Automatic Scene Inference for 3D Object Compositing

Kevin Karsch[1], Kalyan Sunkavalli[2], Sunil Hadap[2],
Nathan Carr[2], Hailin Jin[2], Rafael Fonte[1],
Michael Sittig[1] David Forsyth[1]
[1]University of Illinois
[2]Adobe Research

We present a user-friendly image editing system that supports a drag-and-drop object insertion (where the user merely drags objects into the image, and the system automatically places them in 3D and relights them appropriately), post-process illumination editing, and depth-of-field manipulation. Underlying our system is a fully automatic technique for recovering a comprehensive 3D scene model (geometry, illumination, diffuse albedo and camera parameters) from a single, low dynamic range photograph. This is made possible by two novel contributions: an illumination inference algorithm that recovers a full lighting model of the scene (including light sources that are not directly visible in the photograph), and a depth estimation algorithm that combines data-driven depth transfer with geometric reasoning about the scene layout. A user study shows that our system produces perceptually convincing results, and achieves the same level of realism as techniques that require significant user interaction.

Categories and Subject Descriptors: I.2.10 [**Computing Methodologies**]: Artificial Intelligence—*Vision and Scene Understanding*; I.3.6 [**Computing Methodologies**]: Computer Graphics—*Methodology and Techniques*

Additional Key Words and Phrases: Illumination inference, depth estimation, scene reconstruction, physically grounded, image-based rendering, image-based editing

1. INTRODUCTION

Many applications require a user to insert 3D characters, props, or other synthetic objects into images. In many existing photo editors, it is the artist's job to create photorealistic effects by recognizing the physical space present in an image. For example, to add a new object into an image, the artist must determine how the object will be lit, where shadows will be cast, and the perspective at which the object will be viewed. In this paper, we demonstrate a new kind of image editor – one that computes the physical space of the photo-

Authors' email addresses: {karsch1,sittig2,dafonte2,daf}@illinois.edu, {sunkaval,hadap,ncarr,hljin}@adobe.com

