# Emptying, Refurnishing, and Relighting Indoor Spaces

Edward Zhang[1] Michael F. Cohen[2] Brian Curless[1]

[1] University of Washington
[2] Facebook Inc.

## Abstract

Visualizing changes to indoor scenes is important for many applications. When looking for a new place to live, we want to see how the interior looks not with the current inhabitant's belongings, but with our own furniture. Before purchasing a new sofa, we want to visualize how it would look in our living room. In this paper, we present a system that takes an RGBD scan of an indoor scene and produces a scene model of the empty room, including light emitters, materials, and the geometry of the non-cluttered room. Our system enables realistic rendering not only of the empty room under the original lighting conditions, but also with various scene edits, including adding furniture, changing the material properties of the walls, and relighting. These types of scene edits enable many mixed reality applications in areas such as real estate, furniture retail, and interior design. Our system contains two novel technical contributions: a 3D radiometric calibration process that recovers the appearance of the scene in high dynamic range, and a global-illumination-aware inverse rendering framework that simultaneously recovers reflectance properties of scene surfaces and lighting properties for several light source types, including generalized point and line lights.

**Keywords:** lighting models, indoor reconstruction, diminished reality, reflectance capture, inverse lighting, inverse rendering

**Concepts:** •**Computing methodologies** → *Scene understanding; Computer vision problems; Image manipulation;* Mixed / augmented reality; Virtual reality;

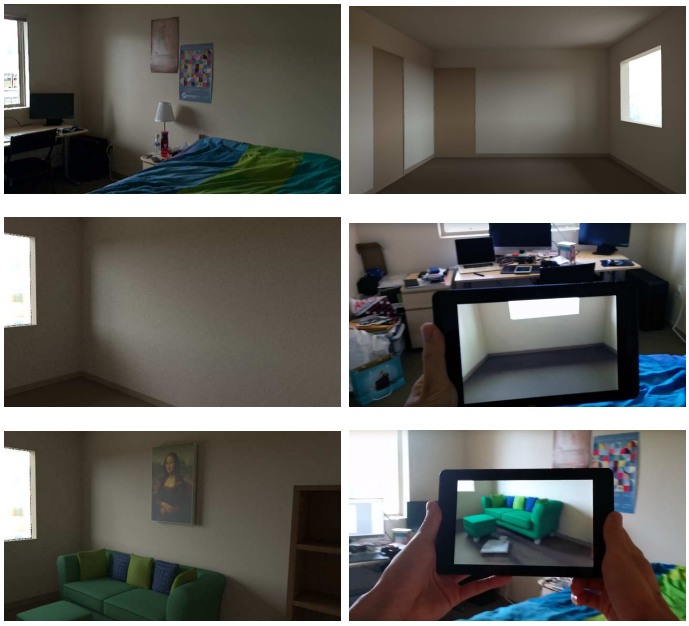


**Figure 1:** *From an RGBD scan of an indoor scene (one RGB frame shown top left), we produce a scene model of the empty room (top right), including lighting and materials. We can visualize this empty room from the original camera scan locations (middle left) or in situ (middle right). Our scene model allows for realistic scene edits, such as refurnishing the room (bottom row).*

## 1 Introduction

When visiting a prospective house or apartment to buy or rent, the spaces are often full of furniture and other clutter which makes it difficult to imagine what it might look like empty or, better yet, with our own belongings in place. We present a system that takes a sequence of RGBD images, and delivers a visual facsimile of the room devoid of all clutter. From the RGBD input we determine the room's geometric layout, the material properties of the outer surfaces (walls, floor, ceiling), a few architectural elements such as the baseboard and doors, and the light emitters. This is enough to re-render the empty room, as well as to insert furniture models and light them realistically. In the refurnished room, we can even change the colors of the walls or floor, and add, remove, or modify the lights.

To achieve this goal, we use a commodity RGBD sensor and leverage auto-exposure to recover initial 3D models with high dynamic range textures. HDR imaging is essential, as windows and directly lit surfaces can be far brighter than many other surfaces, and our analysis will require having all measurements in a single linear radiance space. Auto-exposure provides a natural mechanism for adaptive sampling across exposures for HDR recovery. We then reconstruct the walls, floor, and ceiling surfaces under a Manhattan-world assumption. Noting that these surfaces are typically largely diffuse, we solve for a constant reflectance for each using a non-linear optimization that includes indirect illumination from all observed surfaces in the scene, while simultaneously solving for unknown lighting. Our light source models include point and line lights generalized to allow for angular variation, as well distant illumination (e.g., through a window) and area lights. Our pipeline is automatic, up to a small amount of user interaction for identifying some light sources. Despite the approximations inherent in these steps, such as diffuseness and the Manhattan-world assumptions, we are able to demonstrate a variety of convincing reconstructions and edits visually consistent with the originally captured scenes.

Our primary contribution is an end-to-end system for scene model recovery. This system contains two key technical contributions:

- A method to obtain globally optimal high-dynamic-range radiances over the scene geometry, allowing us to model light propagation in the scene in a radiometrically accurate way.


SA '16 Technical Papers,, December 05 - 08, 2016, , Macao
ISBN: 978-1-4503-4514-9/16/12...$15.00
DOI: http://dx.doi.org/10.1145/2980179.2982432

- A framework for scene-scale inverse rendering that encompasses indirect illumination, occlusion, and other global lighting effects. This framework performs simultaneous recovery of diffuse reflectances and properties of light emitters; these emitters include area lights and distant illumination, as well as general models of point and line lights.

Next, we describe related work (Section 2) and then provide a system overview (Section 3) that outlines the remainder of the paper.

## 2 Background and Related Work

### 2.1 Simultaneous Localization and Mapping

Our system is heavily reliant on Simultaneous Localization and Mapping (SLAM) methods to obtain camera poses and scene geometry. Real-time volumetric fusion, introduced in [Newcombe et al. 2011], has been extended for use in larger scale scenes [Niessner et al. 2013; Whelan et al. 2012], and combined with global consistency methods [Whelan et al. 2015; Dai et al. 2016] to obtain high quality geometry.

These SLAM systems perform very well for many kinds of scenes. However, for indoor scenes, large textureless planar regions are common, and the field of view of most common RGBD sensors is not wide enough to capture these regions while still retaining tracking. Using specialized hardware, such as multiple cameras, wide field-of-view cameras, and/or IMUs, helps address these problems. [Kottas et al. 2013; Mourikis and Roumeliotis 2007] describe the localization system used in Google's Project Tango[1] device, which uses visual-inertial odometry with a wide field-of-view camera for robust localization. We use the Project Tango device in this paper for our results.

### 2.2 Indoor Scene Modelling

A number of works from Furukawa aim to model complete indoor scene geometry from photographs [Cabral and Furukawa 2014; Ikehata et al. 2015] and range data [Xiao and Furukawa 2014]. These methods generally involve Manhattan world assumptions and model outer scene structures such as walls, floor, and ceiling. These models are usually directly textured with photographs. We leverage many of the insights from these works to create our floor plans. Colburn [2013] creates more photorealistic scenes by using high dynamic range view-dependent textures for rough user-specified planar proxies, and allows large-scale scene edits, such as removing entire walls.

### 2.3 High Dynamic Range Capture

Operating in linear color space is important for many computer graphics applications. Images captured from camera sensors must be radiometrically calibrated to account for changes in exposure and for camera response.

Recovering relative exposure values for a set of images has been well studied for 2D problems such as panoramic photo stitching [Goldman 2010] and undoing video autoexposure [Kim and Pollefeys 2008; Grundmann et al. 2013]. Grossberg and Nayar [2002] describe several ambiguities that arise when computing camera responses and unknown exposures simultaneously and how to resolve them. These methods rely on point feature tracking and affine image warping, which are unsuitable for 3D captures with significant parallax. We extend these methods for 3D scenes by leveraging the scene geometry and camera poses reconstructed during scanning, enabling us to produce high dynamic range mesh textures.

Several recent works recover high-dynamic range textures for meshes by extending realtime volumetric fusion frameworks [Meilland et al. 2013; Li et al. 2016], and estimating per frame exposures. These results suffer from global consistency problems when returning to previously scanned regions. Our work on radiometric calibration is complementary to these works because we optimize for a globally consistent texture, analogous to bundle adjustment processes to handle loop closures.

### 2.4 Lighting Models

The incident light on an object is commonly approximated with the distant-scene assumption: the intensity of any incident ray on the object is dependent only on the direction of the ray, and not on the location on the surface. Thus illumination can be represented as a simple environment map, also known as Image-Based Lighting (introduced in [Debevec 1998]). This assumption has been used in most works that aim to insert synthetic objects into real scenes. The distant-scene assumption is insufficient for scene-scale mixed reality effects for two reasons: it does not capture spatial variation of lighting across the scene, and it does not allow the object to affect the appearance of the rest of the scene or participate in global illumination effects. Works such as [Unger et al. 2013] capture spatially-varying environment maps, and in general light-field capture methods such as [Wood et al. 2000] start to address the spatially-varying lighting issue. Cossairt et al. [2008] solve the object-scene interaction issue by computing light-field transfer functions, but have specialized capture requirements and limited scale. Several works use uniformly emitting point lights in addition to environment map or directional lighting [Takai et al. 2004; Stauder 2000; Weber and Cipolla 2001], but again do not consider global illumination.

### 2.5 Inverse Rendering

Physically accurate inverse rendering has a long history. However, very few attempt to simultaneously recover lighting and reflectance parameters [Patow and Pueyo 2003]. As mentioned above, most approaches work on the scale of a single object of interest, and do not consider scene-scale effects such as occlusion or global illumination, or only do so in the local region of the object (such as [Debevec 1998]). Shape-from-Shading (SfS) approaches, such as [Wu et al. 2014; Zollhöfer et al. 2015], also involve inverse rendering problems; these methods use similar assumptions (distant illumination with no occlusion or interreflection) but simultaneously recover scene geometry as well as diffuse albedo and temporally-varying lighting. Ramamoorthi and Hanrahan [2001] theoretically analyze the recovery of distant lighting and reflectance using a signal processing approach, and show when this simultaneous recovery is well-conditioned. Beyond distant illumination, several works recover point and directional light source positions simultaneously with reflectance parameters from multiple images of an object of interest [Mercier et al. 2007; Xu and Wallace 2008].

