Instant Mixed Reality Lighting from Casual Scanning

Thomas Richter-Trummer[1,2]* Denis Kalkofen[1] Jinwoo Park[3] Dieter Schmalstieg[1]
[1]Graz University of Technology [2]Bongfish GmbH [3]Korea Advanced Institute of Science and Technology

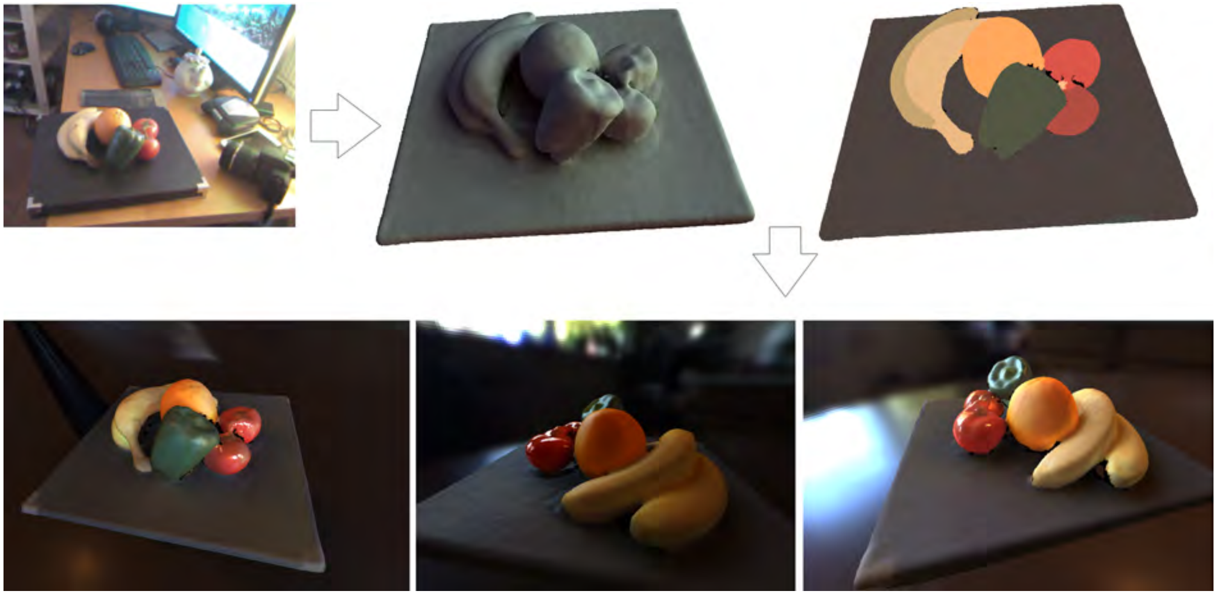


Figure 1: Our method obtains estimations for illumination and material using input from casual scans, such as obtained from a Microsoft Kinect sensor. The resulting estimations enable a variety of applications for Mixed Reality. This example shows lighting our scan using a 360° video.

Abstract

We present a method for recovering both incident lighting and surface materials from casually scanned geometry. By casual, we mean a rapid and potentially noisy scanning procedure of unmodified and uninstrumented scenes with a commodity RGB-D sensor. In other words, unlike reconstruction procedures which require careful preparations in a laboratory environment, our method works with input that can be obtained by consumer users. To ensure a robust procedure, we segment the reconstructed geometry into surfaces with homogeneous material properties and compute the radiance transfer on these segments. With this input, we solve the inverse rendering problem of factorization into lighting and material properties using an iterative optimization in spherical harmonics form. This allows us to account for self-shadowing and recover specular properties. The resulting data can be used to generate a wide range of mixed reality applications, including the rendering of synthetic objects with matching lighting into a given scene, but also re-rendering the scene (or a part of it) with new lighting. We show the robustness of our approach with real and synthetic examples under a variety of lighting conditions and compare them with ground truth data.

Index Terms: H.5.1 [Information Interfaces and Presentation]: Artificial, augmented, virtual realities; I.4.8 [Image Processing and Computer Vision]: Photometric registration—3D Reconstruction

1 Introduction

Recent advances in sensing technologies, in particular, inexpensive RGB-D cameras, have lowered the cost and effort for 3D scanning. Scanning detailed object geometry is now possible in real time, providing essential input for many applications of Mixed Reality (MR). Thus, we can safely say that *geometric* reconstruction is now sufficiently "casual" for MR. However, *photometric* reconstruction proves to be more difficult. To determine *both* material and illumination from the same set of input images, we must solve an ill-posed inverse rendering problem. Just from an image, we can not be sure which combination of light color and surface material caused a particular color observation in an image.

For this reason, most photometric reconstruction methods put additional constraints on the input. These constraints make sense for professional studio work, but cannot be afforded by casual users. For instance, consider the capabilities of a user at home wanting to play a game or shop for furniture with MR:

- The user must be able to capture unmodified environments of up to room size. Special setups, such as a green-screens, are not available.
- Capturing must work rapidly (under 5 minutes) and in one pass. The user cannot be expected to work for a long time or repeat the capturing procedure.
- Input must be obtained from a single, inexpensive camera. We used a Kinect V1, which is currently the most widespread and economical device for RGB-D sensing. Such cameras set exposure and white-balancing automatically and do not even allow to read out these settings.

*e-mail: trt@surface.at

- Multiple cameras cannot be used. A suitable rig for mounting the cameras or a trigger for camera synchronization are not consumer items.
- Controlled camera motion cannot be used. Turntables and other capturing contraptions are not consumer equipment. Even when operated by an experienced user, data collected with a handheld camera will be corrupted by noise and cover the scene with varying density.
- Controlled lighting cannot be used. This rules out calibrated light stages. Moreover, the restriction to a single capturing pass also rules out repeated capturing under multiple (uncalibrated) illumination conditions.

The main contribution of our work is a method for obtaining estimations for illumination and material using input from casual scanning. As we will point out in the discussion of related work below, most existing methods are not suitable for such a scenario. We demonstrate that even with the severe restrictions enumerated above, it is possible to deliver high-quality results. We achieve this by first segmenting the reconstructed geometry into surfaces with homogeneous material properties and computing the radiance transfer on these segments in spherical harmonics form. The inverse rendering problem is solved with an iterative least squares optimization, recovering both incident lighting and materials with diffuse and specular properties.

We demonstrate the versatility of our approach with results across the whole spectrum of photo-realistic MR applications, spanning from Augmented Reality (inserting virtual objects into real scenes) to Augmented Virtuality (inserting scanned objects into virtual scenes).

2 Related work

A large body of work on coherent rendering for MR exists [13]. Here, we focus on methods which provide an estimate of the current lighting in the user's environment and on methods which recover information about the material of the objects around the user.

Direct estimation of light. The seminal work of Debevec demonstrated how to make use of a light probe in order to directly measure the incident light [3]. Light probes are placed inside an environment and provide light measures as high-dynamic range images, which are subsequently used to compute an environment map. Active light probes obtain the environment map directly. A camera with fish-eye lens or an omni-directional camera is placed directly in the scene to acquire images of all directions in one step [11]. Passive light probes make use of a reflective object, commonly a sphere, which is placed within the scene and observed by a camera [4].