DOI 10.1145/1559755.1559763
http://doi.acm.org/10.1145/1559755.1559763

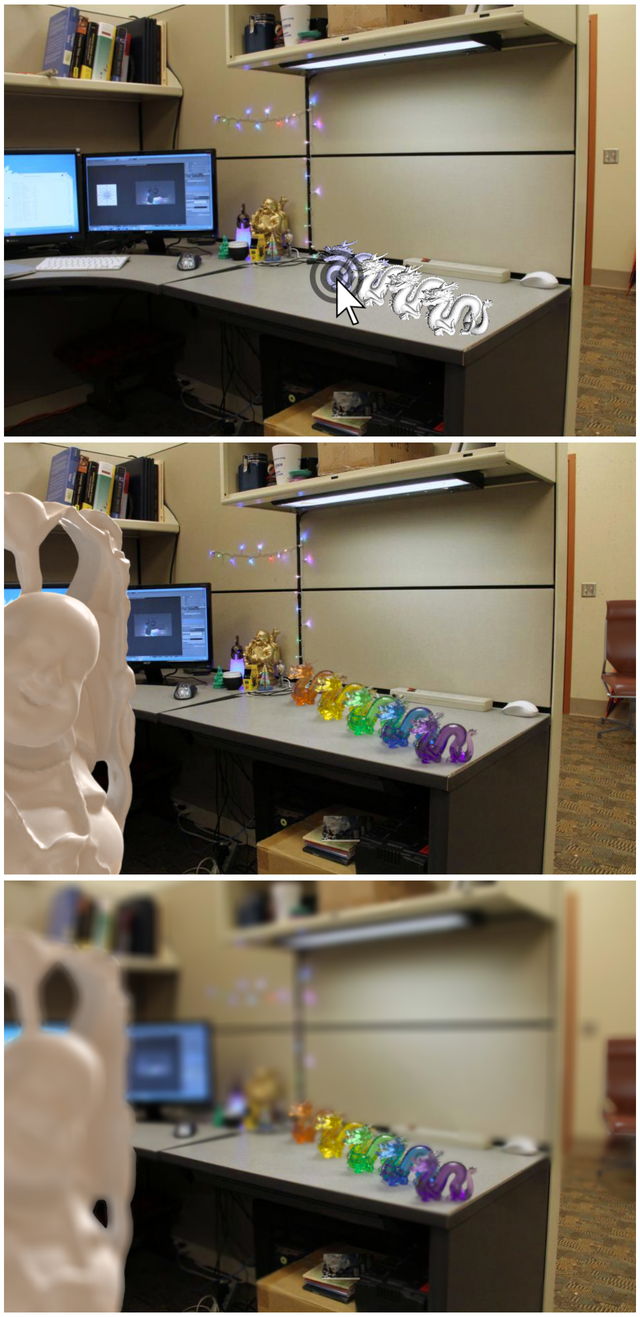

Fig. 1. From a single LDR photograph, our system automatically estimates a 3D scene model without any user interaction or additional information. These scene models facilitate photorealistic, physically grounded image editing operations, which we demonstrate with an intuitive interface. With our system, a user can simply drag-and-drop 3D models into a picture (top), render objects seamlessly into photographs with a single click (middle), adjust the illumination, and refocus the image in real time (bottom). Best viewed in color at high-resolution.

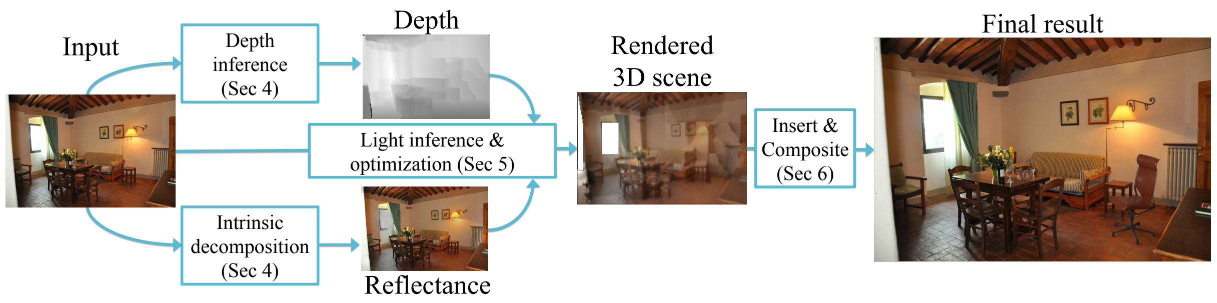
Input
Depth inference (Sec 4)
Depth
Light inference & optimization (Sec 5)
Intrinsic decomposition (Sec 4)
Reflectance
Rendered 3D scene
Insert & Composite (Sec 6)
Final result

Fig. 2. Our system allows for physically grounded image editing (e.g., the inserted dragon and chair on the right), facilitated by our automatic scene estimation procedure. To compute a scene from a single image, we automatically estimate dense depth and diffuse reflectance (the geometry and materials of our scene). Sources of illumination are then inferred without any user input to form a complete 3D scene, conditioned on the estimated scene geometry. Using a simple, drag-and-drop interface, objects are quickly inserted and composited into the input image with realistic lighting, shadowing, and perspective. *Photo credits: ©Salvadonica Borgo.*

graph automatically, allowing an artist (or, in fact, anyone) to make physically grounded edits with only a few mouse clicks.

Our system works by inferring the physical scene (geometry, illumination, etc.) that corresponds to a single LDR photograph. This process is fully automatic, requires no special hardware, and works for legacy images. We show that our inferred scene models can be used to facilitate a variety of physically-based image editing operations. For example, objects can be seamlessly inserted into the photograph, light source intensity can be modified, and the picture can be refocused on the fly. Achieving these edits with existing software is a painstaking process that takes a great deal of artistry and expertise.

In order to facilitate realistic object insertion and rendering we need to hypothesize camera parameters, scene geometry, surface materials, and sources of illumination. To address this, we develop a new method for both single-image depth and illumination inference. We are able to build a full 3D scene model without any user interaction, including camera parameters and reflectance estimates.

Contributions. Our primary contribution is a completely automatic algorithm for estimating a full 3D scene model from a single LDR photograph. Our system contains two technical contributions: illumination inference and depth estimation. We have developed a novel, data-driven illumination estimation procedure that automatically estimates a physical lighting model for the entire scene (including out-of-view light sources). This estimation is aided by our single-image light classifier to detect emitting pixels, which we believe is the first of its kind. We also demonstrate state-of-the-art depth estimates by combining data-driven depth inference with geometric reasoning.

