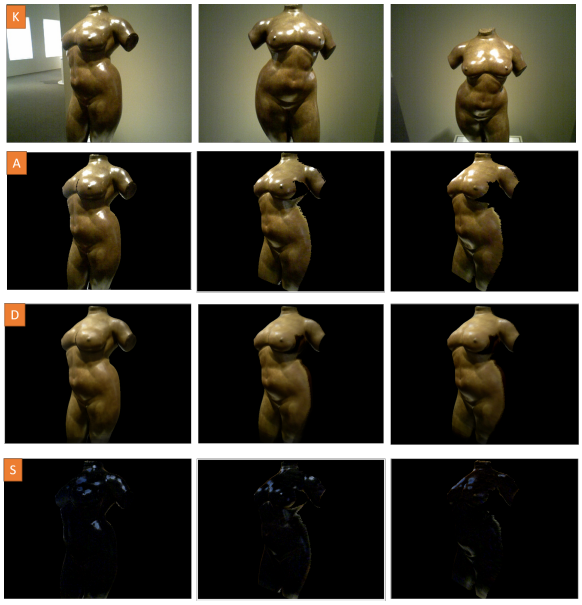
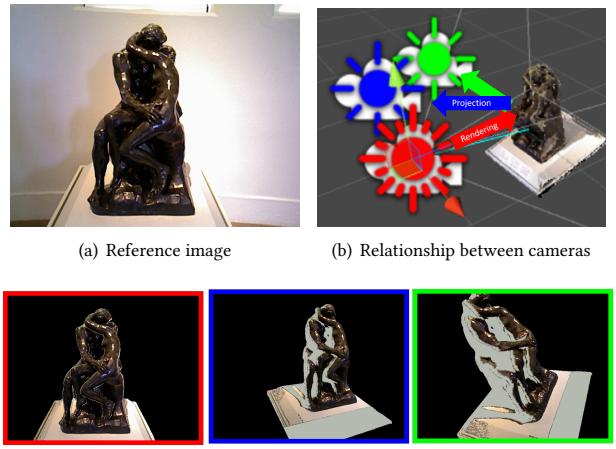


Figure 2: Separation of selected frames into a diffuse map and a specular map. The original texture of selected frames (top row) are projected to the camera pose of the left frame (second row) to form a data matrix A. Low-rank decomposition is able to separate A into a diffuse matrix (third row) and a specular map (fourth row).



(a) Reference image (b) Relationship between cameras

(c) Projected results compared to the closest camera view

Figure 3: An example of the texture synthesis function Ψ. (a) and (b) show a reference image with a known camera pose (red). This image is used to render the model and the other two camera views (blue and green) are projected to its camera view. (c) The projected results of the three camera views. Note that the occluded area (i.e., not visible in (a)) is rendered in gray for better visualization.

do not pass a visibility check (i.e., either out of image or occluded by other vertices) will be discarded. We reformulate the original Eq. 2 as follows:

$$E(C, D, T_G) = \sum_{d_i \in D} \sum_{v \in M} \delta * (C(v) - \Gamma(v, d_i, T_{g_i}))^2 \tag{4}$$

where $D = \{d_1, d_2, \cdots, d_n\}$ is the set of all diffuse maps. If the vertex in image d_i is visible, $\delta = 1$. Otherwise, $\delta = 0$. Because Γ and low-rank decomposition are both non-linear, the Eq. 4 is solved by iteratively updating camera pose T_G and vertex color C.

Texture Synthesis Function Ψ. The low-rank decomposition requires an image matrix A where each column represents an observation of the target object. Each row of A should be highly correlated. Since the key frames (Figure 2 top row) are captured from different viewpoints, directly using those images is not possible to form a reasonable image matrix A. Thus, to utilize those images in our texture decomposition setting, they need to be aligned first. We introduce the texture synthesis function Ψ for synthesizing the textures from each image to a selected camera view. Given a model M and the camera pose t_i of an image i, Ψ renders the model M by image i and projects the rendered model to a known camera pose t_j. Thus, the texture synthesis function is defined as $\Psi(i, t_i, M, t_j)$. For example, to synthesize the texture of Figure 3(a) in other camera views, the model is rendered from the image with its corresponding camera pose (i.e., the red camera in Figure 3(b)). The rendered model is projected to another known camera pose (e.g., the blue or the green camera in Figure 3(b)) to generate the synthetic texture. Figure 3(c) shows the projected results of three camera views. As shown in Figure 3(a) and (c) left, the synthetic texture of the model generated from ψ is the same with the texture from the original image. Note that the occluded area on the model is gray for better visualization in the paper and is set to black (i.e., zero) in our experiments.