While light probes deliver environment maps at full frame rates, they are invasive and not suitable for casual use. If the lighting is assumed to be static, the environment map can be acquired with an offline scanning pass [17]. However, obtaining a full environment map with a handheld camera is significantly more work than just scanning the relevant portion of the environment. Without an artificial light probe or a full, directly scanned environment map, the incident light must be recovered by analyzing reflections in the scene, as explained in the following.

Light estimation from specular reflections. Specular reflections observed on a known object allow direct estimation of the incident light from the reflected direction. This principle can not only be applied to a dedicated light probe, but also to any specular object with known shape in the scene. For example, Lagger and Fua [14] detect specular highlights on small moving objects, and Mashita et al. [16] infer the real-world lighting by detecting specular highlights on planar objects.

Jachnik et al. [10] capture a 4D light field over a small planar surface, such as a glossy book cover. They heuristically separate the observations into diffuse and specular reflection by assuming that the diffuse reflection varies only with position, but not with the viewing direction, while specular reflection varies only with viewing direction, but not with position. Our work uses a similar factorization, but applied to a general scene with arbitrary non-planar objects having unknown material properties. Unlike Jachnik et al., we do not know in advance which objects exhibit specularities, and we must also detect self-shadowing effects.

Light estimation from diffuse reflections. A lack of specular objects can render the search for specular reflections unsuccessful. Alternatively, one can attempt to estimate illumination from diffuse reflections. However, recovering incident light from diffuse surfaces is a more difficult problem, since one must separate the contributions of incoming light from many directions. A mathematically consistent approach for computing diffuse light transport in an efficient manner is provided by spherical harmonics (SH) [20].

Gruber et al. [8] demonstrated that an SH framework is able to recover real-world lighting in real time from an RGB-D camera. They reconstruct the scene from depth images and solve for incident directional lighting in SH form from selected sample points on the reconstructed surfaces. However, the sample points must have a good distribution of surface normals. Since diffuse reflection aggregates light from all directions, shadows from other objects in the scene must be computed for every sample point. With image-space optimizations [9], their system can estimate both, dynamic incident light and shadows cast from dynamic real objects, at 20 frames per second on a desktop GPU. However, the lack of specular effects and the sparse sampling owed to the real-time constraints impose restrictions on the visual quality.

Boom et al. [1] presented a system which estimates a single point light source from arbitrary scene geometry. The approach is based on segmenting the image by color into superpixels, for which constant diffuse albedo is assumed. Knorr and Kurz [12] estimate incident light from human faces. Their method uses offline machine learning from a variety of faces under different illumination conditions. The online method applies a face tracker and matches distinctive observations points of the detected face to the training database to estimate the real-world light in SH. However, the accuracy of the face tracking is a limiting factor on the quality of the recovered illumination. Moreover, the method is limited to applications where a face is seen by the camera.

While previous work on light estimation from diffuse reflections either assumes a constant albedo color over the entire scene or requires information about the lighting, our method recovers unknown incident lighting together with unknown diffuse and specular material properties.

Material estimation. Separating light and material properties without any prior information is an ill-formed problem, because the observed surface color depends on both, the surface materials and the lighting. If we can make the simplifying assumption that surface material is diffuse and constant over the entire scene, we can estimate the incident lighting by using a linear solver based on color and geometry properties [8, 20]. However, in realistic scenes, surfaces have different material properties, leading to wrong illumination estimates, since albedo color variations are incorrectly compensated by the incident lighting reconstruction. For example, a red surface may lead to a solution involving red light coming from the direction of the plane normal.

In material estimation, a popular approach to resolve the ambiguity between lighting and reflectance is to use changing lighting, for example, using a light stage [5]. Weber et al. [25] reconstruct shape and illumination from an image stream, but require calibrated, switchable light sources. With known lighting, inverse global illumination can accurate recover materials [26].

Alternatively, objects can be moved relative to the light source. For example, a turning plate can be used to rotate small objects relative to a static light source, allowing controlled variation of the angle of incident light. Sato [21] calculates diffuse and specular properties based on a per-frame input stream, but needs a known calibrated point light and a rotating setup. Dong et al. [6] recover a highly detailed data-driven BRDF on the whole surface by rotating the object relative to the lighting, but use convex objects shapes and ignore inter-reflections and self-occlusions.

Li et al. [15] capture a scene of a moving human with multiple cameras under unknown, static lighting in a green room. By tracking with a human shape as a prior, they can use the motion of the human relative to the light sources to obtain multiple observations for every surface point over time. They segment the body into uniform materials, for which they recover a Phong BRDF. Our approach combines a similar material segmentation, but uses radiance transfer functions [19] based on self-occlusion to determine materials with only a single relationship between scene and lighting.

3 Method

Our method simultaneously estimates incident lighting and diffuse and specular material properties of the scene. As input to this inverse rendering process, we require a mesh representing the scene geometry and a set of keyframes, consisting of images of the scene taken from various known viewpoints. Both can be obtained with a handheld camera. No other knowledge about the lighting or the object's material is needed. Our method is illustrated in Figure 2. Denoting vectors in lowercase boldface, matrices in uppercase boldface, scalars in italics, and compound structures in uppercase italics, our method can be outlined as follows:

1. **Data association** (section 3.1): Reconstruct a per-vertex lit texture $\mathbf{f}(\mathbf{v}_i)$ by assigning pixels from keyframes K_j to mesh vertices $\mathbf{v}_i$, and iteratively compensate for relative exposure changes e_j and errors in the camera position $\mathbf{pos}(K_j)$ and camera orientation $\mathbf{rot}(K_j)$ associated with the keyframes K_j.

2. **Material segmentation** (section 3.2): Segment the mesh $M = \{\mathbf{v}_i\}$ into patches $M_u \subseteq M$ of vertices sharing the same material.

3. **Inverse rendering** (section 3.3): Iteratively determine the incident lighting $\mathbf{i}$ as well as diffuse albedo color $\mathbf{a}$ from $\mathbf{f}$ using the radiance transfer of M.

4. **Specular coefficients** (section 3.4): Determine estimates for the diffuse part $w_d(M_u)$ and the specular part $w_s(M_u)$ of the material for each segment M_u.

3.1 Data association

Dense reconstructions can not only be acquired with multi-camera setups [2], but, increasingly, with dense simultaneous localization and mapping (SLAM). Kinect Fusion [18], a popular SLAM variant, turns RGB-D input into a volumetric model, from which a surface mesh is extracted using isosurface raycasting. We rely on the implementation in the Microsoft Kinect SDK [1] to obtain an initial mesh geometry. We regularize the mesh to ensure distances between neighboring vertices are within a tight interval. Excessive detail is removed by iterative edge collapses, followed by local Delauney re-triangulation. Areas with large triangles are refined using Catmull-Clark subdivision. After regularization, normals must be re-estimated.