A few inverse rendering works do operate on the scene-scale. Yu et al. [1999] derive specular parameters and spatially-varying diffuse reflectances in a global-illumination-aware fashion across an entire scene, but assume known lighting. Kawai et al. [1993], using the radiosity framework, perform a nonlinear optimization over diffuse reflectance and light emitter intensity to minimize a perceptually-based cost function; for this purpose they treat scene appearance as a variable which can be eliminated, rather than optimizing to match observed appearance across the scene. In his thesis, Colburn [2014] computes light intensities and both specular and diffuse material

[1] https://www.google.com/atap/project-tango/

properties in indoor scenes represented with simplified planar proxies and view-dependent high dynamic range textures; however, the inaccurate geometry severely limits the accuracy of the solution. In addition, his system requires substantial user interaction. Lombardi and Nishino [2016a; 2016b] solve for a general BRDF and distant illumination using statistical and information-theoretical priors on a single object with known geometry. They later extend this work [2016b] to work on a set of RGBD images of a scene with spatially varying BRDFs; this is solved using an expensive gradient-based optimization where gradients are computed via path tracing, allowing for interreflections and occlusions. However, the assumption of distant illumination limits the extent of the scenes that this method can handle. Forsyth [2011] models incident illumination on an object per point on the surface as coming from a single global direction but with spatially varying intensity; while nonphysical, this captures some of the effects of interreflection. Forsyth also incorporates a known shadow map to deal with occlusion.

Other recent works aim to compute scene models from single images [Boivin and Gagalowicz 2002; Karsch et al. 2011; Karsch et al. 2014; Barron and Malik 2013]. While Boivin and Gagalowicz [2002] require known lighting, Karsch et al. [2014] recover full scene models (including light emitters, diffuse albedo, and scene geometry), automatically locating in-scene diffuse area lights, and selecting out-of-scene illumination from beyond the field of view of the image from a database of environment maps. Karsch et al. use recent advances in intrinsic image decomposition to provide reflectances, and single image depth estimation to provide geometry. Barron and Malik [2013] also take an intrinsic image decomposition approach, recovering spatially-varying illumination, diffuse albedo, and geometry from an RGBD image by using soft segmentations of the input image; each illumination segment has an independent environment map, providing spatially-varying lighting that approximates occlusion and interreflection effects. These works based on intrinsic image decomposition rely primarily on natural image and smoothness priors (e.g. [Barron and Malik 2015]) to compute albedo and thus do not give physically accurate reflectances; furthermore, they only create a scene model for a particular camera viewpoint and not an entire 3D scene.

# 3 System Overview

As input, our system takes a set of low dynamic range images of unknown exposure, the camera poses for these images, and a triangle mesh of the scene. A variety of devices and algorithms can provide this data. We use a Project Tango tablet as a capture device. Camera poses were computed by the Project Tango visual-inertial tracking system, while meshes were generated by aligning the raw depth data using the camera poses and then running Poisson Surface Reconstruction [Kazhdan and Hoppe 2013].

Next, we compute linearized radiance maps from the low dynamic range input images (Section 4). We project these radiance maps onto the mesh in order to obtain the diffuse appearance of the scene.

Separately, we identify the major architectural features of the scene, i.e. walls, floor, and ceiling, under a Manhattan World assumption (Section 5). These architectural features are used to obtain clusters of points that have the same reflectance properties and for final rerendering of the room.

We then identify locations of light emitters in the scene in a semi-automated fashion. With scene geometry, material clusters, light source positions and types, and scene appearance, we can recover lighting and diffuse reflectance parameters. This optimization process is described in Section 6.

Finally, we automatically identify other architectural features of the scene, namely doors and baseboards (Section 7). When combined with the reflectance and lighting parameters, these details enable us to rerender the scene as an empty room, to which we can then make arbitrary edits. These edits include adding furniture, repainting the walls, or changing the lighting conditions of the scene.

# 4 High Dynamic Range Meshes

We take a set of low dynamic range images, the associated camera poses for these images, and a triangle mesh of the scene. Our first step is to radiometrically align these images into a common linear radiance space. The original images are taken with automatic camera settings, which vary exposure as well as white balance.

## 4.1 Formulation

For radiometric calibration of exposure, we use similar methods to those used in panoramic photo stitching, such as the one presented by Goldman [2010]. These methods first put pixels of the original images into correspondence; while these pixels may have differing low dynamic range values, they should represent the same radiance incident on the camera. In the panorama formulation, this assumption holds because the camera centers are assumed to be coincident and therefore corresponding pixels fall along the same ray from the camera center. Then the camera exposures and the unknown scene radiances are simultaneously recovered by minimizing the reprojection error in pixel space. This method can also recover global camera parameters such as vignetting functions (as in [Goldman 2010]) and response functions (as in [Grossberg and Nayar 2003]).

In our 3D formulation, correspondences are obtained by projecting images onto the scene geometry, associating each vertex with the set of pixels projected onto it. We assume that these pixels all represent the same radiance incident on the camera by assuming a diffuse world. While camera response and vignetting parameters can be solved for as in [Grossberg and Nayar 2003; Goldman 2010], we focus on the exposure and radiance recovery and assume pre-linearized images with no vignetting. The result is a nonlinear optimization problem:

$$\min_{t_j, b_i} \sum_{i,j} (t_j b_i - X_{ij})^2 \tag{1}$$

where $t_j$ are the per-image exposures, $b_i$ are the vertex radiances, and $X_{ij}$ is the observed pixel value of vertex $i$ in image $j$.

With linear response, there is a scale ambiguity: given an optimal $(t, b)$, $(ct, b/c)$ for some constant $c$ yields the same value for the cost function. Thus we fix the exposure in the first frame of the input $t_0 = 1$ to remove this ambiguity.

Camera white-balance is modelled as per-frame, per-channel scale factors $W_j^R, W_j^G, W_j^B$. Combined with exposure, the per-channel scales are then $(t_j^R t_j^G t_j^B) = t_j(W_j^R W_j^G W_j^B)$. We solve Equation 1 for each color channel independently, giving us the final per-frame, per-channel scale factors $(t_j^R t_j^G t_j^B)$ directly.

## 4.2 Implementation

We first perform an inverse gamma correction ($\gamma = 2.2$) on the input images. We verified using a Colorchecker Chart that this was sufficient to bring the images to be close to linear in the range of lighting conditions we captured. Vignetting was not found to significantly affect Project Tango images.

After linearizing the images, we then project them onto the scene geometry: for every mesh vertex $V_i$, we trace a ray from each camera center $C_j$. If ray $\overrightarrow{C_j V_i}$ does not intersect the scene, and falls within the field of view of the camera, we locate the pixel that the ray intersects and add it to the list of pixels associated with $V_i$.

Errors in camera poses and scene geometry result in incorrectly projected pixels at depth discontinuities and texture edges. We ignore projections of pixels that fall near strong gradients in the input images, which usually coincide with depth discontinuities and texture edges.

For efficiency, we optimize over a random subset of the vertices ($N < 200,000$). For each associated pixel of each vertex, we add a data term to the cost function. The minimization is performed using Ceres Solver [Agarwal et al. ]. We found that results were not highly dependent on initialization, so we arbitrarily initialized radiances and exposures to 1.

The cost function closely resembles the one used in Structure from Motion, where, instead of camera poses and 3D point locations, we solve for camera exposures and scene point radiances. The similar sparsity patterns let us take advantage of advances in sparse bundle adjustment, as described in [Lourakis and Argyros 2009], for more efficient minimization using Schur-based methods.

### 4.3 Radiance Reprojection

In this step, we associate radiance samples with mesh vertices. For every image, we trace a ray from the camera center to each mesh vertex. If this ray does not intersect any geometry and falls within the bounds of the camera frustum, we add a radiance sample with value $\frac{X_{ij}}{t_j}$ to that vertex.

A radiance sample from camera $j$ assigned to vertex $i$ receives a weight $w_{ij}$. We set this weight to be proportional to the projected differential area of the mesh vertex onto the pixel, analogous to the differential form factor between the camera pixel and the mesh vertex [Cohen et al. 1993]. Let $g_{ij}$ be this projected area, $v_{ij}$ be the vector from the center of camera $j$ to vertex $i$, $\hat{v}_{ij} = \frac{v_{ij}}{||v_{ij}||}$, $o_j$ be the direction of the optical axis of camera $j$, and $n_i$ be the normal at vertex $i$; then

$$g_{ij} = \frac{(-\hat{v}_{ij} \cdot n_i)(\hat{v}_{ij} \cdot o_j)}{||v_{ij}||^2}. \tag{2}$$

Intuitively, the weight factor $g_{ij}$ represents how much the pixel value is affected by vertex $i$'s appearance: if a large mesh surface area projects onto the same pixel, then each vertex in that area has a correspondingly smaller impact on the value of that pixel. This weight thus favors radiance samples that are taken from head-on directions that are closer to the surface.