Updating D. To update the diffuse map d_i, the selected frames $G = \{g_1, g_2, \cdots, g_n\}$ and their current camera poses T_K are fixed. We first apply $\Psi(i, t_i, M, t_j)$, where $t_j \in T_K$, to synthesize the textures from all keyframes. We also generate the depth map of the model from the camera pose. A binary mask derived from the depth map represents the existence region of the model in the synthetic texture. The mask is used to select the region of interest for each projected synthetic view and stacks it as a vector. Since the mask is based on the geometry instead of appearance, the length of all vectors is the same. The image matrix $A_i = \{a_1, a_2, \cdots, a_n\}$ is formed by concatenating all the image vectors. Since the prerequisites for low-rank decomposition require that the low-rank matrix is strongly correlated, using the synthetic texture of an image far from the camera position will increase the sparse error. The ratio of nonzero pixels between synthetic images $a_j, j \neq i$ and the rendered image a_i is used as a constraint to discard those unqualified synthetic textures.

$$\begin{aligned} A'_i &= \{a_1, a_2, a_3, \cdots, a_m\}, \quad m \leq n \\ \forall a_j \in A'_i, &\quad NZ(a_j)/NZ(a_i) \geq \gamma \end{aligned} \tag{5}$$

where $NZ(i)$ is the function to compute non-zero pixels in an image i. In our experiments, $\gamma = 0.8$.

The objective function aims to separate the data A'_i to a low-rank diffuse matrix L_i and sparse specular matrix S_i. Note that the illumination intensity should not be negative, and so we add these

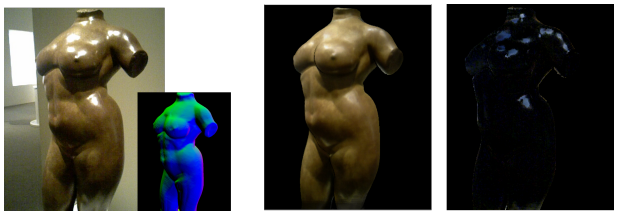


Figure 4: (Left) An example of a selected frame and its surface normal. (Middle) The diffuse map is unable to recover the lighting condition. (Right) In contrast, the specular map is sparse and the lighting direction can roughly assumed from the top. Using this assumption from all selected images, we can approximately calculate the number of lights and their directions for reflectance estimation.

constraints to the original low-rank decomposition. The function can be rewritten as follows:

$$\begin{aligned} \min_{L_i, S_i} \quad & ||L_i||_* + \lambda||S_i||_1 \\ A'_i = L_i + S_i, \quad & L_i \geq 0, \quad S_i \geq 0 \end{aligned} \tag{6}$$

Because specular highlights are view-dependent, they will not be accurate when projected to a different viewpoint using Ψ. Therefore, they will be treated as sparse error and can be separated by the low-rank decomposition. The intermediate results of diffuse and specular map separation are saved as $D = \{d_1, d_2, \cdots, d_n\}$ and $S = \{s_1, s_2, \cdots, s_n\}$. In the third row and fourth row of Figure 2, we show the final separation results.

Updating T and C. Since D is fixed, we use the original color mapping optimization to further refine the camera poses for each frame. The original image in Eq. 2 is replaced with diffuse map d_i in the color mapping optimization. This further improves camera pose estimation accuracy due to the removal of specular highlights.

5 MATERIAL AND LIGHTING ESTIMATION

5.1 Objective Function

Material Estimation from Specular Map. In Section 4, we have already separated each selected image into a diffuse and specular map. Without the knowledge of the amount of lights M and their directions $L_m, I_d = \sum_{j \in J} k_d (L_m \cdot N) i_j$ has infinite solutions that can minimize the objective function. Thus, it is impossible to recover k_d of the virtual model from the diffuse maps. However, in Figure 4, the specularity is sparse and highly depends on the viewing angle and the direction of lights. Using both the specular map $S = \{s_1, s_2, \cdots s_n\}$ and surface normals, we are able to derive the lighting directions first and then estimate the light and material properties of the highlighted region of the object.

Error Function. We introduce the $\rho(S_a, b)$ function to retrieve the value from specular map S_a using the corresponding transformation matrix T_a and the camera intrinsic parameters. To simplify notation, we use k instead of k_s in our equation. For any vertex $v \in M$, the error function is defined as follows:

$$E\{k, \alpha, I_J\} = \sum_{S_a \in S'} (\rho(S_a, v) - k \sum_{j \in J} (R_j \cdot V_j)^{\alpha} i_j)^2 \tag{7}$$

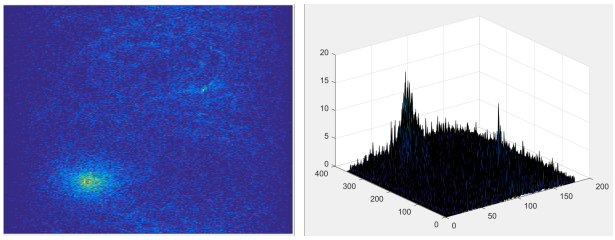


Figure 5: The results of possible incident light direction. Note that each highlight pixel votes for a possible direction. Mean-shift algorithm is applied to get the number of clusters and their centers.

where S' is the set of frames that v is not occluded and i_j is the intensity of each light.