Since our method depends on the quality of the color information, we do not make use of the color information from Kinect Fusion. Instead, we greedily select a minimal set of keyframes K_j from the input sequence covering the entire scene, such that the camera centers are spaced sufficiently far apart:

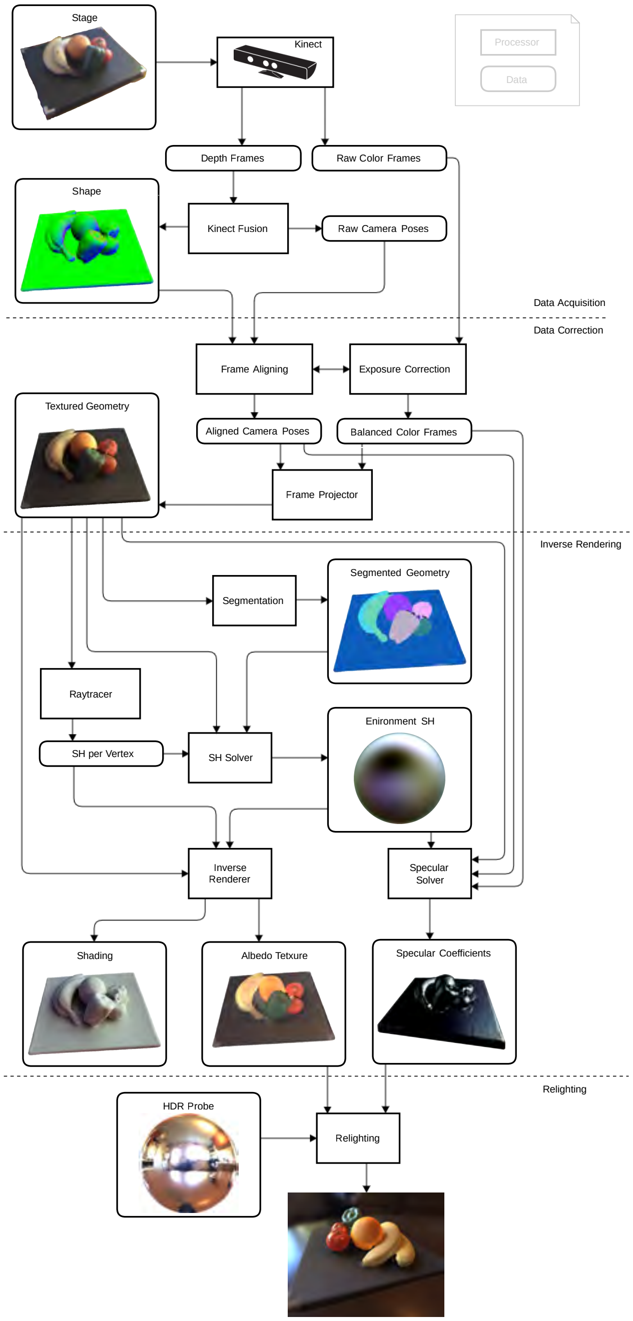


Figure 2: System overview.

$$|\mathbf{pos}(K_a) - \mathbf{pos}(K_b)| \geq c_1 \; \forall a \neq b \tag{1}$$

Initially, pixels from all keyframes are projected to the mesh vertices $\mathbf{v}_i$ using the perspective projection defined by the keyframes camera pose and collected in a per-vertex surface light field $\tilde{\mathbf{f}}(\mathbf{v}_i, K_j)$.

Before we can factor the observations into incident lighting and material properties, we must refine the data association. We do this by estimating a lit-color texture $\mathbf{f}(\mathbf{v}_i)$ from all corresponding samples $\tilde{\mathbf{f}}(\mathbf{v}_i, K_j)$. We compute $\mathbf{f}(\mathbf{v}_i)$ by searching for the y which minimizes the error of a robust estimator ρ:

[1] https://dev.windows.com/en-us/kinect

$$\mathbf{f}(\mathbf{v}_i) = \arg\min_{y} \rho(\mathbf{1}\cdot y - \tilde{\mathbf{f}}(\mathbf{v}_i, K_j)\cdot e_j) \tag{2}$$

The samples are corrected for different exposure between keyframes with weights e_j per keyframe K_j. In the first iteration, we assume $e_j = 1\ \forall j$. Given enough samples, it suffices to use the median or even the mean for ρ.

In the initial estimation of $\mathbf{f}(\mathbf{v}_i)$, we only consider samples for which the angular difference between projection direction and surface normal $\mathbf{n}(\mathbf{v}_i)$ exceeds a threshold:

$$\frac{\mathbf{v}_i - \mathbf{pos}(K_j)}{|\mathbf{v}_i - \mathbf{pos}(K_j)|}\cdot \mathbf{n}(\mathbf{v}_i) > c_2 \tag{3}$$

Moreover, vertices close to a geometric edge are initially not considered, because they can cause color bleeding. This problem can be best inspected in the image space of the keyframe K_j from which a sample $\tilde{\mathbf{f}}(\mathbf{v}_i, K_j)$ originates. For a sample from K_j corresponding to a pixel position $\mathbf{x}$, we trace rays through neighboring pixels and determine a set of intersection points M_s on the mesh. If a point in M_s is too far from $\mathbf{v}_i$ or the normals deviate too much, we do not consider $\mathbf{v}_i$:

$$\exists \mathbf{v}_s \in M_s(\mathbf{v}_i) \wedge (|\mathbf{v}_s - \mathbf{v}_i| > c_3 \vee \mathbf{n}(\mathbf{v}_s)\cdot\mathbf{n}(\mathbf{v}_i) > c_4) \tag{4}$$

For every pixel $pix(K_j, \mathbf{x})$ at location $\mathbf{x}$ in keyframe K_j, we obtain the corresponding $sample(\mathbf{f}, K_j, \mathbf{x})$ by rendering the mesh, and we compute the pixel-wise ratio of the pixel and the sample. To robustly estimate an exposure correction factor e_j for keyframe K_j with respect to the brightness represented by $\mathbf{f}$, we search for the y which minimizes a robust estimator ρ the differences among these ratios.

$$e_j = \arg\min_{y} \rho\left(\mathbf{1}\cdot y - \tfrac{pix(K_j,\mathbf{x})}{sample(\mathbf{f},K_j,\mathbf{x})}\right) \tag{5}$$

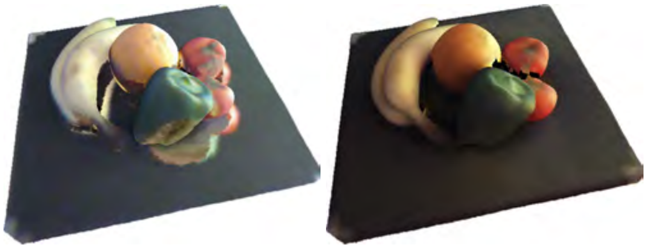
After updating $\mathbf{f}$ using the e_j as weights (Equation 2), we correct for small errors in the camera poses. We make small changes to the external camera parameters $\mathbf{pos}(K_j)$ and $\mathbf{rot}(K_j)$, re-render the images, and search for a local minimum along the image gradient using a Lucas-Kanade method.

We repeat the cycle consisting of exposure compensation and pose correction. In later iterations, the restrictions on eligible vertices can be stepwise relaxed to incorporate more data. The iteration terminates, if the overall error is small enough or no more improvements are achieved.

The result of the data association is demonstrated in Figure 3(b). Note that the images reconstructed from the lit-color texture represent averaged colors and include averaged camera-dependent lighting effects, such as specular highlights. We remove these artifacts later, after estimating specular reflection properties (see section 3.4). For comparison, Figure 3(a) shows the color reconstruction produced by Kinect Fusion, which heavily suffers from color bleeding, unstable exposures and erroneous camera poses.

3.2 Material-based segmentation

To guide the inverse rendering, we compute a material prior by segmenting the mesh M into disjoint clusters M_u, each consisting of vertices with a similar material. We begin by analyzing the mesh connectivity and isolate unconnected surface components. For every mesh component, we apply a density-based scan (DBSCAN) [7]. The algorithm searches for connected clusters of samples that have a certain density with respect to a user-defined distance function $dist$. We used a distance function that is a weighted combination of differences in the lit-color texture $\tilde{\mathbf{f}}$, the negative normal gradient distance $nngd(\mathbf{v}_a, \mathbf{v}_b)$ and the shape diameter function $sdf(\mathbf{v}_i)$.