The total weight is also proportional to the confidence value $c$ of the radiance sample. This confidence value is based on the original 8-bit input images; saturated and underexposed pixels are less likely to be reliable. We use a hat function for the confidence value, as suggested in [Debevec and Malik 1997], but slightly modified to have nonzero confidence for a saturated pixel so that vertices only having saturated projected pixels will not be black:

$$c(X) = \begin{cases} \frac{256-X}{127}, & X > 127 \\ \frac{X}{127}, & X \leq 127 \end{cases}. \tag{3}$$

**(a)** *Input Frame 2557* **(b)** *Re-exposed Frame 2557*

**(c)** *Input Frame 2585* **(d)** *Re-exposed Frame 2585*

**(e)** *Uncalibrated Mesh Texture* **(f)** *Calibrated Mesh Texture*

**(g)** *Calibrated Mesh, Low Exposure* **(h)** *Calibrated Mesh, High Exposure*

**Figure 2:** *Comparison of office mesh, before and after radiometric calibration. (a) and (c) are two of the original input frames showing a similar area of the scene; note that (c) is significantly darker in appearance than (a) due to autoexposing for the lights. (b) and (d) are derived from (a) and (c), respectively, but rescaled to have the same exposure level. (e) shows a naive texturing of the scene geometry without any exposure correction; there are many noticeable artifacts on the wall. (f) shows the texturing after radiometric calibration (linearly tonemapped and clipped). The artifacts on the wall have been completely smoothed. (g) and (h) show another view of the mesh with calibrated texture at differing exposure values; notice how the lights are all brought within range.*

If $X_{ij}$ is the original LDR pixel value, then the total weight is

$$w_{ij} = c(X_{ij}) g_{ij}. \tag{4}$$

For this work, we assume diffuse surfaces and assign each vertex a single radiance value, which for robustness is computed as the per-channel weighted median of the vertex's radiance samples. Handling specular surfaces by analyzing the full set of radiance samples is an area for future work.

An example of the results of the radiometric calibration process is shown in Figure 2.

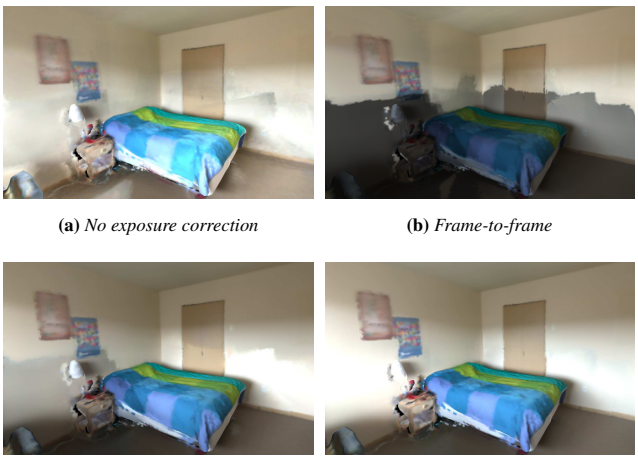
### 4.4 Evaluation

We demonstrate the importance of our globally optimal exposure estimation framework by comparing to two other methods: a frame-to-frame method, and a frame-to-model method.

The frame-to-frame baseline method is based on video radiometric calibration pipelines such as [Kim and Pollefeys 2008; Grundmann et al. 2013], which typically only compute exposure ratios between consecutive frames. In our implementation of the frame-to-frame method, dense correspondences between consecutive frames of the input are obtained via optical flow, and the relative exposure is estimated as the median ratio of corresponding pixel values. This method does not have any notion of global consistency across all frames, since it only evaluates exposures between consecutive frames.

The frame-to-model method is based on the KinectFusion-like methods proposed in [Meilland et al. 2013; Li et al. 2016]. In our implementation of the frame-to-model method, we maintain an online estimate of radiance for each vertex in the 3D model. For each frame, we first estimate the exposure by projecting pixels onto the 3D model and taking the mean ratio of pixel value to current vertex radiance. We then integrate the frame into the model by updating the average vertex radiances. As in basic KinectFusion, some global consistency is maintained but the lack of explicit loop closure handling means that drift still occurs.

For these comparisons, we avoid the difficulties of simultaneous estimation of camera response in existing frame-to-frame methods, and pose estimation in existing frame-to-model methods. Thus, we use our camera poses and prelinearized images as input to our reimplementations of the exposure estimation components of these methods. This allows a qualitative comparison of these two methods to ours by reprojecting the estimated radiances onto the scene geometry (see Section 4.3). The results are shown in Figure 3.



**(a)** *No exposure correction* **(b)** *Frame-to-frame*

**(c)** *Frame-to-model* **(d)** *Ours*

**Figure 3:** *Qualitative comparison of exposure estimation methods on Bedroom 1 dataset.*

The frame-to-frame method smooths some of the exposure variation (e.g. along the bottom of the wall to the right of the bed), but shows extensive banding and discontinuities across loop closures. The frame-to-model method is a significant improvement, but still shows some artifacts due to its lack of explicit loop closure handling. Our method gives results that, while not perfect, are smooth almost everywhere.

## 5 Floor Plan Estimation

We assume that walls, floor, and ceiling obey the Manhattan-world assumption, and furthermore that the floor and ceiling are each a single plane. While these are restrictive assumptions, they suffice for many indoor scenes. We refer to the mesh resolution, approximated by the mean edge length, as $r$.

### 5.1 Manhattan-World Coordinate System

We first determine a coordinate system that aligns the $y$ axis with the world up vector (with the floor plane at $y = 0$) and the $x, z$ axes with the Manhattan-world coordinate system for the walls. We do this by histogramming mesh vertex normals and then extracting quasi-perpendicular axis vectors, as in [Furukawa et al. 2009].

### 5.2 Floor and Ceiling

We determine the floor plane by histogramming the $y$ coordinate of all vertices with approximately vertical normals, where the width of a histogram bin is set to $r$. The first local maximum is classified as the floor plane. The ceiling plane is determined in the same manner.

### 5.3 Walls

Our floor plan determination process is similar to Cabral and Furukawa's [2014] graph-based formulation. We first project all vertices onto the floor plane and discretize into a grid. The width of a grid cell is $r$. A wall likelihood score $s$ is computed for each grid cell $G_p$ at coordinates $p = (x_p, z_p)$ and each of the four axis-aligned normal directions $n \in \{(1,0), (-1,0), (0,1), (0,-1)\}$, by examining the vertices $V$ (with coordinates $(x_V, y_V, z_V)$ and normal $n_V$) that project into the cell:

$$s(G_p, n) = \sum_{V \in G_p} \log(1 + y_V)(n_V \cdot n). \tag{5}$$

This score counts vertices with normals aligned with the axis in question, weighted by the degree to which the normal aligns with the axis as well as by the height of the vertex. Since walls are vertical, a uniformly sampled mesh of the scene should project many vertices onto the same cell. Note that the dot product term penalizes any points with normals facing away from the axis. The height-weighting is inspired by [Xiao and Furukawa 2014], which makes the observation that, while walls may be occluded at lower heights, they tend to be unoccluded near the ceiling.

We also observe that walls tend to be local bounding planes of the scene (if not globally bounding), which is not directly leveraged in prior floor plan work. If cell $G_p$ were a wall, then we would expect not only that $||G_p||$ is large, but also that the cells behind the wall (i.e. adjacent cells in the negative normal direction) are empty, and thus any points projecting into those cells are due to noise. Assume a Gaussian noise distribution on vertex coordinates, with $\sigma = \frac{r}{2}$. Due to discretization, it is possible that the directly adjacent cell $G_{p-n}$ also has a large count (the worst case being a wall at $p - \frac{n}{2}$). However, in this worst case no more than $||G_p|| \frac{1-\text{erf}(\sqrt{2})}{2} \approx 0.023||G_p||$ vertices should fall in the next cell $G_{p-2n}$ (two cells away from the original $G_p$). Thus, we set $s(G_p, n) = 0$ if $||G_{p-2n}|| > 0.023||G_p||$.

We trace scanlines of the grid, looking for contiguous sets of cells with $s(G_p, n) > N$ to extract candidate wall segments, where $N$ is the expected minimum weight of a wall cell. If the height of the room is $h$, then we would expect a perfect wall to have weight $\sum_{i=0}^{h/r} \log(1 + ir)$; assuming that for any vertical section of wall,

at least half of it is unoccluded, then we set $N = \frac{\sum_{i=0}^{h/r} \log(1+ir)}{2}$. In practice, if the wallfinding fails we decrease $N$ and repeat. The candidate wall segments are pruned using non-maximum suppression. We seek a cyclic, ordered list of these wall segment candidates. This is performed via a graph-based approach, where the endpoints of wall segment candidates are the nodes in a graph. A single endpoint of a segment can only be connected to one of the endpoints of a perpendicular segment, since the normals of the two segments must be consistently oriented. The weight of each edge is simply the Manhattan distance between the two endpoints it connects. There is an edge between the two endpoints of a wall segment candidate, which has weight $\alpha = 1$ to penalize model complexity (favoring models with fewer edges), as in [Cabral and Furukawa 2014]. We assume the longest candidate line segment will be part of the floor plan, and find the minimum cost path between its endpoints using Djikstra's algorithm.

### 5.4 Vertex Labelling and Material Clusters

After determining walls, floor, and ceiling, we assign each vertex of the mesh a label. A vertex is classified as a wall, floor, or ceiling if its vertex normal approximately matches the plane normal and it is no farther than $r$ from the plane; the vertices that do not fall in these categories are labelled as "other". We morphologically dilate and erode to close small holes in the labelled surfaces. These labelled surfaces form material clusters; all vertices with the same label are assumed to have the same material parameters.

## 6 Inverse Rendering

We now have scene geometry, diffuse appearance of the scene, and several clusters of points with the same material properties. We proceed to model the light emitters in the scene, and recover physically consistent values for the light emitter intensities and diffuse reflectances. We do this by analyzing the incident direct and indirect light on vertices in the mesh using methods similar to radiosity-based formulations of inverse lighting [Yu et al. 1999; Boivin and Gagalowicz 2002].

### 6.1 Formulation

Consider a single, non-emitting vertex $V$. We start with the rendering equation over surfaces in the scene [Kajiya 1986]:

$$L(V \to x') = \int_S f_r(x \to V \to x') G(V, x) L(x \to V) dA \quad (6)$$

where $L(a \to b)$ is the radiance along the ray from a point $a$ to a point $b$, $f_r(x \to V \to x')$ is the bidirectional reflectance distribution function (BRDF) at point $V$ giving what proportion of light coming from the ray from $x$ to $V$ is reflected towards $x'$, and $G(V, x)$ is the geometric visibility term between $V$ and $x$ including occlusion and projected area.