We have created an intuitive interface for inserting 3D models seamlessly into photographs, using our scene approximation method to relight the object and facilitate drag-and-drop insertion. Our interface also supports other physically grounded image editing operations, such as post-process depth-of-field and lighting changes. In a user study, we show that our system is capable of making photorealistic edits: in side-by-side comparisons of ground truth photos with photos edited by our software, subjects had a difficult time choosing the ground truth.

Limitations. Our method works best when scene lighting is diffuse, and therefore generally works better indoors than out (see our user studies and results in Sec 7). Our scene models are clearly not canonical representations of the imaged scene and often differ significantly from the true scene components. These coarse scene reconstructions suffice in many cases to produce realistically edited images. However, in some case, errors in either geometry, illumination, or materials may be stark enough to manifest themselves in unappealing ways while editing. For example, inaccurate geometry could cause odd looking shadows for inserted objects, and inserting light sources can exacerbate geometric errors. Also, our editing software does not handle object insertion *behind* existing scene elements automatically, and cannot be used to deblur an image taken with wide aperture. A Manhattan World is assumed in our camera pose and depth estimation stages, but our method is still applicable in scenes where this assumption does not hold (see Fig 10).

2. RELATED WORK

In order to build a physically based image editor (one that supports operations such as lighting-consistent object insertion, relighting, and new view synthesis), it is requisite to model the three major factors in image formation: geometry, illumination, and surface reflectance. Existing approaches are prohibitive to most users as they require either manually recreating or measuring an imaged scene with hardware aids [Debevec 1998; Yu et al. 1999; Boivin and Gagalowicz 2001; Debevec 2005]. In contrast, Lalonde et al. [2007] and Karsch et al. [2011] have shown that even coarse estimates of scene geometry, reflectance properties, illumination, and camera parameters are sufficient for many image editing tasks. Their techniques require a user to model the scene geometry and illumination – a task that requires time and an understanding of 3D authoring tools. While our work is similar in spirit to theirs, our technique is fully automatic and is still able to produce results with the same perceptual quality.

Similar to our method, Barron and Malik [2013] recover shape, surface albedo and illumination for entire scenes, but their method requires a coarse input depth map (e.g. from a Kinect) and is not directly suitable for object insertion as illumination is only estimated near surfaces (rather than the entire volume).

Geometry. User-guided approaches to single image modeling [Horry et al. 1997; Criminisi et al. 2000; Oh et al. 2001] have been successfully used to create 3D reconstructions that allow for viewpoint variation. Single image depth estimation techniques have used learned relationships between image features and geometry to estimate depth [Hoiem et al. 2005b; Saxena et al. 2009; Liu et al. 2010; Karsch et al. 2012]. Our depth estimation technique improves upon these methods by incorporating geometric constraints, using intuition from past approaches which estimate depth by assuming a Manhattan World ([Delage et al. 2005] for single images, and [Furukawa et al. 2009; Gallup et al. 2010] for multiple images).

Other approaches to image modeling have explicitly parametrized indoor scenes as 3D boxes (or as collections of orthogonal planes) [Lee et al. 2009; Hedau et al. 2009; Karsch et al. 2011; Schwing and Urtasun 2012]. In our work, we use appearance features to infer image depth, but augment this inference with priors based on geometric reasoning about the scene.

The contemporaneous works of Satkin et al. [Satkin et al. 2012] and Del Pero et al. [Pero et al. 2013] predict 3D scene reconstructions for the rooms and their furniture. Their predicted models can be very good semantically, but are not suited well for our editing applications as the models are typically not well-aligned with the edges and boundaries in the image.

Illumination. Lighting estimation algorithms vary by the representation they use for illumination. Point light sources in the scene can be detected by analyzing silhouettes and shading along object contours [Johnson and Farid 2005; Lopez-Moreno et al. 2010]. Lalonde et al. [2009] use a physically-based model for sky illumination and a data-driven sunlight model to recover an environment map from time-lapse sequences. In subsequent work, they use the appearance of the sky in conjunction with cues such as shadows and shading to recover an environment map from a single image [Lalonde et al. 2009]. Nishino et al. [2004] recreate environment maps of the scene from reflections in eyes. Johnson and Farid [2007] estimate lower-dimensional spherical harmonics-based lighting models from images. Panagopoulos et al. [2011] show that automatically detected shadows can be used to recover an illumination environment from a single image, but require coarse geometry as input.