(a) Kinect Fusion (b) Iterative Vertex Coloring

Figure 3: Comparison of per vertex colors obtained from Kinect Fusion and our approach.

$$\begin{aligned} dist(\mathbf{v}_a, \mathbf{v}_b) = {} & w_1 \cdot |\mathbf{f}(\mathbf{v}_a) - \mathbf{f}(\mathbf{v}_b)| + w_2 \cdot nngd(\mathbf{v}_a, \mathbf{v}_b) + \\ & w_3 \cdot (sdf(\mathbf{v}_a) - sdf(\mathbf{v}_b)) \end{aligned} \tag{6}$$

The lit-color texture differences describe material discontinuities. We assume low-frequency distance lighting, which does not produce hard shadows. Therefore, high-frequency color changes can only result from surface texture.

The negative normal gradient distance describes how two nearby surface points are oriented towards one another. If the surface to which the points belongs is planar or convex, we heuristically assume that they belong to the same object. If their arrangement is concave, we assume they may belong to two different objects touching each other. This measure can be computed from the angle between the normal of the first surface point and the vector to the second surface point:

$$nngd(\mathbf{v}_a, \mathbf{v}_b) = \max(0, \cos^{-1}((\mathbf{v}_a - \mathbf{v}_b)\cdot(\mathbf{n}(\mathbf{v}_a)) - \frac{\pi}{2})) \tag{7}$$

The shape diameter function models the local thickness of the object underneath a mesh vertex. It has been shown to be a reliable indicator for detecting object parts [22]. We sample it stochastically by shooting rays into the negative half-space underneath a vertex.

After clustering all mesh vertices, we merge small segments, which consist only of a few points, with their best matching neighboring cluster. An example of the resulting segmentation is shown Figure 2. Note that each segment combines surface points sharing the same material.

3.3 Inverse rendering

In this section, we introduce an inverse rendering method to factor the lit-color texture into incident lighting and albedo colors.

Let us first consider forward rendering using radiance transfer [24], which describes how light is reflected at a surface point $\mathbf{v}_i$. We write $\mathbf{illum}(\omega)$ for the light intensity from direction ω. For a purely diffuse material and only directional illumination, the radiance transfer $\mathbf{t}$ is essentially a sum over the illumination from all incoming directions, weighted by a Lambertian term and the visibility $visib$, which tells us if the light in a given direction ω is occluded or not. We sample the visibility at each vertex by tracing rays into the scene, including a few bounces. The result is scaled with the diffuse albedo color $\mathbf{a}$:

$$\mathbf{t}(\mathbf{v}_i, \mathbf{illum}) = \frac{\mathbf{a}(\mathbf{v}_i)}{\pi} \sum_{\omega} visib(\mathbf{v}_i, \omega)\cdot \mathbf{illum}(\omega) \max(\mathbf{n}(\mathbf{v}_i)\cdot\omega, 0) \tag{8}$$

For the inverse rendering, we express both the radiance transfer and the incident lighting in SH form. The radiance transfer becomes a

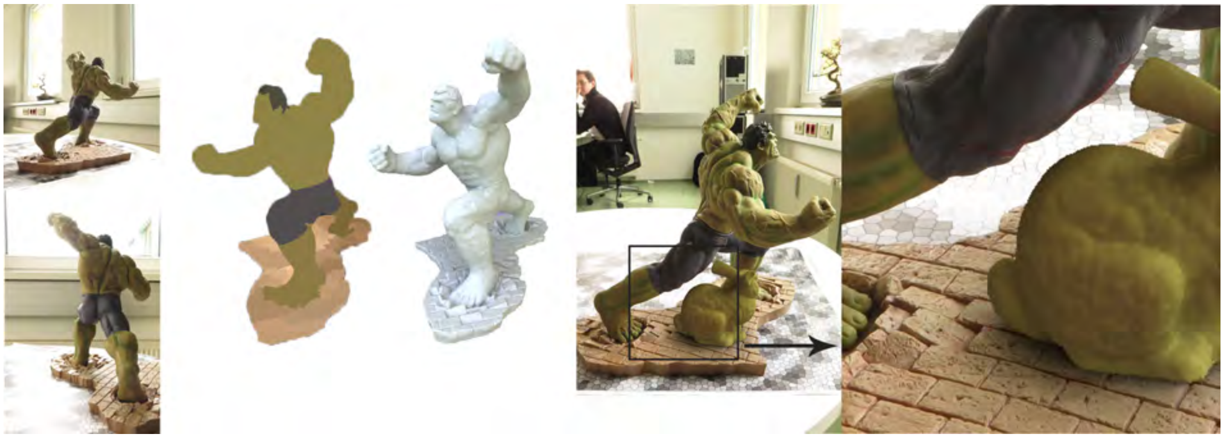


Figure 4: Visual Coherence. (left) The action figure is placed in front of two large windows, from which most of the incident light is coming. (middle) The resulting estimation of the albedo color and the rendering using the estimated lighting. (right) Differential rendering using the hulk reconstruction and a virtual bunny with the same material.

matrix $\mathbf{T} = [\mathbf{t}_{i,k}]$ (see Figure 10(a) for an example) storing the k^{th} SH-coefficient at vertex $\mathbf{v}_i$ in $\mathbf{t}_{i,k}$. The lit color $\mathbf{f}$ can be determined from the dot product of the incident lighting $\mathbf{i}$ in SH form and the radiance transfer.

$$\mathbf{f}(\mathbf{v}_i) = \mathbf{a}(\mathbf{v}_i) \sum_k \mathbf{t}_{i,k} \cdot \mathbf{i}_k \tag{9}$$

Many real world environments contain area light sources, such as ceiling lights or windows, which can be characterized as distant and low frequency. This has two important implications. First, distant lighting implies that color variations of two diffuse surfaces facing in the same direction are related to surface texture, since distant lighting has the same lighting effect on them. Therefore, different lit colors can only result from different diffuse albedo colors. Second, under low-frequency distant lighting and in the absence of occlusions, high-frequency color changes on a surface can only result from albedo colors, because the lighting does not produce hard shadows.

Taking these considerations into account, we could compute the incident lighting together with albedo colors using a non-linear solver in just one pass. However, we found that this approach tends to deliver rather unstable estimates, especially on noisy input data. Instead, we solve the resulting linear equation system in the least squares sense [8]:

$$\mathbf{i} = \arg\min_{\mathbf{i}} |\mathbf{f} - \mathbf{a} \cdot \mathbf{1}^{\mathrm{T}} \cdot \mathbf{T} \cdot \mathbf{i}|_2 \tag{10}$$

We iterate the linear solver and regularize it using the material-based segmentation. In the first iteration, the diffuse albedo colors $\mathbf{a}$ are unknown and are initialized to a constant setting (medium-intensity white). Consequently, the solver tries to derive all lit-colors from colored lighting, which leads to an error proportional to the color intensity incorrectly attributed to the lighting and not the albedo color.