For a diffuse world, each vertex has a reflectance $\rho_V$ where $f_r(x \to V \to x') = \frac{\rho_V}{\pi}$, and $L(V \to x') = L_V$ is independent of the viewing direction:

$$L_V = \int_S \frac{\rho_V}{\pi} G(V, x) L(x \to V) dA. \quad (7)$$

We then decompose the integral into indirect and direct lighting components, where $\mathcal{L}$ is the set of light emitting surfaces and each $l \in \mathcal{L}$ is one of these emitting surfaces:

$$\begin{aligned} L_V = &\frac{\rho_V}{\pi} \int_{S \notin S_{\mathcal{L}}} G(V, x) L(x \to V) dA \\ &+ \frac{\rho_V}{\pi} \sum_{l \in \mathcal{L}} \int_{S \in S_l} G(V, x) L(x \to V) dA \end{aligned}. \quad (8)$$

For brevity we refer to the incident indirect lighting term as

$$L_{V,\text{indirect}} = \int_{S \notin S_{\mathcal{L}}} G(V, x) L(x \to V) dA. \quad (9)$$

For the incident direct lighting, we parameterize each light emitter by a set of intensities $I_l = (I_{l0}, ...I_{ln})^T$ such that the incident direct light on a vertex $V$ due to a light $l$ is a linear combination of these intensities. In other words, each light emitter is composed of a set of basis lights. Thus,

$$\int_{S \in S_l} G(v, x) L(x \to v) dA = F_l(V) \cdot I_l \quad (10)$$

where $F_l(V) = (F_{l0}(V), ...F_{ln}(V))^T$, and $F_{li}(V)$ is the transfer coefficient of light $l$'s $i$th basis light's radiance incident on vertex $V$ (i.e. if basis light $i$ had unit intensity, $F_{li}(V)$ would be the incident radiance on vertex $V$ due to that basis light). We also include direct lighting due to infinitesimal light sources such as point and line emitters; although they do not have areas and thus are not included in $S$, the incident lighting can still be represented as $F_l(V) \cdot I_l$. Substituting Equations 9 and 10 into Equation 8, we obtain

$$L_V = \frac{\rho_V}{\pi} \left( L_{V,\text{indirect}} + \sum_l F_l(V) \cdot I_l \right). \quad (11)$$

We calculate the coefficients of basis lights $F_l(V)$ using the scene geometry, given the locations of the light emitters (described in more detail in Section 6.3). We compute the incident indirect light directly from the final scene appearance. Using the hemicube algorithm [Cohen et al. 1993], we rasterize the mesh onto the faces of a 500 by 500 pixel cube centered at $V$ with the per-vertex median radiance values computed in Section 4.3. We compute the the total indirect irradiance at $V$ as the sum of the pixel values weighted by the hemicube form factors[2]. Note that Monte Carlo sampling methods could also be used to compute indirect illumination; given our diffuse assumption, the rasterization approach provides similar accuracy but is significantly more performant.

We have an estimate of the outgoing radiance at vertex $V$ as $\bar{L}_V$, the weighted median of projected radiances described in Section 4.3. Thus, the unknowns are the $\rho_V, I_l$. To solve for these, we formulate a constrained nonlinear optimization

$$\arg\min_{\rho_V, I_l} \sum_V d\left( \bar{L}_V - \frac{\rho_V}{\pi} \left( L_{V,\text{indirect}} + \sum_l F_l(V) \cdot I_l \right) \right) \quad (12)$$

subject to $0 \leq \rho_V < 1$ (where $d(\cdot)$ is a loss function). We perform this optimization simultaneously over several sets of vertices that

[2]Note that using the hemicube form factors given in [Cohen et al. 1993] actually results in an extra $\frac{1}{\pi}$ factor in the irradiance. For clarity, we use the standard hemicube weights multiplied by $\pi$.

share diffuse reflectances $\rho$ for robustness. Additional regularization terms for different light types are described in Section 6.3.

We use Ceres Solver [Agarwal et al. ] to perform this optimization using the Huber norm as a robust loss function. This norm helps handle outlier objects such as posters, rugs, and doors that are geometrically indistinguishable from the surfaces we classified, but have differing reflectance parameters.

## 6.2 Lighting Ambiguity

Inverse rendering problems that do not consider indirect illumination have a scale ambiguity between diffuse reflectance and light intensity; if an optimal solution were $(I, \rho)$ then $(kI, \rho/k)$ would also be an equivalent solution because the final appearance is $L_V = \frac{\rho}{\pi} F_l(V) \cdot I_l = \frac{\rho}{\pi} L_{\text{V,direct}}$.

Once indirect lighting and occlusion is taken into account, this ambiguity disappears: $L_V = \frac{\rho}{\pi}(L_{V,\text{indirect}} + L_{V,\text{direct}})$, and $L_{V,\text{indirect}}$ is a known computed quantity.

However, if $L_{V,\text{direct}}$ is much greater than $L_{V,\text{indirect}}$, the ambiguity comes back into play. This is not uncommon in real world scenes; for example, many lights throw most of their light upwards, where it bounces off the ceiling and then lights the room indirectly. For points on the ceiling near the light, the amount of incident indirect light is negligible compared to the amount of incident direct light. This results in an unstable solution.

To deal with this problem, we reweight the optimization to favor data terms for vertices that have nonnegligible amounts of incident indirect light. Some of these vertices will likely be in the same material cluster as the vertices that receive too much direct light, but prevent the diffuse reflectance estimate from varying too much.

The reweighting term we use is based on the ratio between the vertex exitant radiance and the incident indirect light on the vertex. If this ratio is $R = \frac{L_V}{L_{\text{V,indirect}}}$, then the reweighting term is for some constant $c$ (we use $c = 10$)

$$w = \begin{cases} 1, & R \leq 1 \\ \exp(c(1-R)), & R > 1 \end{cases}. \tag{13}$$

## 6.3 Light Source Models

In this work, we group lights into one of the following categories: distant environment lights, diffuse area lights, generalized point lights, and generalized line lights.

As described in Section 6.1, each of these lights is parameterized as a linear combination of basis lights, rather than parametric models (such as the Phong-like spotlight). This simplifies the structure of the problem (since now a single light can be treated as a set of independent lights) and keeps the cost function well-behaved (linear for fixed reflectances). In our system, the user must specify the number of lights and their types, although line and area lights may be suggested by our system.

In this subsection, we describe the direct lighting computation at a vertex $V$ for each light type. This includes describing how each light is decomposed into basis lights, computing the coefficients $F_{li}(V)$ for each basis light, and specifying the geometric properties of each light type (these properties are not explicitly optimized for). For conciseness, for any vector $v$, the normalized vector is represented as $\hat{v} = \frac{v}{||v||}$.

### 6.3.1 Diffuse Area Lights

Diffuse area lights are assumed to emit a constant radiance in every direction in the hemisphere and are thus simply parameterized by a single intensity (i.e. only one basis light).

To identify diffuse area lights, we threshold the vertex radiances on the mesh using a user-specified value, and find connected components of vertices with $L_V$ above this threshold. The vertices in each component of more than 50 vertices are labelled as belonging to the same area light. Some minor user interaction allows us to prune these candidate lights. Diffuse area lights are useful to model ceiling panel lights that are often found in office scenes.

To compute $F_l(V)$ for area light $l$, we simply render the hemicube using the per-vertex light labels as vertex colors, and then take the sum of the hemicube form factors that correspond to light $l$.

### 6.3.2 Distant Environment Lights

Distant environment lights come from an infinitely large sphere surrounding the scene. Rays which do not intersect the scene contribute direct illumination from the distant environment to the vertex, with a dependence only on the ray direction (and not the position of the vertex). These lights are used to represent outdoor illumination coming from a window, and approximates window illumination well assuming that no objects out the window are too near. In practice, windows are hole-filled during the Poisson Surface Reconstruction of the scene geometry. Because of this, windows are identified concurrently with area lights during the thresholding process described above. Our system suggests that an area light is in fact a window if it lies near a wall and approximately forms a rectangular shape.

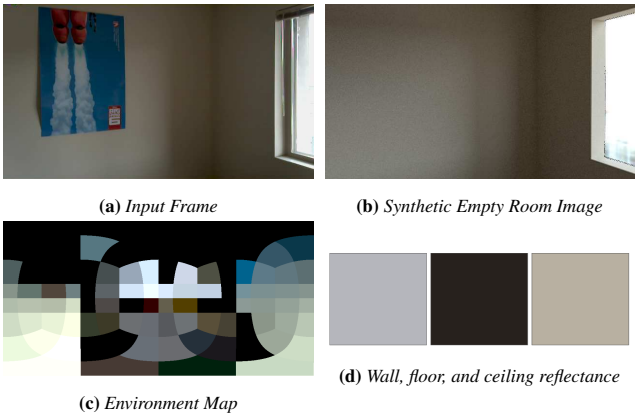
As a spherical function, distant illumination can be represented in many ways. We implemented two bases: a five-band spherical harmonic basis (with 25 light components), and a cubemapped environment map (piecewise constant basis). For the cubemap basis, we used $4 \times 4$ cubemaps, for a total of 96 component lights (i.e. 96 parameters $I$). While most distant illumination works use spherical harmonic bases, we found that for our application the piecewise constant basis was more suitable. This is primarily because the spherical harmonic lights are ill-behaved at viewpoints that were not observed in the optimization (e.g. sharp upward angles out the window); the piecewise basis makes enforcing nonnegativity and smoothness at glancing angles much simpler. We omit further discussion of the spherical harmonic basis for brevity.

To compute the set of $F_{li}(V)$ for distant illumination, we again use the hemicube. For hemicube pixels that correspond to distant illumination rays, we compute the orientation of the ray through the center of the pixel, determine which cubemap cell $j$ that direction corresponds to, and then increment $F_{lj}(V)$ by the hemicube pixel form factor.