To find the optimized lighting and material properties for all vertices, the objective function is defined as follows:

$$\{K, A, I_J\} = \arg\min \sum_{S_a \in S} \sum_{v \in v_a} (\rho(S_a, v) - k_v \sum_{j \in J} (R_{vj} \cdot V_{av})^{\alpha_v} i_j)^2 \tag{8}$$

where v_a is the set of visible vertices in S_a. For each visible vertex v, k_v is the specular coefficient, R_{vj} is the reflection direction with respect to incident light j, and V_{av} is the viewing direction from the camera pose of S_a to the vertex v. In this objective function, we aim to find the optimized specular coefficient $K = \{k_1, k_2, \cdots, k_m\}$, shininess parameter $A = \{\alpha_1, \alpha_2, \cdots, \alpha_m\}$, and the intensity of lights $I_J = \{i_1, i_2, \cdots, i_J\}$.

Light Direction Estimation. To find the optimized solution for Eq. 8, we need to first determine the number of lights and their directions. As the specular reflection in the Phong shading model is defined as $(R \cdot V)^{\alpha}$, the specularity only appears when the viewing direction v and the reflection direction r is similar. Therefore, we are able to compute the approximate incident direction l by using $l = 2 * (v \cdot n) * n - r$, where n is the surface normal.

For each highlight pixel in the specular map, an approximated direction is added as a candidate. We assume that the incident lights are all directional lights, so they can be converted from Cartesian coordinates to a sphere coordinate system (radius = 1). Figure 5 shows the voting results of the incident light. The number of clusters and their centers can be obtained by mean-shift [8]. We discard clusters with less than 0.2 times number of votes received by the largest cluster.

Optimization. Given the optimized camera poses calculated in Section 4 the estimated lighting directions, and the pre-computed vertex normals, the above equation can be rewritten as follows:

$$\begin{aligned} \{k, \alpha, i\} = \arg\min & \sum_{a=1}^{n} \sum_{b \in v_a} (y_{ab} - k_b \cdot \sum_{c=1}^{l} (x_{abc})^{\alpha_b} \cdot i_c)^2 \\ s.t \quad & \sum_{c=1}^{l} k_b \cdot (x_{abc})^{\alpha_b} \cdot i_c < y_{ab}, \\ & 1 \geq k_b \geq 0, \qquad \alpha_{max} \geq \alpha \geq 0, \qquad 1 \geq i \geq 0 \end{aligned} \tag{9}$$

Figure 6: Visualization of estimated K_s and the α. Darker areas correspond to smaller values for each coefficient.

where $y_{ab} = \rho(S_a, b)$ is a scalar for camera view a and vertex b, and $x_{abc} = (R_{bc} \cdot V_{ab})$ is a scalar based on camera view a, vertex b, and incident light c. α_{max} is the constraint that bounds the value of *alpha* to avoid over-fitting. In our experiment, we use $\alpha_{max} = 60$.

The optimized specular reflectance k and the shininess parameter α can be used in real-time rendering. In Figure 6, the estimated K_s and α are shown. Note that for better visualization, we scale the K_s value range from $0-1$ to $0-255$ and α from $0-\alpha_{max}$ to $0-255$. Note that those highlight regions in the original image had higher K_s and α, while the regions without specular effects had lower values.

6 REAL-TIME RENDERING

A specialized shader was developed to render the reconstructed objects in the Unity game engine using the reconstructured 3D mesh, diffuse texture, and estimated reflectance parameters. In Figure 7, we compare our rendering results with their original images. Note that our approach can replicate the specular highlights in the selected frames. A virtual sphere was added to illustrate the estimated lighting directions in the real world capture scene.

Data Sets. The virtual object dataset used to test the proposed pipeline contains thousands of RGB-D sequences captured by non-experts with a Primesense camera [7]. The raw color and depth are not synchronized, so we assigned the color images to the depth images with the smallest time-stamp difference. Since the streams are both 30 fps, the shifting error is small and can also be handled by the optimization. We demonstrate results from three scans that exhibited large amounts of specular reflectance: Torso of Elevation, the Kiss by Rodin, and an antique leather chair (IDs 3887, 4252, and 5989 in the database). Information about the captured dataset for each object is shown in Table 1.

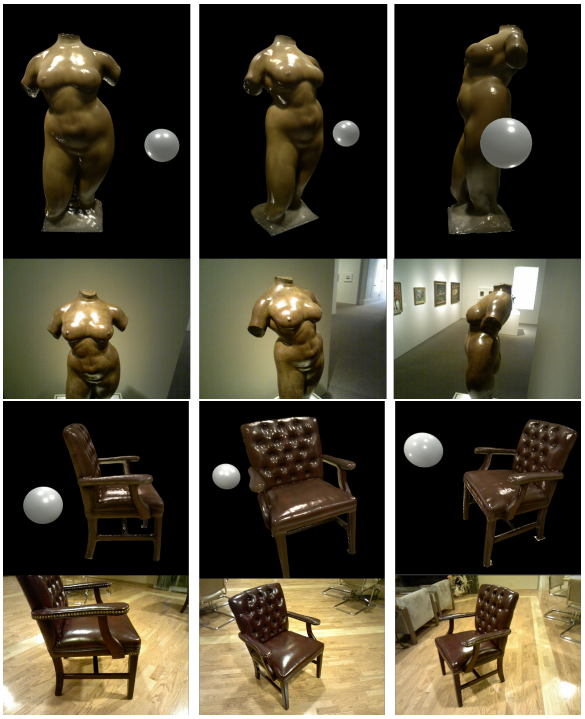


Figure 7: Comparison of rendered virtual objects and original capture frames. The highlights on the sphere visualizes the direction and color of the virtual lights estimated from the real world scene.