The error can be estimated by comparing forward rendering and inverse rendering. We re-evaluate Equation 9 with the newly found $\mathbf{i}$ and subtract the result from $\mathbf{f}$ to obtain individual per-vertex differences $\Delta\mathbf{f}(\mathbf{v}_i)$. The differences are averaged for all vertices in one material segment M_u, yielding a per-segment error $\Delta\mathbf{f}_u$.

$$\Delta\mathbf{f}_u = \frac{1}{|M_u|} \sum_{\mathbf{v}_i \in M_u} \Delta\mathbf{f}(\mathbf{v}_i) \tag{11}$$

The per-segment error is subtracted from all vertices in the segment and added to all albedo colors in the segment:

$$\mathbf{v}_i \in M_u \rightarrow \mathbf{f}(\mathbf{v}_i) = \mathbf{f}(\mathbf{v}_i) - \Delta\mathbf{f}_u,\ \mathbf{a}(\mathbf{v}_i) = \mathbf{a}(\mathbf{v}_i) + \Delta\mathbf{f}_u \tag{12}$$

The iteration terminates, if all segment errors falls below a certain threshold. An example of the resulting incident lighting is shown as an environment map in Figure 10(b)).

After convergence, we obtain an estimate of the incident illumination $\mathbf{i}$, which is free of bias from the albedo, and an average unlit albedo color $\mathbf{a}$ per material segment.

We can derive the high-frequency parts of the albedo color by dividing the remaining lit-color texture $\mathbf{f}$ by the corresponding lighting effect. We obtain the lighting effect by forward-rendering the scene once again (Equation 9) with $\mathbf{i}$ and the constant albedo color used in the first iteration. The high-frequency part is added to the average per-segment albedo color to obtain the complete albedo color.

Finally, we improve the radiance transfer by using the reconstructed albedo color. With meaningful albedo colors, radiance transfer can take color bleeding into account.

3.4 Recovering specular coefficients

In order to recover specular material properties, we return to the original keyframes K_j. Our method is able to detect all specular reflections, if a complete light field exist for each point. However, in practice, we usually have only measurements from a fraction of all possible light directions for each point. To compensate for this sparse information, we use an approach similar to Jachnik et al. [10] and compute the specular values per material segment, rather than per vertex.

We use a material, which assumes that the the observed lit colors $\tilde{\mathbf{f}}$ are the result of a weighted sum of diffuse and specular reflection. At a given vertex, the diffuse lit color is given by $\mathbf{f}$. The purely specular color $\mathbf{f}_s$ can be determined for a given viewpoint $\mathbf{pos}(K_j)$ by sampling $\mathbf{i}$ in the direction of the reflection, unless the reflected ray is blocked by the scene. This leads to the following equation:

$$\tilde{\mathbf{f}}(\mathbf{v}_i, K_j) = w_d \cdot \mathbf{f}(\mathbf{v}_i) + w_s \cdot \mathbf{f}_s(\mathbf{v}_i, K_j) \tag{13}$$

For every segment M_u, we build and solve an overdetermined linear equation system in two unknowns $w_d(M_u)$ and $w_s(M_u)$ for all observations of a vertex $\mathbf{v}_i \in M_u$. We empirically observed that w_d corresponds well to subjective roughness of the surface material. More importantly, equation 13 can be used directly in a shader. Figure 3(d) demonstrates the resulting specularity map.

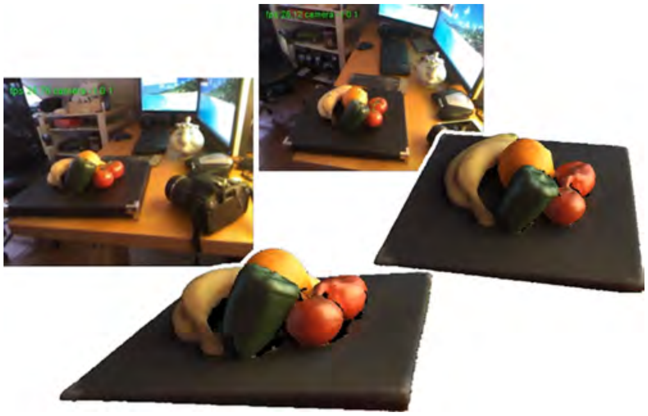


Figure 5: Visual comparsion between video frame and our rendering which uses our estimations.

4 Results

4.1 Applications

In order to illustrate our system, we have reconstructed several scenes under various light conditions. In particular, we have scanned a bowl of fruits (Figure 2), a plastic toy car (Figure 7(a)), and a action figure (Figure 4). While the action figure is mostly diffuse, the toy car and the fruits include diffuse and specular reflections.

Besides reconstructing material and geometry, our system is able to estimate the current lighting without any special laboratory setup. This enables instant addition of virtual objects to any real environment without any special hardware constraints (Figure 4). The scene consists of a real action figure (The Hulk), which we used to estimate the current lighting. In addition, we added a virtual bunny, rendered using the estimated light. Note how the two windows, the sources of most of the incident lighting, have been correctly estimated.

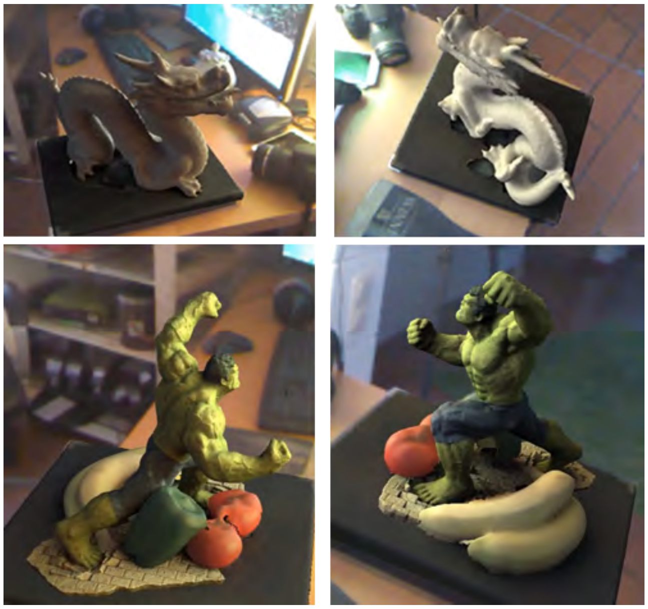


Figure 6: Mediated Reality. (top) The fruits have been replaced by a dragon. (bottom) The real orange was substituted with the reconstructed Hulk. The lighting matches the rest of the fruit scene.

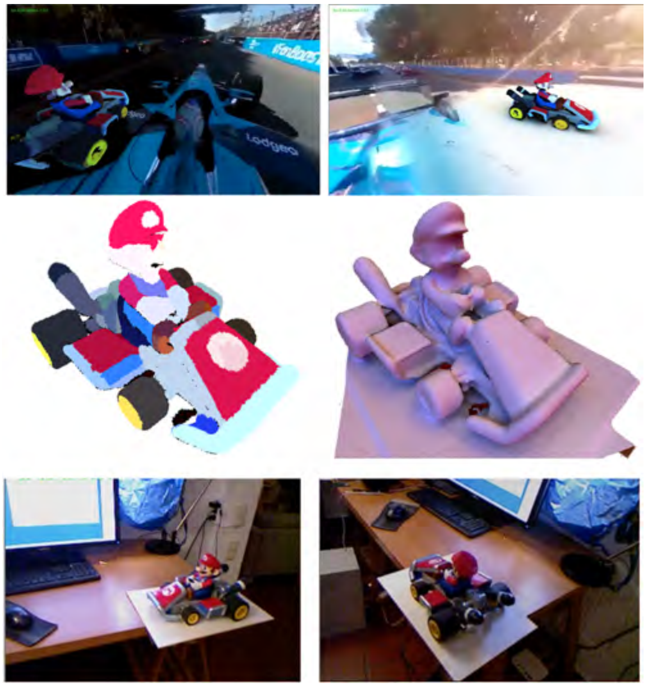


Figure 7: Augmented Virtuality. (top) We place the scan inside a 360° video and relight the geometry using our material estimated and an estimation of lighting in each video frame video. (bottom) Input data and estimated albedo color and lighting.