For piecewise constant basis distant illumination, we constrain light intensities to be nonnegative. We also add a weak smoothness term between adjacent cells of the cubemap. In particular, cubemap cells that are not incident on any vertices in the optimization will still have reasonable values. This is important because these directions may be sampled when rerendering the scene. If $A = \{(i,j) | i, j \text{ are adjacent cubemap cells}\}$, then we add a set of regularization terms to the cost function ($\lambda_{\text{smooth}} = 1 \times 10^{-5}$):

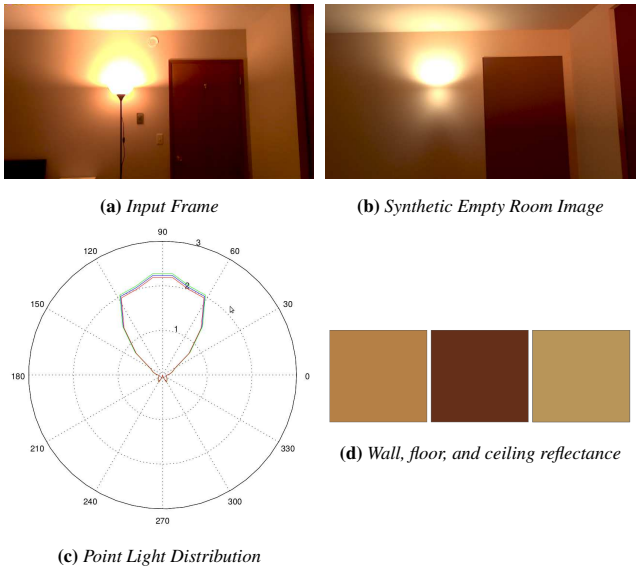
$$\lambda_{\text{smooth}} \sum_{(i,j) \in A} (I_{li} - I_{lj})^2.$$

An example of our solved distant lighting is shown in Figure 4.



**(a)** *Input Frame* **(b)** *Synthetic Empty Room Image* **(c)** *Environment Map* **(d)** *Wall, floor, and ceiling reflectance*

**Figure 4:** *Recovered distant illumination parameters. (a) is an input image from the capture sequence. (b) shows a rerendered version with our distant lighting model. (c) shows a $(\theta, \phi)$ parameterization of the recovered cubemap; note that the central region corresponds to the directions out the window. While this approximation does not appear representative of the actual appearance out the window and only imprecisely matches shadow boundaries, it captures most of the illumination effects that affect the indoor scene. (d) shows the recovered wall, floor, and ceiling reflectances.In the actual scene, the wall and ceiling have identical properties; the solved reflectances are very similar even though we did not constrain them to be the same in our system.*




**(a)** *Input Frame* **(b)** *Synthetic Empty Room Image* **(c)** *Point Light Distribution* **(d)** *Wall, floor, and ceiling reflectance*

**Figure 5:** *Recovered point light parameters. (a) is an input image from the capture sequence. (b) shows a rerendered version with our point light model. (c) shows a goniometric diagram of the recovered point light distribution using a piecewise constant basis. In rendered images, the distribution is interpolated using a Catmull-Rom spline. (d) shows the recovered wall, floor, and ceiling reflectances. In the actual scene, the wall and ceiling have identical properties; the solved reflectances are very similar even though we did not constrain them to be the same in our system. Note that this scene is also illuminated by a window.*

#### 6.3.3 Generalized Point Lights

Many light manufacturers describe the intensity distribution of their lights using an IES (Illumination Engineering Society) lighting profile, as specified in [Subcommittee on Photometry of the IESNA Computer Committee 2002]. Light intensity emitted from a point source $I(\phi, \theta)$ is a spherical function, and the IES format specifies the values of this function at a discrete set of exitant angles (much like our piecewise constant representation of distant lighting). The IES format also builds in strong symmetry assumptions, most commonly assuming radially symmetric light distributions. These profiles are used to specify many types of lights found in residential indoor scenes, such as lamps with shades, standing lights, recessed ceiling lights, and general spotlights, and are commonly used in architectural visualization. Thus we follow the example of the IES and model point lights as radially symmetric emitters.

Such point lights have two geometric parameters: a 3D location $P$, and an axis of symmetry $\hat{v}$. We default to a vertical axis of symmetry $\hat{v} = (0, 1, 0)$, since this is how most radially symmetric lights in IES formats are used. The positions of point lights must be manually specified in our system.

This radially symmetric distribution means the spherical function $I(\phi, \theta)$ (centered around $\hat{v}$) is independent of $\phi$, the azimuthal angle. To represent this 1D function in a linear basis, we can use either the Fourier basis or a discretized basis (analogously to environment maps). In this work, we use a 32-bin discretization of angle $\theta$, the inclination angle.

As these point lights are infinitesimal, their contribution to the direct lighting cannot be determined using the hemicube. Therefore we separately compute the incident direct lighting on every vertex due to these point lights. The incident lighting basis coefficient due to a point light using the piecewise constant parameterization is

$$F_{li}(V) = G(V, P)\frac{(\hat{a} \cdot n_V)}{||a||^2} \tag{14}$$

where $a = P - V$, $n_V$ is the surface normal at $V$, and $G(V, P)$ is a visibility term. The visibility term is computed by checking if a ray between $V$ and $P$ intersects the scene geometry. $i$ corresponds to the angle subdivision cell for direction $\theta = \cos^{-1}(\hat{a} \cdot \hat{v})$ (and $F_{lj}(V) = 0$ if $j \neq i$).

Constraints and regularization for point lights are analogous to distant lighting (nonnegativity and smoothness). However, we found that adding the additional constraint of monochromaticity created better results. To achieve this, we give each point light a three-component color, and let the intensity of this color vary around the light with the (now single-channel) basis light intensities,

$$\left(I_l^R(\theta), I_l^G(\theta), I_l^B(\theta)\right) = (r_l, g_l, b_l)\, I_l(\theta),$$

giving the light three additional parameters $r_l, g_l, b_l$. We then add a term strongly ($\lambda_{\text{monochrome}} = 1 \times 10^8$) enforcing the color to have unit norm:

$$\lambda_{\text{monochrome}}(r_l^2 + g_l^2 + b_l^2 - 1).$$

The monochromaticity constraint couples the color channels, so our optimization actually minimizes data terms for all three channels simultaneously when monochromatic lights are included.

An example of our solved point lighting is shown in Figure 5.

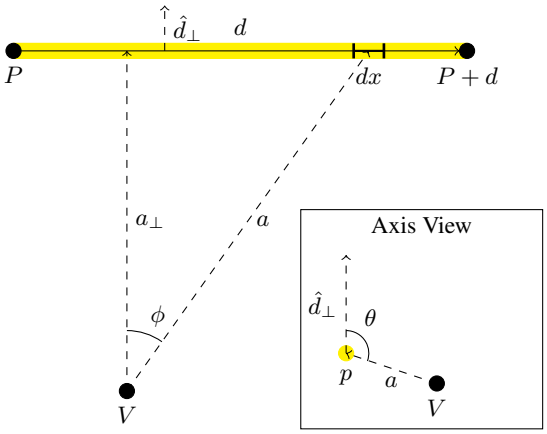



**Figure 6:** *Side view of line light, perpendicular to axis d of light. Inset shows view along the axis d of the light*

#### 6.3.4 Generalized Line Lights

Fluorescent tube lights are commonly encountered in non-residential indoor scenes. While uniformly emitting line lights have been analyzed in the literature, they are unsuitable for modelling these tube lights. These lights often have baffles that change the directional variation of light intensity.

Figure 6 illustrates how we parameterize line lights. Geometrically, generalized linear lights have the following properties: an endpoint $P$, a direction vector $d$ (so that the line extends from $P$ to $P+d$), and a perpendicular vector $\hat{d}_\perp$. The light distribution of the line light varies around the light with $\theta$, but does not vary in the direction parallel to $d$.

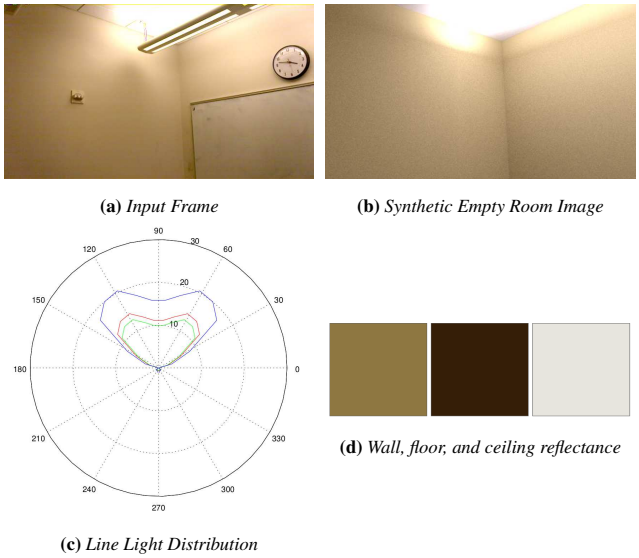
Analogously to environment maps and point lights, line lights can use the Fourier basis or a constant basis based on discretized $\theta$ as component lights. In this work, as for distant and point illumination, we use the piecewise constant local basis, uniformly dividing the circle of directions into 32 subdivisions.

The equation for the incident lighting basis coefficient at a vertex $V$ due to a line light is given by integrating over the length of the light; each differential line segment is treated similarly to a point light, except with an extra term accounting for the projected length of the segment onto $V$:

$$F_{li}(V) = \int_0^{||d||} G(V, P + x\hat{d}) \frac{n_V \cdot \hat{a}}{||a||^2} (\hat{a} \cdot \hat{a}_\perp) dx \qquad (15)$$

where $a = (P + x\hat{d}) - V$, the vector from $V$ to the differential line segment $dx$, $a_\perp = a - (a \cdot \hat{d})\hat{d}$ is the component of $a$ perpendicular to the light, and $G(V, P + x\hat{d})$ is the visibility term as described for point lights. The $(\hat{a} \cdot \hat{a}_\perp)dx = \cos(\phi)dx$ term represents the projected length of the segment. Figure 6 visually shows these quantities relative to the light and $V$.