While all these techniques estimate physically-based lighting from the scene, Khan et al. [2006] show that wrapping an image to create the environment map can suffice for certain applications.

Our illumination estimation technique attempts to predict illumination both within and outside the photograph's frustum with a data-driven matching approach; such approaches have seen previous success in recognizing scene viewpoint [Xiao et al. 2012] and view extrapolation [Zhang et al. 2013].

We also attempt to predict a one-parameter camera response function jointly during our inverse rendering optimization. Other processes exist for recovering camera response, but require multiple of images [Diaz and Sturm 2013].

Materials. In order to infer illumination, we separate the input image into diffuse albedo and shading using the Color Retinex algorithm [Grosse et al. 2009]. We assume that the scene is Lambertian, and find that this suffices for our applications. Other researchers have looked at the problem of recovering the Bi-directional Reflectance Density Function (BRDF) from a single image, but as far as we know, there are no such methods that work automatically and at the scene (as opposed to object) level. These techniques typically make the problem tractable by using low-dimensional representations for the BRDF such as spherical harmonics [Ramamoorthi and Hanrahan 2004], bi-variate models for isotropic reflectances [Romeiro et al. 2008; Romeiro and Zickler 2010], and data-driven statistical models [Lombardi and Nishino 2012a; 2012b]. All these techniques require the shape to be known and in addition, either require the illumination to be given, or use priors on lighting to constrain the problem space.

Perception. Even though our estimates of scene geometry, materials, and illumination are coarse, they enable us to create realistic composites. This is possible because even large changes in lighting are often not perceivable to the human visual system. This has been shown to be true for both point light sources [Lopez-Moreno et al. 2010] and complex illumination [Ramanarayanan et al. 2007].

3. METHOD OVERVIEW

Our method consists of three primary steps, outlined in Fig 2. First, we estimate the physical space of the scene (camera parameters and geometry), as well as the per-pixel diffuse reflectance (Sec 4). Next, we estimate scene illumination (Sec 5) which is guided by our previous estimates (camera, geometry, reflectance). Finally, our interface is used to composite objects, improve illumination estimates, or change the depth-of-field (Sec 6). We have evaluated our method with a large-scale user study (Sec 7), and additional details and results can be found in the corresponding supplemental document. Figure 2 illustrates the pipeline of our system.

Scene parameterization. Our geometry is in the form of a depth map, which is triangulated to form a polygonal mesh (depth is unprojected according to our estimated pinhole camera). Our illumination model contains polygonal area sources, as well as one or more spherical image-based lights.

While unconventional, our models are suitable for most off-the-shelf rendering software, and we have found our models to produce better looking estimates than simpler models (e.g. planar geometry with infinitely distant lighting).

Automatic indoor/outdoor scene classification. As a pre-processing step, we automatically detect whether the input image is indoors or outdoors. We use a simple method: k-nearest-neighbor matching of GIST features [Oliva and Torralba 2001] between the input image and all images from the indoor NYUv2 dataset and the outdoor Make3D Dataset. We choose $k = 7$, and decide use majority-voting to determine if the image is indoors or outdoors (e.g. if 4 of the nearest neighbors are from the Make3D dataset, we consider it to be outdoors). More sophisticated methods could also work.

Our method uses different training images and classifiers depending on whether the input image is classified as an indoor or outdoor scene.

4. SINGLE IMAGE RECONSTRUCTION

The first step in our algorithm is to estimate the physical space of the scene, which we encode with a depth map, camera parameters, and spatially-varying diffuse materials. Here, we describe how to estimate these components, including a new technique for estimating depth from a single image that adheres to geometric intuition about indoor scenes.

Single image depth estimation. Karsch et al. [2012] describe a non-parametric, "depth transfer" approach for estimating dense, per-pixel depth from a single image. While shown to be state-of-the-art, this method is purely data-driven, and incorporates no explicit geometric information present in many photographs. It requires a database of RGBD (RGB+depth) images, and transfers depth from the dataset to a novel input image in a non-parametric fashion using correspondences in appearance. This method has been shown to work well both indoors and outdoors; however, only appearance cues are used (multi-scale SIFT features), and we have good reason to believe that adding geometric information will aid in this task.