Our system supports coherent lighting in Diminished and Mediated Reality applications. Figure 6 (top) shows a virtual dragon replacing the fruits from Figure 2, correctly lit by the computer monitors in front of the removed fruits. The estimated light and the reconstructed material of the fruits are compared to video images in Figure 5 The sphere represents the reconstructed lighting. Since our approach reconstructs geometry, material and current lighting, real world objects from different scenes can be mixed into into into a single, coherently illuminated scene (Figure 6, bottom). Our reconstruction of material parameters allows to relight 3D scans of real objects. Thus, we can instantly add scanned real objects into virtual environments. Figure 7 (left) shows our reconstruction of a toy car, which we subsequently placed into a 360° video (Figure 7, right).

4.2 Comparisons

Figure 8 shows the results using different sets of light transfer functions applied on a single material mesh. In addition, the error compared to the reference rendering is illustrated using color coding. Figure 8(a) shows the reference scene, which has been rendered by a multi-bounce ray-tracer and lit by an HDR environment map. Figure 8(b) uses a light transfer function which is only taking local geometry information like normals into account and cannot correctly separate albedo from lighting. The reconstructed albedo color in Figure 8(b) contains artifacts from self-shadows and light bleeding, since this information is not part of the transfer function used during reconstruction. Figure 8(c) is taking self-shadow information into account. Because the light transfer function does not include light/surface reflection, the error in the albedo reconstruction is high in concave areas where light bouncing and bleeding have a noticeable impact on lighting. The recovered lighting shows less error than Figure 8(b), but still contains strong ringing and blurred highlights. The albedo texture would lead to artifacts when used for relighting. Figure 8(d) shows that GI has the least amount of er-

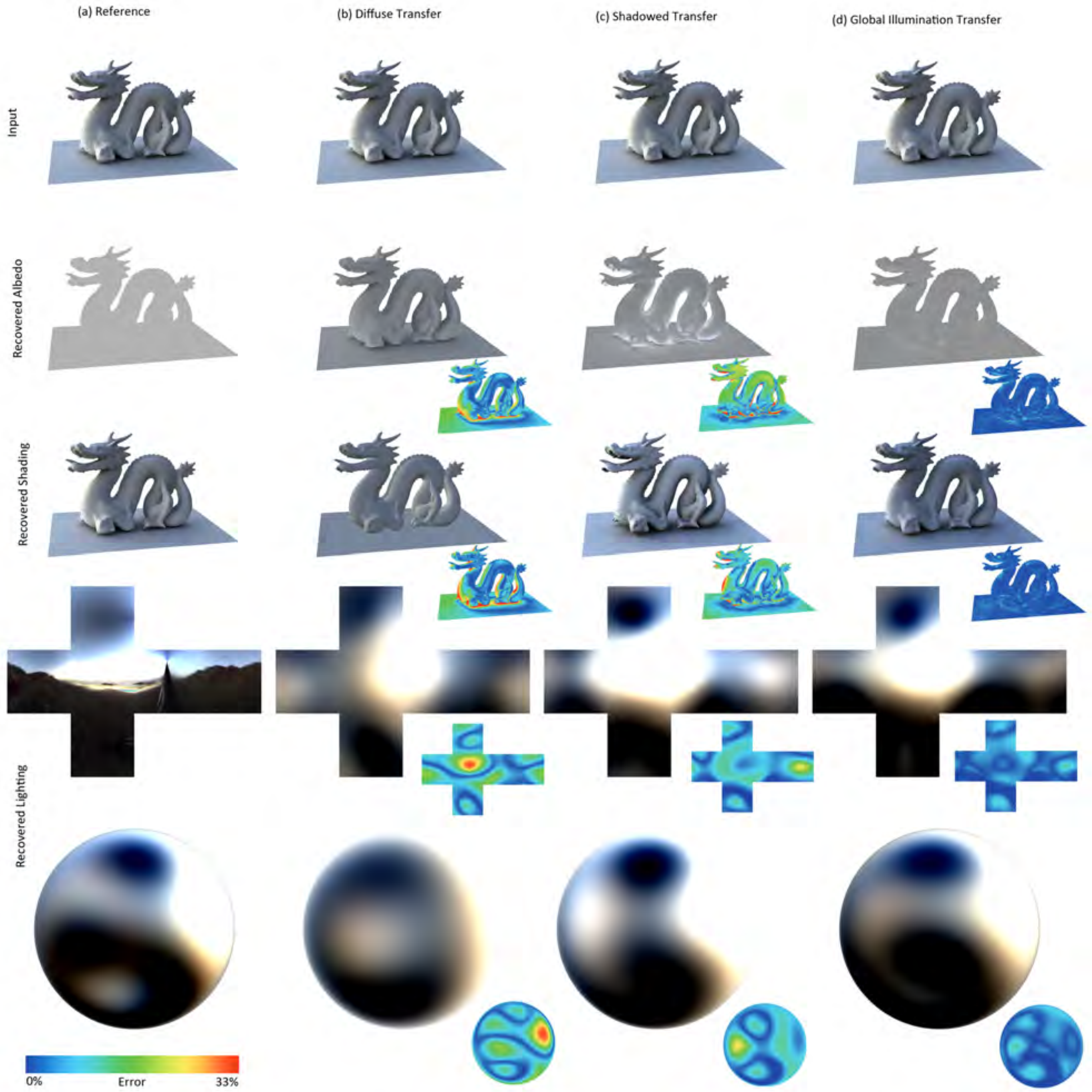


Figure 8: Visual comparison based on the type of light transfer functions used for light reconstruction. The small inset renderings present the error using the color code in the lower left corner.

ror. Note that the reference was conventionally ray-traced without lossy SH compression, we we cannot expect to match it exactly. We expect that using more SH bands would further reduce the error.

We have compared our approach to nonsegmented solving [8, 20], using GI for all approaches in the comparison. Nonsegmented solving assumes a scene with uniform albedo (Figure 9(a)). Since this approach can explain lit colors only by varying the lighting, it fails on multi-material scenes (Figure 9(b)). In contrast, our approach (Figure 9(c)) is able to segment and separate albedo and shading. Note that without GI nonsegmented solving would even perform worse.

4.3 Performance

Visually coherent rendering of virtual objects in the user's real environment requires a quick estimation, since, otherwise, the lighting may change during processing. Our method scales with the number of vertices in the mesh. The fruit scene consists of an instant geometric reconstruction of medium size received from the Kinect Fusion SDK. Estimating all parameters for this scene took approximately 2 minutes on a mid-range notebook computer. For comparison, we also used a structure-from-motion system, which is able to create a high quality 3D mesh reconstruction within a couple of minutes. We used this system on the Hulk model to create a large 3D mesh consisting of 470K vertices. On this large data set,

Table 1: ProcessingOur times in seconds.