While $\phi$ varies along the length of the light, $\theta$ is constant for a particular vertex $V$ relative to any point along the light. Since we use a piecewise constant basis, each vertex will thus only have one of the basis light directions incident on it, so only one of the $F_{li}$ will be nonzero. We compute which $i$ this is by finding $\theta = \cos^{-1}(\hat{a}_\perp \cdot \hat{d}_\perp)$. Note that this expression for $\theta$ conveniently enforces symmetry around $\hat{d}_\perp$, which is common for real world line lights (where $\hat{d}_\perp$ is the world up vector). However, for full generality, $\theta$ would range from 0 to $2\pi$.




**(a)** *Input Frame* **(b)** *Synthetic Empty Room Image*

**(c)** *Line Light Distribution* **(d)** *Wall, floor, and ceiling reflectance*

**Figure 7:** *Recovered line light parameters. (a) is an input image from the capture sequence. (b) shows a rerendered version with our line light model. (c) shows a goniometric diagram of the recovered line light distribution using a piecewise constant basis. In rendered images, the distribution is interpolated using a Catmull-Rom spline. (d) shows the recovered wall, floor, and ceiling reflectances.*

The integral in Equation 15 could be analytically solved in the absence of occlusion; however the visibility term makes this considerably more difficult. We discretize the line light into $N$ segments for the purposes of computing occlusion, so this becomes

$$F_{li}(V) = \sum_{k=1}^{N} G(V, P + (k - 0.5)\hat{d}) \frac{n_V \cdot \hat{a}}{||a||^2} (\hat{a} \cdot \hat{a}_\perp) \frac{||d||}{N}. \qquad (16)$$

Generally RGBD scans of fluorescent tube lights will pick up the surface of the light (or at least its housing and reflectors). Thus, we can identify line lights by looking for long, thin mesh components that are located near the ceiling. We do this by examining the eigenvectors and eigenvalues of the matrix of coordinates of the component's vertices. If the second and third eigenvalues are small relative to the first, then the component is likely to be a line light, for which the largest eigenvector provides $\frac{d}{||d||}$. We determine the extent of the line light by transforming the vertex coordinates into the basis formed by the eigenvectors and finding the maximum and minimum coefficients for the principal eigenvector. The perpendicular vector $\hat{d}_\perp$ is set to be the world up vector (more precisely, the component of the up vector perpendicular to $d$). Other non-axis aligned line lights must be added manually.

Regularization for line lights is analogous to the regularization on point lights: nonnegativity and smoothness between adjacent cells in the constant basis, and monochromaticity across the light.

An example of our solved line lighting is shown in Figure 7.

### 6.4 Missing Data

In the formulation above, we assume that the hemicube provides an accurate measurement of the direct and indirect lighting at a point. However, in real world captures, many vertices or regions of the mesh will be missing data – they will have no radiance estimates. For example, it is impossible to scan the top of a tall shelf or a light fixture.

We first determine what proportion of the hemicube consists of unobserved incident radiance values. We then assume that, on average, the incident indirect light coming from unobserved rays will be the same as the average indirect incident light from the rest of the scene. Thus, if $p_0$ is the proportion of the hemisphere consisting of unobserved radiances, and $p_i$ is the proportion of the hemisphere consisting of indirect illumination, we rescale the indirect illumination as

$$L'_{\text{indirect}} = L_{\text{indirect}} \frac{p_0 + p_i}{p_i}. \tag{17}$$

## 7 Architectural Features

We identified doors and baseboard as common elements of scenes that are easy to identify automatically, simple to model, and important for the perception of the scene.

### 7.1 Manhattan-World Analysis

Most architectural features, not only baseboards and doors but also windows, light switches, and outlets, are located on walls (possibly recessed or protruding), and are axis-aligned and rectangular. Due to inaccuracies in camera poses and geometry, finding rectangles in reprojected mesh space is difficult. Instead, we locate axis-aligned edges in individual input images and then analyze the edge responses in 3D mesh space.

First, we transform the camera poses into the axis-aligned coordinate system described in Section 5, and then compute the coordinates of the three vanishing points for each image. We then run the oriented edge filter used in [Furukawa et al. 2009] to obtain oriented edge images for each camera viewpoint. We use the wall, floor, and ceiling labels to identify pixels that project onto walls (rather than occluders) and only consider edge responses for these pixels.

### 7.2 Baseboards

Baseboards are primarily parameterized by a height, although they may also have a depth (i.e. they may protrude from the wall). We locate a baseboard by assuming that there should be a horizontal edge on all observed walls at the same height.

We first discretize each wall in the room into a 2D grid of cells, and then project the masked oriented edge images onto these grids. Each cell will record how many pixels project onto it $N(x, y)$, as well as the total horizontal edge response of pixels projecting onto it $F(x, y)$. We then define a function $B(y)$ to measure the likelihood of the baseboard height being at $y$

$$B(y) = \frac{1}{\sum_x \text{nonzero}(N(x, y))} \sum_x \frac{F(x, y)}{N(x, y)}$$

where $\text{nonzero}(x) = 1$ if $x > 0$ and 0 otherwise. Note that when $N(x, y) = 0$, that grid cell is occluded for all views. We take the first local maximum for $B$ as the baseboard height.

The motivation for this function is as follows: $\frac{F(x,y)}{N(x,y)}$ is the average horizontal edge weight in a cell. The $\frac{1}{\sum_x \text{nonzero}(N(x,y))}$ term accounts for occlusions: $\sum_x \text{nonzero}(N(x, y))$ is a count of how many wall cells at height $y$ were observed. If only a few cells at a particular height were visible, but they all had strong edge responses, it is still likely to be a baseboard candidate and that height should not be unfairly penalized.

### 7.3 Doors

We assume doors to be axis-aligned rectangles on walls that extend from the floor to a height above 1.5 meters. They are parameterized by: the index of the wall they are on $i$, the height $h$, the position of the center of the door $p$, and the width $w$.

Similarly to the baseboard case, we project the masked oriented edge image into the discretized walls. We then trace scanlines of the grid of responses to extract candidate line segments and prune the candidates using non-maximum suppression. Note that when tracing line segments, we allow line segments to extend into unobserved cells to account for occlusion.

For each wall, we iterate through all pairs of vertical line segments on that wall. If both segments start near the ground and end near the same height greater than 1.5 meters, and there exists a horizontal line segment at approximately that height that covers at least the extent of the door, then that defines a candidate door. If several candidate doors overlap, we take the widest candidate to be the true door.

### 7.4 Appearance of Minor Architectural Features

For each feature, we also compute a diffuse reflectance and texture. This is done analogously to the process described in Section 6.1, except that we fix the lighting parameters to the values we obtained from the first optimization. This amounts to finding the (robustified) average reflectance across all vertices in the feature.

## 8 Results and Discussion

We demonstrate our full inverse rendering system on four scenes with varying lighting conditions, including three different types of emitters: distant illumination, generalized point lights, and generalized line lights. We show results from these scenes, including the relit empty room as well as virtually refurnished rooms, in Figures 1 and 8. All renderings are performed with PBRT [Pharr and Humphreys 2004] for physically accurate global illumination effects. For scenes with distant illumination through windows, we render the scene using the recovered environment map as a light source. For the final image, we composite in the environment map derived from the original input images to render what is seen through the window.

The unoptimized runtimes of various components in our system running on a 3.4GHz Intel i7 processor are as follows: ray-traced image reprojection, which is used in several places in our pipeline, takes approximately 5 minutes for 1000-1200 images at $640 \times 360$ resolution and a mesh of approximately 200,000 vertices. The optimization process for radiometric calibration takes approximately 15 minutes for these images. Floor plan recovery takes about 5 seconds. The inverse lighting optimization process ranges from 15 to 30 minutes depending on the light types present in the scene.

As mentioned in Section 6.3, our system requires some user assistance to specify light source positions. For point lights in Bedroom 2 and Play Room, light coordinates were manually specified. The axis of symmetry was left as the default world up vector. In the