Table 1: Information about the capture images and models.

Object	vertex	surface	frames used	color/depth stream
Torso	208K	406K	101	3210 / 3225
The Kiss	280K	544K	116	3989 / 4007
Chair	255K	495K	98	3299 / 3313

Apparatus and Implementation. All the results were performed in Unity 2017.2.0f3 on a MacBook Pro with an Intel i7-4850HQ CPU, Nvidia GeForce GT750M GPU, and 16 GB of RAM. Our method can render the models in 10-15 milliseconds (i.e., 70-90 fps), making it sufficient for real-time rendering even on a laptop.

6.1 Visual Analysis

Low-rank Decomposition. In a previous approach [48], the albedo of the model is estimated by averaging the color from all observed frames to represent the diffuse map of the model. In Figure 8, we compare the diffuse maps computed from original images and the diffuse map from the low-rank texture separation. The specular regions are outliers with high intensity, thus, averaging the color leaves some highlight effects on the diffuse map. In contrast, the diffuse map generated from low-rank decomposition removes most of the highlights and moves them to the specular maps.

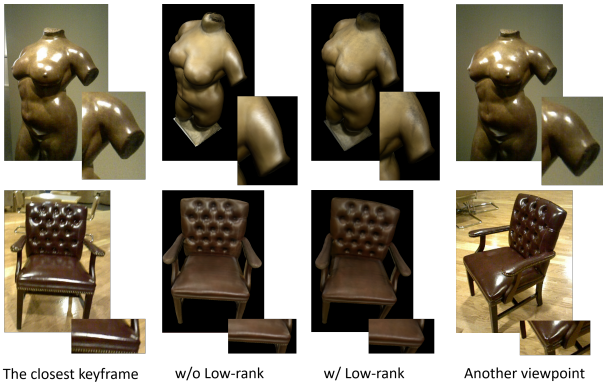


Figure 8: Comparison of the diffuse appearance using the average vertex color from the original images (without low-rank decomposition) and the derived diffuse maps (with low-rank decomposition). Our approach can remove most of the specular highlights such as the shoulder area on the Torso and the self inter-reflection on the seat cushion. Moreover, by removing the highlights, the details from the original images can be correctly preserved in the diffuse map.

Method Comparison. To compare the fidelity of model, we compare our method with fixed textures [48] and Dynamic Omnidirectional Texture Synthesis (DOTS) [6], a previously proposed view-dependent texture mapping method. As shown in Figure 9, three original images are selected for comparison. The top two rows show selected key frames, and the bottom row shows results from unselected frames (i.e., an unseen view). Note that the appearance does not change when using fixed textures, and the fidelity is recognizably lower than the other two methods. DOTS can generate photorealistic results if the user viewpoint is close to selected frames because it directly renders the texture onto the model. However, the synthesized appearance is inaccurate if the user viewpoint moves further away from the camera trajectories in the original dataset, which occurs often for objects captured using handheld cameras. Since specular effects are local phenomena based on the user viewpoint and lighting direction, it is not always possible to interpolate from the textures of other views. In our approach, the specular reflection is optimized based on the texture, lighting, and the geometry; thus, the results are still visually plausible for unseen views that are further away from images in the capture dataset.

6.2 Dynamic Relighting

Virtual objects created with fixed textures or VDTM are difficult to integrate with arbitrary scenes because the illumination during capture is baked into the textures. In contrast, with the reflectance properties estimated using our method, we can more readily adapt the reconstructed models into virtual environments with varying lighting conditions. In Figure 10, we demonstrate the integration of three reconstructed virtual objects into a scene with dynamic illumination. We can increase or decrease the intensity of the lights (a), and the appearance changes of both models are consistent with the illumination in the surrounding environment. Furthermore, the color and direction of lights can also be changed dynamically