Scene	Ball	Dragon	Bunny	Fruits	Hulk
Vertices	10371	22982	34817	72119	229059
SH(Diffuse)	0.029	0.099	0.100	0.212	0.656
SH(Shade)	0.780	4.260	3.140	7.550	32.887
SH(GI)	10.669	76.990	22.410	112.603	470.782
Segmentation	0.048	0.113	0.145	0.314	1.109
Iteration	6.333	10.090	10.970	15.993	48.341
Total	17.050	87.193	33.525	128.910	502.232

our method ran for under 1 hour to derive all parameter. However, down-sampling the mesh to 200K vertices reduced the computation time to a few minutes (see Table 1).

Table 1 lists computation times (on a mid-range notebook) in seconds for the different stages of our pipeline. For comparison with previous work, we have measured SH calculation at three complexity levels. SH (Diffuse) only takes only vertex normals into account [20], so its computation effort scales with the number of vertices only. SH (Shade) is calculated as in the approach by Gruber et al. [9], considering only ambient (self-)occlusions. The performance depends on the amount of concavity in the scene. Finally, SH (GI) takes multiple light bounces into account for calculating full global illumination (GI). This approach is dependent on the overall scene complexity.

5 Conclusion and future work

We have presented a method for obtaining estimations for illumination and material using input from casual scanning and demonstrated that, even with the severe restrictions of casual scanning, it is possible to deliver high-quality results. Moreover, we have demonstrated the versatility of our approach with results across the whole spectrum of photo-realistic MR applications, spanning from Augmented Reality (inserting virtual objects into real scenes) to Augmented Virtuality (inserting scanned objects into virtual scenes).

While our method enables many applications, it suggests several directions for future work based on the assumptions we have made. For example, we assume directional lighting with only low- to mid-range frequencies and piecewise constant specularity, which limits specular effects. Furthermore, since we assume a static scene, dynamic elements, such as moving or deformable objects require a dynamic geometric reconstruction as well as information about the light transfer for each point in time. Furthermore, we plan improved color management using LAB instead of RGB color space as proposed by Sheng et al. [23].

Acknowledgements

This work was funded in part by FFG contract 849901 (IBROC) and EU FP7-ICT contract 611526 (MAGELLAN). Also, it was supported by the National Research Foundation of Korea (NRF) grant and the BK21 plus program through NRF funded by the Korea government (MSIP) (No. 2014R1A2A2A01003005) and the Ministry of Education of Korea respectively.

References

[1] B. Boom, X. Ning, S. McDonagh, P. Sandilands, R. Fisher, and S. Orts-Escolano. Point light source estimation based on scenes recorded by a rgb-d camera. In *Proceedings of the BMVC*, 2013.

[2] A. Collet, M. Chuang, P. Sweeney, D. Gillett, D. Evseev, D. Calabrese, H. Hoppe, A. Kirk, and S. Sullivan. High-quality streamable free-viewpoint video. *ACM Trans. on Graphics*, 34(4):69:1–69:13, 2015.

[3] P. Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Proc. of SIGGRAPH*, pages 189–198, 1998.

[4] P. Debevec. Light probe image gallery. *SIGGRAPH Course*, 2003.

[5] P. Debevec, T. Hawkins, C. Tchou, H.-P. Duiker, W. Sarokin, and M. Sagar. Acquiring the reflectance field of a human face. In *Proc. of SIGGRAPH*, pages 145–156, 2000.

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

[7] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. of Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[8] L. Gruber, T. Richter-Trummer, and D. Schmalstieg. Real-time photometric registration from arbitrary geometry. In *Proc. of ISMAR*, pages 119–128, 2012.

[9] L. Gruber, J. Ventura, and D. Schmalstieg. Image-space illumination for augmented reality in dynamic environments. In *Proc. of IEEE VR*, pages 127 – 134, 2015.

[10] J. Jachnik, R. A. Newcombe, and A. J. Davison. Real-time surface light-field capture for augmentation of planar specular surfaces. In *Proc. of ISMAR*, pages 91–97, 2012.

[11] M. Knecht, C. Traxler, O. Mattausch, W. Purgathofer, and M. Wimmer. Differential instant radiosity for mixed reality. In *Proc. of ISMAR*, pages 99–107, 2010.

[12] S. B. Knorr and D. Kurz. Real-time illumination estimation from faces for coherent rendering. In *Proc. of ISMAR*, pages 349–350, 2014.

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

[14] P. Lagger and P. Fua. Using specularities to recover multiple light sources in the presence of texture. In *Proc. of ICPR*, pages 587–590, 2006.

[15] G. Li, C. Wu, C. Stoll, Y. Liu, K. Varanasi, Q. Dai, and C. Theobalt. Capturing Relightable Human Performances under General Uncontrolled Illumination. *Computer Graphics Forum*, 32:275–284, 2013.

[16] T. Mashita, H. Yasuhara, A. Plopski, K. Kiyokawa, and H. Takemura. In-situ lighting and reflectance estimations for indoor AR systems. In *Proc. of ISMAR*, pages 275–276, 2013.

[17] A. C. Maxime Meilland, Christian Barat. 3d high dynamic range dense visual slam and its application to real-time object re-lighting. *Proc. of ISMAR*, pages 143–152, 2013.

[18] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Proc. of ISMAR*, pages 127–136, 2011.

[19] R. Ramamoorthi and P. Hanrahan. An efficient representation for irradiance environment maps. *Proc. of SIGGRAPH*, pages 497–500, 2001.

[20] R. Ramamoorthi and P. Hanrahan. A signal-processing framework for inverse rendering. In *Proc. of SIGGRAPH*, pages 117–128, 2001.

[21] Y. Sato, M. D. Wheeler, and K. Ikeuchi. Object shape and reflectance modeling from observation. In *Proc. of SIGGRAPH*, pages 379–387, 1997.

[22] L. Shapira, A. Shamir, and D. Cohen-Or. Consistent mesh partitioning and skeletonisation using the shape diameter function. *The Visual Computer*, 24(4):249–259, 2008.

[23] Y. Sheng, B. Cutler, C. Chen, and J. Nasman. Perception: Perceptual global illumination cancellation in complex projection environments. *Computer Graphics Forum*, 30(4):1261–1268, 2011.

[24] P.-P. Sloan, J. Kautz, and J. Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *Proc. of SIGGRAPH*, pages 527–536, 2002.

[25] M. Weber, A. Blake, and R. Cipolla. Towards a complete dense geometric and photometric reconstruction under varying pose and illumination. *Image and Vision Computing*, pages 787–793, 2004.

[26] Y. Yu, P. Debevec, J. Malik, and T. Hawkins. Inverse global illumination: Recovering reflectance models of real scenes from photographs. In *Proc. of SIGGRAPH*, pages 215–224, 1999.