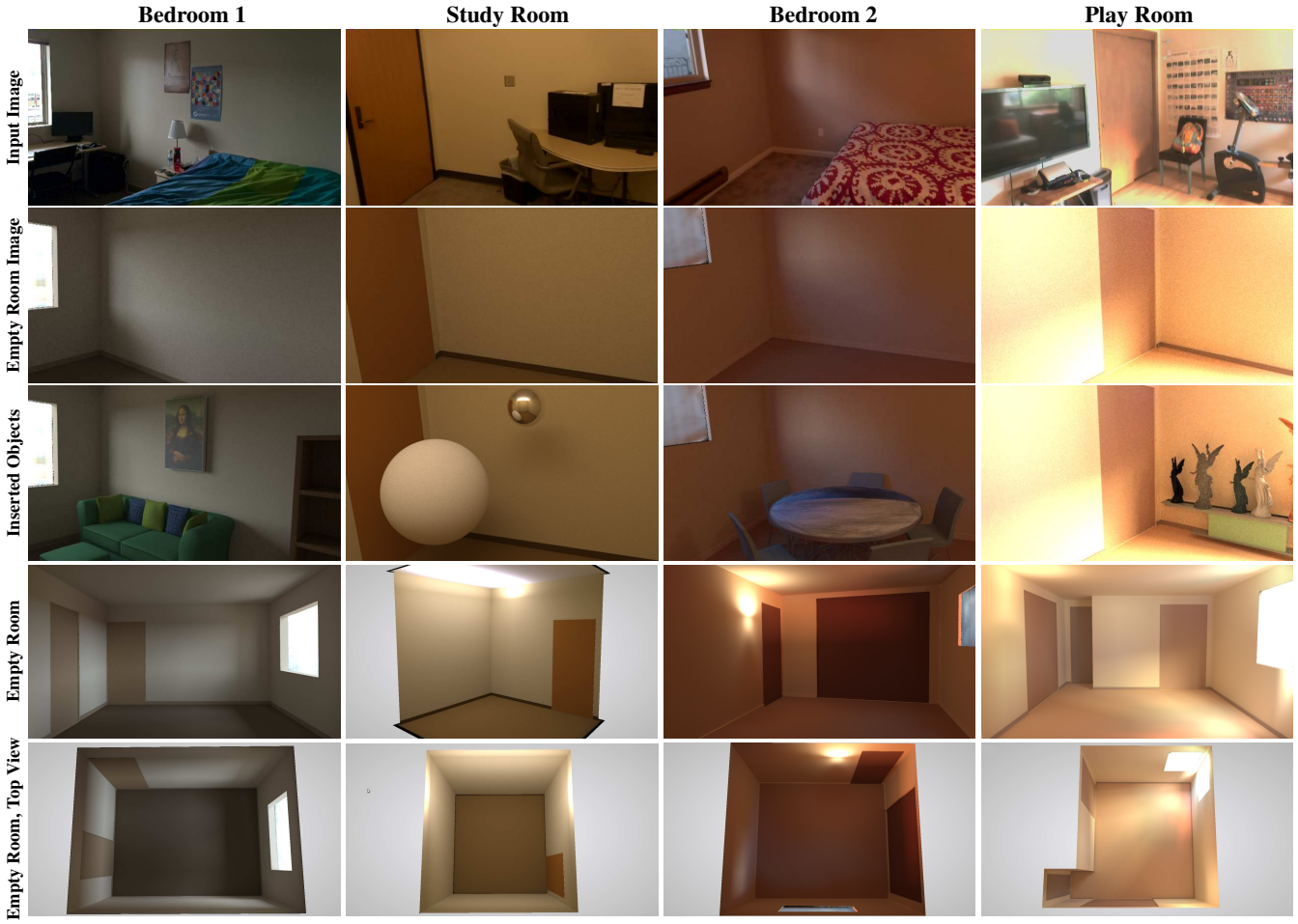



**Figure 8:** *Results from our system in four scenes. The first row is an unadjusted frame from the original input. The second row shows a synthetic rendering of the empty room from the same camera viewpoint and the same exposure. The third row inserts some synthetic objects into the scene. The fourth and fifth rows show wider views of the entire relit mesh with baked global illumination at a suitable global exposure.*

Bedroom 1 scene, regions of high specularity on the desk that reflected the window light were labelled as light emitters based on the threshold. These were manually pruned. Similarly, in Bedroom 2, bright regions near the point light were pruned from being light sources. The Study Room's line light was automatically detected and modelled. The geometry of the corner windows in the Play Room was not captured by the sensor and subsequent Poisson Surface Reconstruction. We manually adjusted the geometry of these windows. Automating these steps is an area for future work.

In addition to adding virtual objects to a scene, we also demonstrate performing other edits using our recovered, decluttered scene model. These edits include changing the illumination conditions in the scene and changing the material properties of surfaces in the scene (Figure 9).

Our system enables interactive mixed reality applications, where we can actually walk through the scene and visualize our edits in situ (Figure 1). Carrying the Tango tablet in the scene, we can localize its position and orientation. We then simply render the view of the empty room or room filled with new furniture as it would look like from that vantage point. These are rendered as static meshes with baked global illumination computed using PBRT, while the Project Tango software ensures that the camera pose is correct relative to the original scene.

### 8.1 Limitations

Our results show that our system handles a number of scenes and leads to convincing rerendered results. However, our work has several limitations that suggest future directions for research. Firstly, our framework assumes diffuse surfaces. It is fairly straightforward, albeit computationally intensive, to extend our framework into non-diffuse surfaces by rendering incident light hemicubes using view-dependent texture mapping, and optimizing over each observation of each vertex rather than over the per-vertex weighted median. However, preliminary experiments suggest that significantly more complete data captures, with views of surfaces from many angles, are necessary for this optimization to accurately recover specular properties. In addition to specular surfaces, our system also does not currently handle spatially varying reflectances; we plan to extend our system to synthesize floor or wall textures.

Having an accurate estimate of the indirect light in a scene is important to our inverse rendering framework. Incomplete scans reduce the accuracy of our estimate. To determine the importance of having complete scans, we conducted an experiment where we artificially removed sections of one of our datasets and examined the effects on the resulting reflectance estimates.

We deleted random connected regions of the mesh, each comprising 5% of the total triangle count, by selecting a random triangle

**Figure 9:** *This figure shows how our scene model enables edits to the lighting and materials of the scene. These edits are shown in increments; the first column shows the input frame, while the second shows the refurnished room. The third column shows the walls repainted a different color. The fourth column changes the lighting conditions, such as changing the time of day to sunset (top), or moving the point light illuminating the room (bottom).*

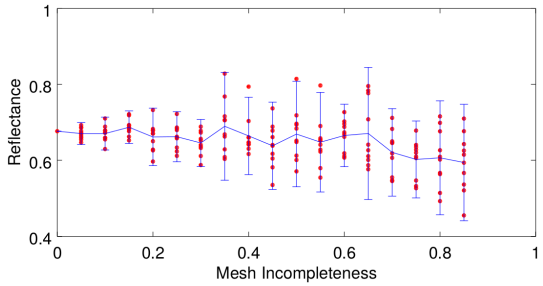



**Figure 10:** *Reflectance estimates in the Bedroom 1 dataset as a function of mesh incompleteness, with 95% confidence intervals.*

and deleting adjacent triangles in a breadth-first fashion. For each increment of mesh incompleteness between 0% and 90%, we generated 10 meshes randomly ablated in this fashion. We then ran our inverse rendering pipeline on each mesh, and plotted the resulting reflectance estimates in Figure 10. We can see that the variance in reflectance estimates rises dramatically after about 20% of the mesh is deleted. There also appears to be a slight bias toward lower reflectances as mesh incompleteness increases; this bias is likely to be scene-dependent as a result of our averaging scheme to fill in missing data (see Section 6.4).

In this paper we do not show results of directly editing the original room contents. We explicitly recover reflectances of only wall, floor, and ceiling vertices simultaneously with lighting. While we can directly recover per-vertex reflectances on the original mesh (Equation 12) using the solved lighting parameters (shown in Figure 11), the resulting reflectances are often inconsistent or nonphysical ($\rho_V > 1$). This is primarily due to the relatively low resolution and accuracy of the input mesh and camera poses, resulting in incorrect occlusion boundaries, incorrect normals, and subsequent incorrect calculations of incident lighting, as shown in Figure 11. Modest mesh quality and inaccurate poses also create artifacts in the HDR diffuse scene appearance. Furthermore, the low resolution of the input data results in unconvincing rerenderings. Combining existing SLAM algorithms with better hardware to handle typical indoor scenes more robustly will greatly benefit our system.

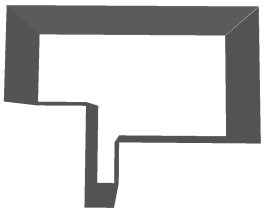
Our floorplan estimation algorithm is designed for single closed rooms. On more complex scenes, such as extending the "Playroom" dataset into the adjacent hallway and bedroom (see Figure 12), we see difficulties in reconstructing thin walls and doorways. However, with some manual adjustment, we obtain a reasonable floor plan. As our inverse lighting component is independent of the reconstructed floorplan, we still obtain plausible results.

**Figure 11:** *We compute a "reflectance mesh" by taking the radiances from the input mesh (inset) and the solved lighting, and compute a per-vertex reflectance as radiance divided by incident light. We expect that the walls will have a uniform reflectance, but some artifacts are visible. The red circles show regions where the inaccurate wall geometry results in inaccurate normals. The blue circles show regions where the inaccurate scene geometry results in incorrectly projected scene radiance. The yellow circles show regions where the inaccurate scene geometry results in incorrect occlusion boundaries. The green circles show artifacts from the low resolution of the lighting. The large white spot on the desk (circled in purple) is due to the specularity of the desk; the radiance given by the input mesh is an overestimate of the diffuse appearance of the desk, and thus the computed reflectance is also too high. Although not an issue in this case, this specularity can also cause an overestimate in the the incident indirect illumination on other points in the scene.*

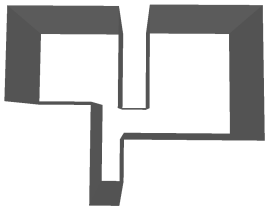
While our recovered lighting matches the original appearance of the scene, it may not represent the true emitters in the scene accurately. However, [Karsch et al. 2011; Ramanarayanan et al. 2007] present evidence that a range of lighting conditions are vi-

**(a)** *Input data*

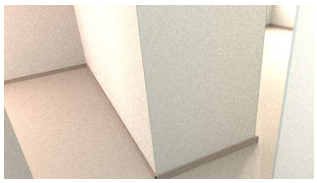
**(b)** *Floorplan with default parameters*



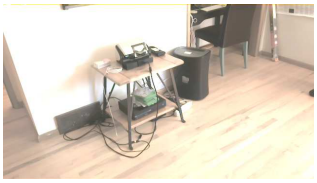
**(c)** *Floorplan with increased N*



**(d)** *Floorplan with tweaked geometry*



**(e)** *Both rooms, no doorways*



**(f)** *Original input image*

**(g)** *Rerendered empty room image*

**Figure 12:** *Results of a multiroom dataset. (a) shows a top view of the input scan. (b) shows the reconstructed floorplan with default parameters, which only includes one room. (c) adjusts parameters such that doorways do not get detected as walls, but the thin wall on the side of the circled region of (a) still does not get reconstructed. With some manual adjustment of the circled region, we arrive at a reasonable floorplan (d). Although this floorplan does not include the doorways, it still gives reasonable results when rerendered in (e) and (g); some incorrect shadow boundaries can be seen near the doorway in (g) due to the missing wall around the doorframe.*

sually similar to humans, suggesting that rough lighting estimates are enough for many applications. Furthermore, Ramamoorthi and Hanrahan [2001] determine that it is impossible to recover true high-frequency lighting from diffuse object appearance.