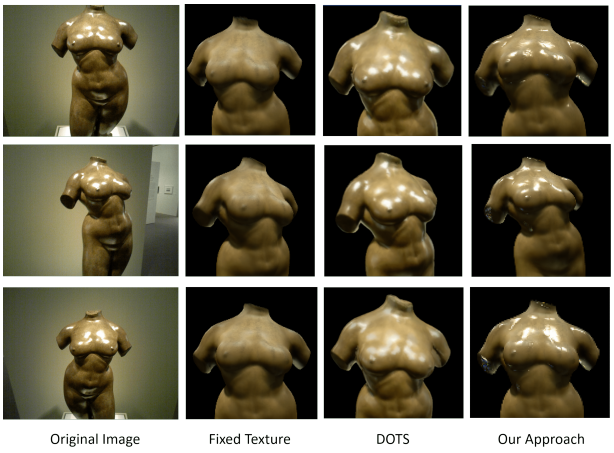


Figure 9: Comparison of our method with fixed textures [48] and DOTS [6]. Note that the fixed texture results in lower fidelity (blurriness) due to averaging the observed images. DOTS or other view-dependent texture mapping methods are able to generate photorealistic rendering results. However, the specular highlights of an unseen view cannot be correctly interpolated from the source images (bottom row). In contrast, our method estimated the light sources and the specular reflectance properties of the object, and is able to synthesize the highlights of unseen views.

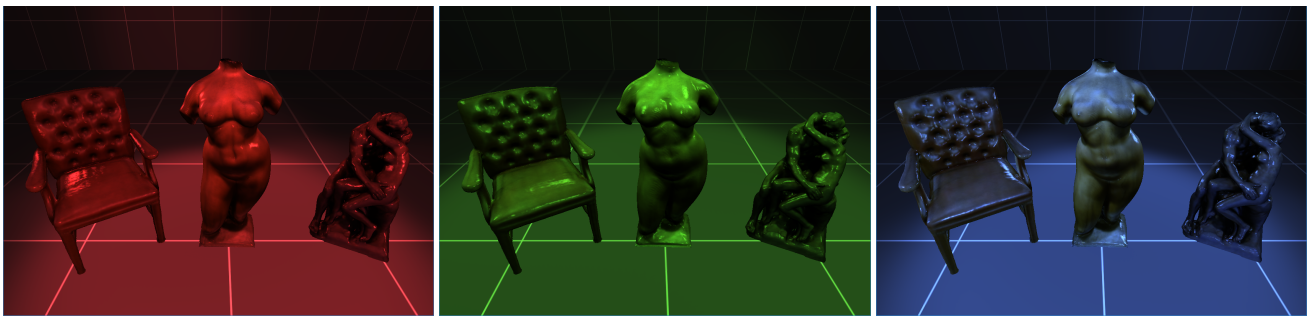
in real-time (b). To further illustrate view-dependent specular reflectance, we moved the camera to three different view positions while keeping the lighting consistent (c). Note that the specular highlights on the reconstructed objects are dynamically changing based on the camera's viewpoint. Users can freely explore the virtual reality scene without any constraints, and because objects are illuminated using virtual lights instead of image interpolation, the visual transitions between viewpoints will remain smooth.

7 CONCLUSION

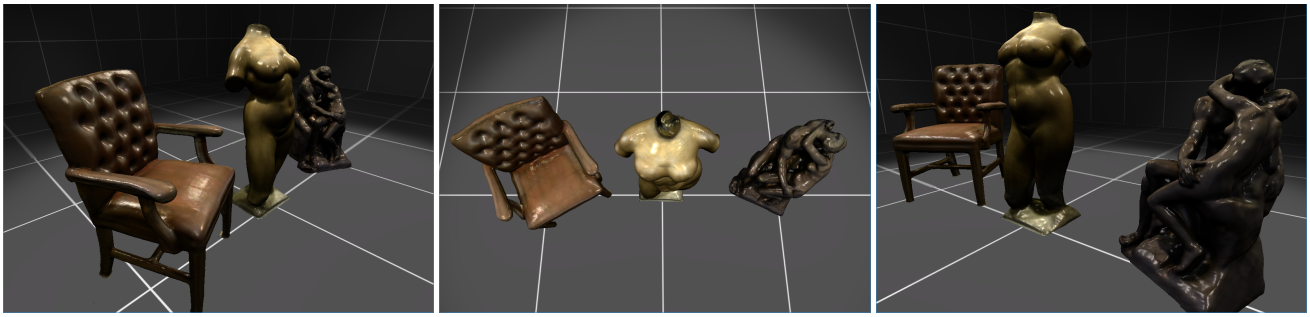
In this paper, we presented an end-to-end content creation pipeline to create dynamically relightable virtual objects from a single RGB-D sequence. First, each image from the original color stream is separated into diffuse and specular components using low-rank decomposition. The illumination and surface reflectance properties are then estimated from these maps. The reconstructed objects can be readily integrated with virtual scenes and rendered under arbitrary lighting conditions.