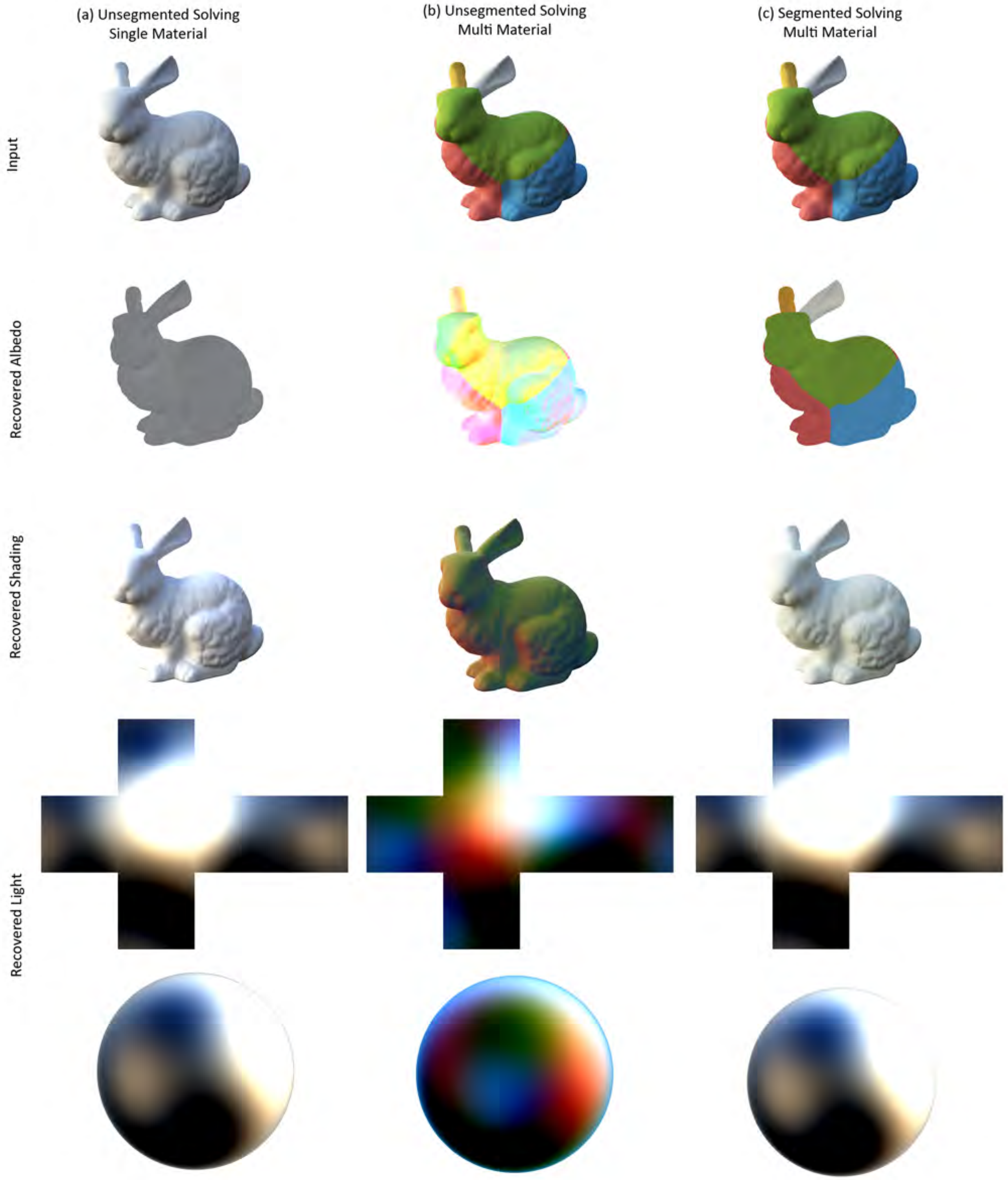
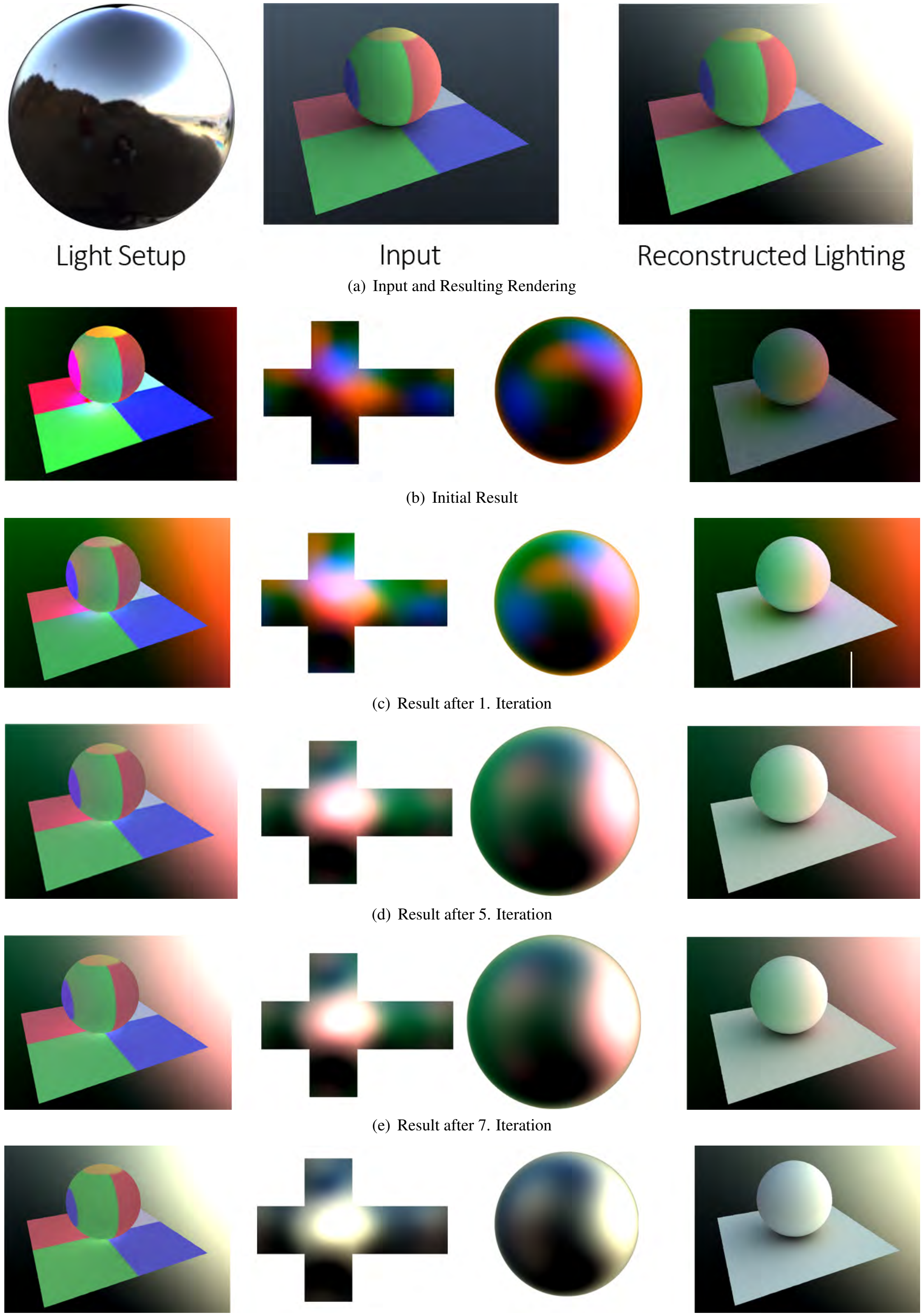


Figure 9: Comparing reconstructions based on segmentation.



(a) Input and Resulting Rendering

(b) Initial Result

(c) Result after 1. Iteration

(d) Result after 5. Iteration

(e) Result after 7. Iteration

(f) Result after 33. Iteration

Figure 10: Reconstructing Light. (left) Recovered Albedo Color (middle) Recovered Lighting (right) Geometry lit with recovered lighting.