Automatically determining the locations and classification of light sources is perhaps the most interesting problem. It is difficult to directly optimize for light source positions because occlusion boundaries cause discontinuities in the solution space. It is especially challenging to compute the properties of unobserved light sources, or to identify e.g. that the reflection of an unseen light source in a mirror is not itself an emitter. We believe that using machine learning methods for light source identification and classification will be an important addition to these inverse rendering problems.

## Acknowledgements

We would like to thank Sameer Agarwal for his advice on optimization and Ceres Solver. We also thank Pratheba Selvaraju for her assistance in modelling the contents of refurnished scenes. These contents include 3D models from CGTrader (users scopia, den_krasik, buchak72, belgrade_sim, peter_janov) and Turbosquid (user shop3ds). Several additional 3D models were obtained from the Stanford Computer Graphics Laboratory.

This work was supported by the NSF/Intel Visual and Experimental Computing Award #1538618, with additional support from Google, Microsoft, Pixar, and the University of Washington Animation Research Labs.

## References

AGARWAL, S., MIERLE, K., AND OTHERS. Ceres Solver.

BARRON, J. T., AND MALIK, J. 2013. Intrinsic scene properties from a single rgb-d image. In *CVPR*.

BARRON, J. T., AND MALIK, J. 2015. Shape, illumination, and reflectance from shading. *PAMI 37*, 8.

BOIVIN, S., AND GAGALOWICZ, A. 2002. Inverse rendering from a single image. *Conference on Colour in Graphics, Imaging, and Vision 2002*, 1.

CABRAL, R., AND FURUKAWA, Y. 2014. Piecewise Planar and Compact Floorplan Reconstruction from Images. In *CVPR*.

COHEN, M. F., WALLACE, J., AND HANRAHAN, P. 1993. *Radiosity and realistic image synthesis*. Academic Press Professional.

COLBURN, A., AGARWALA, A., HERTZMANN, A., CURLESS, B., AND COHEN, M. F. 2013. Image-based remodeling. *TVCG 19*, 1.

COLBURN, R. A. 2014. *Image-Based Remodeling: A Framework for Creating, Visualizing, and Editing Image-Based Models*. PhD thesis, University of Washington.

COSSAIRT, O., NAYAR, S., AND RAMAMOORTHI, R. 2008. Light field transfer: global illumination between real and synthetic objects. *ACM Trans. Graph. 27*, 3.

DAI, A., NIESSNER, M., ZOLLHÖFER, M., IZADI, S., AND THEOBALT, C. 2016. BundleFusion: Real-time Globally Consistent 3D Reconstruction using On-the-fly Surface Reintegration. arXiv:1604.01093.

DEBEVEC, P. E., AND MALIK, J. 1997. Recovering high dynamic range radiance maps from photographs. In *SIGGRAPH*.

DEBEVEC, P. 1998. Rendering synthetic objects into real scenes. In *SIGGRAPH*.

FORSYTH, D. A. 2011. Variable-source shading analysis. *IJCV 91*, 3.

FURUKAWA, Y., CURLESS, B., SEITZ, S., AND SZELISKI, R. 2009. Manhattan-world stereo. In *CVPR*.

GOLDMAN, D. B. 2010. Vignette and exposure calibration and compensation. *PAMI 32*, 12.

GROSSBERG, M. D., AND NAYAR, S. K. 2002. What Can Be Known about the Radiometric Response from Images? In *ECCV*.

GROSSBERG, M., AND NAYAR, S. 2003. What is the space of camera response functions? In *CVPR*.

GRUNDMANN, M., MCCLANAHAN, C., AND ESSA, I. 2013. Post-processing approach for radiometric self-calibration of video. In *ICCP*.

IKEHATA, S., YANG, H., AND FURUKAWA, Y. 2015. Structured indoor modelling. In *ICCV*.

KAJIYA, J. T. 1986. The rendering equation. In *SIGGRAPH*.

KARSCH, K., HEDAU, V., FORSYTH, D., AND HOIEM, D. 2011. Rendering synthetic objects into legacy photographs. *ACM Trans. Graph. 30*, 6.

KARSCH, K., SUNKAVALLI, K., HADAP, S., CARR, N., JIN, H., FONTE, R., SITTIG, M., AND FORSYTH, D. 2014. Automatic Scene Inference for 3D Object Compositing. *ACM Trans. Graph. 33*, 3.

KAWAI, J. K., PAINTER, J. S., AND COHEN, M. F. 1993. Radioptimization. In *SIGGRAPH*.

KAZHDAN, M., AND HOPPE, H. 2013. Screened poisson surface reconstruction. *ACM Trans. Graph. 32*, 3.

KIM, S. J., AND POLLEFEYS, M. 2008. Robust radiometric calibration and vignetting correction. *PAMI 30*, 4.

KOTTAS, D. G., HESCH, J. A., BOWMAN, S. L., AND ROUMELIOTIS, S. I. 2013. On the consistency of vision-aided inertial navigation. In *Experimental Robotics*, Springer, 303–317.

LI, S., HANDA, A., ZHANG, Y., AND CALWAY, A. 2016. HDRFusion: HDR SLAM using a low-cost auto-exposure RGB-D sensor. arXiv:1604.00895.

LOMBARDI, S., AND NISHINO, K. 2016. Reflectance and Illumination Recovery in the Wild. *PAMI 38*, 1.

LOMBARDI, S., AND NISHINO, K. 2016. Radiometric Scene Decomposition: Scene Reflectance, Illumination, and Geometry from RGB-D Images. arXiv:1604.01354.

LOURAKIS, M. I. A., AND ARGYROS, A. A. 2009. SBA. *ACM Transactions on Mathematical Software 36*, 1.

MEILLAND, M., BARAT, C., AND COMPORT, A. 2013. 3D High Dynamic Range dense visual SLAM and its application to real-time object re-lighting. In *ISMAR*.

MERCIER, B., MENEVEAUX, D., AND FOURNIER, A. 2007. A framework for automatically recovering object shape, reflectance and light sources from calibrated images. *IJCV 73*, 1.

MOURIKIS, A. I., AND ROUMELIOTIS, S. I. 2007. A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation. In *ICRA*.

NEWCOMBE, R. A., DAVISON, A. J., IZADI, S., KOHLI, P., HILLIGES, O., SHOTTON, J., MOLYNEAUX, D., HODGES, S., KIM, D., AND FITZGIBBON, A. 2011. KinectFusion: Real-time dense surface mapping and tracking. In *ISMAR*.

NIESSNER, M., ZOLLHÖFER, M., IZADI, S., AND STAMMINGER, M. 2013. Real-time 3D Reconstruction at Scale Using Voxel Hashing. *ACM Trans. Graph. 32*, 6.

PATOW, G., AND PUEYO, X. 2003. A Survey of Inverse Rendering Problems. *Computer Graphics Forum 22*, 4.

PHARR, M., AND HUMPHREYS, G. 2004. *Physically Based Rendering: From Theory To Implementation*. Morgan Kaufmann.

RAMAMOORTHI, R., AND HANRAHAN, P. 2001. A signal-processing framework for inverse rendering. In *SIGGRAPH*.

RAMANARAYANAN, G., FERWERDA, J., WALTER, B., AND BALA, K. 2007. Visual equivalence. *ACM Trans. Graph. 26*, 3.

STAUDER, J. 2000. Point light source estimation from two images and its limits. *IJCV 36*, 3.

SUBCOMMITTEE ON PHOTOMETRY OF THE IESNA COMPUTER COMMITTEE. 2002. Iesna standard file format for the electronic transfer of photometric data and related information. Tech. Rep. LM-63-02.

TAKAI, T., NIINUMA, K., MAKI, A., AND MATSUYAMA, T. 2004. Difference sphere: an approach to near light source estimation. In *CVPR*.

UNGER, J., KRONANDER, J., LARSSON, P., GUSTAVSON, S., LÖW, J., AND YNNERMAN, A. 2013. Spatially varying image based lighting using HDR-video. *Computers & Graphics 37*, 7.

WEBER, M., AND CIPOLLA, R. 2001. A practical method for estimation of point light-sources. *BMVC 2*.

WHELAN, T., KAESS, M., FALLON, M., JOHANNSSON, H., LEONARD, J., AND MCDONALD, J. 2012. Kintinuous: Spatially Extended KinectFusion. Tech. rep., MIT CSAIL.

WHELAN, T., LEUTENEGGER, S., MORENO, R. S., GLOCKER, B., AND DAVISON, A. 2015. ElasticFusion: Dense SLAM Without A Pose Graph. *Robotics: Science and Systems 11*.

WOOD, D. N., AZUMA, D. I., ALDINGER, K., CURLESS, B., DUCHAMP, T., SALESIN, D. H., AND STUETZLE, W. 2000. Surface light fields for 3D photography. In *SIGGRAPH*.

WU, C., ZOLLHÖFER, M., NIESSNER, M., STAMMINGER, M., IZADI, S., AND THEOBALT, C. 2014. Real-time shading-based refinement for consumer depth cameras. *ACM Trans. Graph. 33*, 6.

XIAO, J., AND FURUKAWA, Y. 2014. Reconstructing the World's Museums. *IJCV 110*, 3.

XU, S., AND WALLACE, A. M. 2008. Recovering surface reflectance and multiple light locations and intensities from image data. *Pattern Recogn. Lett. 29*, 11.

YU, Y., DEBEVEC, P., MALIK, J., AND HAWKINS, T. 1999. Inverse global illumination. In *SIGGRAPH*.

ZOLLHÖFER, M., DAI, A., INNMANN, M., WU, C., STAMMINGER, M., THEOBALT, C., AND NIESSNER, M. 2015. Shading-based refinement on volumetric signed distance functions. *ACM Trans. Graph. 34*, 4.