Limitations and Future Work. Our method maximizes the agreement of all the observed images and the estimated reflection of the virtual object. However, because we are focusing on overcoming challenges of unconstrained capture using handheld consumer-grade cameras, our approach assumes fixed real world lighting conditions, camera exposure, and white balance. We also assume that the object remains stationary during capture. Although several methods have been proposed for dealing with dynamic geometry reconstruction (e.g., [20, 26]), to-our-best-knowledge, texture/reflectance reconstruction for dynamically moving objects

(a) Rendering from the same viewpoint with different light intensities



(b) Rendering from the same viewpoint with different light directions and colors



(c) Rendering from different viewpoints under the same lighting conditions

Figure 10: Demonstration of three reconstructed virtual objects in a scene with dynamic illumination. The proposed reflectance estimation method can provide plausible results with varying virtual light direction, color, and intensity. Furthermore, the specular highlights on the object will smoothly change in real-time as the user moves between different viewpoints.

remains an unsolved problem. In the future, we would like to extend our approach in several directions. First, we would like to introduce a greater variety of real world lights (e.g, area, point sources) into our shading model. Second, our proposed method is per-vertex optimized to avoid limiting the number of materials. If the prior knowledge such as the number of material is given, we would like to segment the vertices of the object into several categories for faster modeling and rendering results. The spatial smoothness term can also be added to the objective function for more consistent results. Additionally, we used low-rank matrix recovery to decompose the diffuse and specular maps; however, other separation techniques have also been explored in prior work (e.g., [30, 35, 45, 47]). It would be worthwhile to implement and compare different methods for texture separation within this overall content creation pipeline. Finally, although we focused on virtual reality content creation in this paper, this work can also be applied to augmented reality. In the future, we are particularly excited about creating dynamically relightable objects for augmented reality that can realistically adapt to illumination in the user's surrounding environment in real-time.

ACKNOWLEDGMENTS

The authors would like to thank Daniel Keyes for his assistance in revising and proofreading the paper.

REFERENCES

[1] Armen Avetisyan, Manuel Dahnert, Angela Dai, Manolis Savva, Angel X. Chang, and Matthias Niessner. 2019. Scan2CAD: Learning CAD Model Alignment in RGB-D Scans. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[2] Sai Bi, Nima Khademi Kalantari, and Ravi Ramamoorthi. 2017. Patch-based Optimization for Image-based Texture Mapping. *ACM Trans. Graph.* 36, 4, Article 106 (jul 2017), 11 pages. https://doi.org/10.1145/3072959.3073610

[3] Mark Bolas, Ashok Kuruvilla, Shravani Chintalapudi, Fernando Rabelo, Vangelis Lympouridis, Christine Barron, Evan Suma, Catalina Matamoros, Cristina Brous, Alicja Jasina, Yawen Zheng, Andrew Jones, Paul Debevec, and David Krum. 2015. Creating Near-field VR Using Stop Motion Characters and a Touch of Light-field Rendering. In *ACM SIGGRAPH 2015 Posters* (Los Angeles, California) *(SIGGRAPH '15)*. ACM, New York, NY, USA, Article 19, 1 pages. https://doi.org/10.1145/2787626.2787640

[4] Emmanuel J. Candès, Xiaodong Li, Yi Ma, and John Wright. 2011. Robust Principal Component Analysis? *J. ACM* 58, 3, Article 11 (jun 2011), 37 pages. https://doi.org/10.1145/1970392.1970395

[5] Anpei Chen, Minye Wu, Yingliang Zhang, Nianyi Li, Jie Lu, Shenghua Gao, and Jingyi Yu. 2018. Deep surface light fields. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 1, 1 (2018), 1–17.

[6] C. Chen and E. S. Rosenberg. 2018. Dynamic Omnidirectional Texture Synthesis for Photorealistic Virtual Content Creation. In *2018 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. 85–90. https://doi.org/10.1109/ISMAR-Adjunct.2018.00040

[7] Sungjoon Choi, Qian-Yi Zhou, Stephen Miller, and Vladlen Koltun. 2016. A Large Dataset of Object Scans. *arXiv:1602.02481* (2016).

[8] Dorin Comaniciu and Peter Meer. 2002. Mean Shift: A Robust Approach Toward Feature Space Analysis. *IEEE Trans. Pattern Anal. Mach. Intell.* 24, 5 (may 2002), 603–619. https://doi.org/10.1109/34.1000236

[9] Frederique Crete, Thierry Dolmiere, Patricia Ladret, and Marina Nicolas. 2007. The blur effect: perception and estimation with a new no-reference perceptual blur metric. *Proc. SPIE* 6492 (2007), 64920I–64920I–11. https://doi.org/10.1117/12.702790

[10] Abe Davis, Marc Levoy, and Fredo Durand. 2012. Unstructured Light Fields. *Comput. Graph. Forum* 31, 2pt1 (may 2012), 305–314. https://doi.org/10.1111/j.1467-8659.2012.03009.x

[11] Paul Debevec, Yizhou Yu, and George Boshokov. 1998. *Efficient View-Dependent Image-Based Rendering with Projective Texture-Mapping*. Technical Report. University of California at Berkeley, Berkeley, CA, USA.

[12] Valentin Deschaintre, Miika Aittala, Fredo Durand, George Drettakis, and Adrien Bousseau. 2018. Single-image SVBRDF Capture with a Rendering-aware Deep Network. *ACM Trans. Graph.* 37, 4, Article 128 (July 2018), 15 pages. https://doi.org/10.1145/3197517.3201378

[13] Yue Dong, Guojun Chen, Pieter Peers, Jiawan Zhang, and Xin Tong. 2014. Appearance-from-motion: Recovering Spatially Varying Surface Reflectance Under Unknown Lighting. *ACM Trans. Graph.* 33, 6, Article 193 (Nov. 2014), 12 pages. https://doi.org/10.1145/2661229.2661283

[14] Yanping Fu, Qingan Yan, Long Yang, Jie Liao, and Chunxia Xiao. 2018. Texture Mapping for 3D Reconstruction With RGB-D Sensor. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 4645–4653.

[15] Peter Hedman, Tobias Ritschel, George Drettakis, and Gabriel Brostow. 2016. Scalable Inside-out Image-based Rendering. *ACM Trans. Graph.* 35, 6 (nov 2016), 231:1–231:11. https://doi.org/10.1145/2980179.2982420

[16] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. 2011. KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology* (Santa Barbara, California, USA) *(UIST '11)*. ACM, New York, NY, USA, 559–568. https://doi.org/10.1145/2047196.2047270

[17] Junho Jeon, Yeongyu Jung, Haejoon Kim, and Seungyong Lee. 2016. Texture map generation for 3D reconstructed scenes. *The Visual Computer* 32, 6 (01 Jun 2016), 955–965. https://doi.org/10.1007/s00371-016-1249-5

[18] Hou Ji, Angela Dai, and Matthias Nießner. 2019. 3D-SIS: 3D Semantic Instance Segmentation of RGB-D Scans. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*.

[19] S. Jiddi, P. Robert, and E. Marchand. 2016. Reflectance and Illumination Estimation for Realistic Augmentations of Real Scenes. In *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR-Adjunct)*. 244–249. https://doi.org/10.1109/ISMAR-Adjunct.2016.0085

[20] C. Kozlov, M. Slavcheva, and S. Ilic. 2018. Patch-Based Non-rigid 3D Reconstruction from a Single Depth Stream. In *2018 International Conference on 3D Vision (3DV)*. 42–51. https://doi.org/10.1109/3DV.2018.00016

[21] Marc Levoy and Pat Hanrahan. 1996. Light Field Rendering. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '96)*. ACM, New York, NY, USA, 31–42.

[22] Zhengqin Li, Zexiang Xu, Ravi Ramamoorthi, Kalyan Sunkavalli, and Manmohan Chandraker. 2018. Learning to Reconstruct Shape and Spatially-varying Reflectance from a Single Image. *ACM Trans. Graph.* 37, 6, Article 269 (dec 2018), 11 pages. https://doi.org/10.1145/3272127.3275055

[23] Jorge Lopez-Moreno, Elena Garces, Sunil Hadap, Erik Reinhard, and Diego Gutierrez. 2013. Multiple Light Source Estimation in a Single Image. *Computer Graphics Forum* 32, 8 (2013), 170–182. https://doi.org/10.1111/cgf.12195

[24] A. Meka, M. Maximov, M. Zollhöfer, A. Chatterjee, H. Seidel, C. Richardt, and C. Theobalt. 2018. LIME: Live Intrinsic Material Estimation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 6315–6324. https://doi.org/10.1109/CVPR.2018.00661

[25] Yuta Nakashima, Yusuke Uno, Norihiko Kawai, Tomokazu Sato, and Naokazu Yokoya. 2015. AR image generation using view-dependent geometry modification and texture mapping. *Virtual Reality* 19, 2 (2015), 83–94. https://doi.org/10.1007/s10055-015-0259-3

[26] R. A. Newcombe, D. Fox, and S. M. Seitz. 2015. DynamicFusion: Reconstruction and tracking of non-rigid scenes in real-time. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 343–352. https://doi.org/10.1109/CVPR.2015.7298631

[27] Jeong Joon Park, Richard Newcombe, and Steve Seitz. 2018. Surface Light Field Fusion. In *2018 International Conference on 3D Vision (3DV)*. IEEE, 12–21. https://doi.org/10.1109/3DV.2018.00013

[28] Bui Tuong Phong. 1975. Illumination for Computer Generated Pictures. *Commun. ACM* 18, 6 (jun 1975), 311–317. https://doi.org/10.1145/360825.360839

[29] Fabio Remondino and Sabry El-Hakim. 2006. Image-based 3D Modelling: A Review. *The Photogrammetric Record* 21, 115 (2006), 269–291. https://doi.org/10.1111/j.1477-9730.2006.00383.x

[30] Weihong Ren, Jiandong Tian, and Yandong Tang. 2017. Specular reflection separation with color-lines constraint. *IEEE Transactions on Image Processing* 26, 5 (2017), 2327–2337.

[31] T. Richter-Trummer, D. Kalkofen, J. Park, and D. Schmalstieg. 2016. Instant Mixed Reality Lighting from Casual Scanning. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 27–36. https://doi.org/10.1109/ISMAR.2016.18

[32] B. Shi, Z. Wu, Z. Mo, D. Duan, S. Yeung, and P. Tan. 2016. A Benchmark Dataset and Evaluation for Non-Lambertian and Uncalibrated Photometric Stereo. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 3707–3716. https://doi.org/10.1109/CVPR.2016.403

[33] Meng-Li Shih, Shih-Yang Su, Johannes Kopf, and Jia-Bin Huang. 2020. 3D Photography using Context-aware Layered Depth Inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8028–8038.

[34] Harry Shum and Sing B. Kang. 2000. A Review of image-based rendering techniques. *Proceedings of IEEE/SPIE Visual Communications and Image Processing (VCIP)* 4067 (2000), 2–13. https://doi.org/10.1117/12.386541

[35] Jinli Suo, Dongsheng An, Xiangyang Ji, Haoqian Wang, and Qionghai Dai. 2016. Fast and high quality highlight removal from a single image. *IEEE Transactions on Image Processing* 25, 11 (2016), 5441–5454.

[36] Justus Thies, Michael Zollhöfer, and Matthias Nießner. 2019. Deferred neural rendering: Image synthesis using neural textures. *ACM Transactions on Graphics (TOG)* 38, 4 (2019), 1–12.

[37] Michael Waechter, Mate Beljan, Simon Fuhrmann, Nils Moehrle, Johannes Kopf, and Michael Goesele. 2017. Virtual Rephotography: Novel View Prediction Error for 3D Reconstruction. *ACM Trans. Graph.* 36, 1, Article 8 (Jan 2017), 11 pages. https://doi.org/10.1145/2999533

[38] Michael Waechter, Nils Moehrle, and Michael Goesele. 2014. Let There Be Color! Large-Scale Texturing of 3D Reconstructions. In *Computer Vision – ECCV 2014*. Springer International Publishing, Cham, 836–850.

[39] Xing Wei, Xiaobin Xu, Jiawei Zhang, and Yihong Gong. 2018. Specular highlight reduction with known surface geometry. *Computer Vision and Image Understanding* 168 (2018), 132 – 144. https://doi.org/10.1016/j.cviu.2017.10.010 Special Issue on Vision and Computational Photography and Graphics.

[40] T. Whelan, M. Kaess, M.F. Fallon, H. Johannsson, J.J. Leonard, and J.B. McDonald. 2012. Kintinuous: Spatially Extended KinectFusion. In *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*. Sydney, Australia.

[41] John Wright, Arvind Ganesh, Shankar Rao, Yigang Peng, and Yi Ma. 2009. Robust Principal Component Analysis: Exact Recovery of Corrupted Low-Rank Matrices via Convex Optimization. In *Advances in Neural Information Processing Systems 22*, Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta (Eds.). Curran Associates, Inc., 2080–2088.

[42] H. Wu, Z. Wang, and K. Zhou. 2016. Simultaneous Localization and Appearance Estimation with a Consumer RGB-D Camera. *IEEE Transactions on Visualization and Computer Graphics* 22, 8 (Aug 2016), 2012–2023. https://doi.org/10.1109/TVCG.2015.2498617

[43] Zhe Wu, Sai-Kit Yeung, and Ping Tan. 2016. Towards Building an RGBD-M Scanner. *CoRR* abs/1603.03875 (2016). arXiv:1603.03875

[44] Zexiang Xu, Sai Bi, Kalyan Sunkavalli, Sunil Hadap, Hao Su, and Ravi Ramamoorthi. 2019. Deep View Synthesis from Sparse Photometric Images. *ACM Trans. Graph.* 38, 4, Article 76 (July 2019), 13 pages. https://doi.org/10.1145/3306346.3323007

[45] Jianwei Yang, Lixing Liu, and Stan Li. 2013. Separating specular and diffuse reflection components in the HSI color space. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 891–898.

[46] L. Yang, Q. Yan, Y. Fu, and C. Xiao. 2018. Surface Reconstruction via Fusing Sparse-Sequence of Depth Images. *IEEE Transactions on Visualization and Computer Graphics* 24, 2 (Feb 2018), 1190–1203. https://doi.org/10.1109/TVCG.2017.2657766

[47] Qingxiong Yang, Shengnan Wang, and Narendra Ahuja. 2010. Real-time specular highlight removal using bilateral filtering. In *European conference on computer vision*. Springer, 87–100.

[48] Qian-Yi Zhou and Vladlen Koltun. 2014. Color Map Optimization for 3D Reconstruction with Consumer Depth Cameras. *ACM Trans. Graph.* 33, 4, Article 155 (July 2014), 10 pages. https://doi.org/10.1145/2601097.2601134

[49] M. Zollhöfer, P. Stotko, A. Görlitz, C. Theobalt, M. Nießner, R. Klein, and A. Kolb. 2018. State of the Art on 3D Reconstruction with RGB-D Cameras. *Computer Graphics Forum (Eurographics State of the Art Reports 2018)* 37, 2 (2018).