A continuous optimization problem is solved to find the most likely estimate of depth given an input image. In summary, images in the RGBD database are matched to the input and warped so that SIFT features are aligned, and an objective function is minimized to arrive at a solution. We denote $\mathbf{D}$ as the depth map we wish to infer, and following the notation of Karsch et al., we write the full

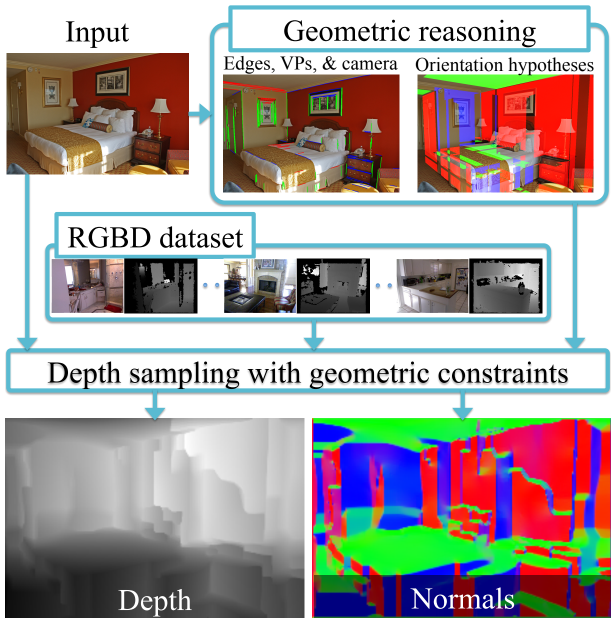

Fig. 3. Automatic depth estimation algorithm. Using the geometric reasoning method of Lee et al. [2009], we estimate focal length and a sparse surface orientation map. Facilitated by a dataset of RGBD images, we then apply a non-parametric depth sampling approach to compute the per-pixel depth of the scene. The geometric cues are used during inference to enforce orientation constraints, piecewise-planarity, and surface smoothness. The result is a dense reconstruction of the scene that is suitable for realistic, physically grounded editing. *Photo credits: Flickr user © "Mr.TinDC".*

objective here for completeness:

$$\operatorname*{argmin}_{\mathbf{D}} E(\mathbf{D}) = \sum_{i \in \text{pixels}} E_t(\mathbf{D}_i) + \alpha E_s(\mathbf{D}_i) + \beta E_p(\mathbf{D}_i), \quad (1)$$

where E_t is the data term (depth transfer), E_s enforces smoothness, and E_p is a prior encouraging depth to look like the average depth in the dataset. We refer the reader to [Karsch et al. 2012] for details.

Our idea is to reformulate the depth transfer objective function and infuse it with geometric information extracted using the geometric reasoning algorithm of Lee et al. [2009]. Lee et al. detect vanishing points and lines from a single image, and use these to hypothesize a set of sparse surface orientations for the image. The predicted surface orientations are aligned with one of the three dominant directions in the scene (assuming a Manhattan World).

We remove the image-based smoothness (E_s) and prior terms (E_p), and replace them with geometric-based priors. We add terms to enforce a Manhattan World (E_m), constrain the orientation of planar surfaces (E_o), and impose 3D smoothness (E_{3s}, spatial smoothness in 3D rather than 2D):

$$\operatorname*{argmin}_{\mathbf{D}} E_{geom}(\mathbf{D}) = \sum_{i \in \text{pixels}} E_t(\mathbf{D}_i) + \lambda_m E_m(N(\mathbf{D})) + \quad (2)$$
$$\lambda_o E_o(N(\mathbf{D})) + \lambda_{3s} E_{3s}(N(\mathbf{D})),$$

where the weights are trained using a coarse-to-fine grid search on held-out ground truth data (indoors: $\lambda_m = 100$, $\lambda_o = 0.5$, $\lambda_{3s} = 0.1$, outdoors: $\lambda_m = 200$, $\lambda_o = 10$, $\lambda_{3s} = 1$); these weights dictate the amount of influence each corresponding prior has during optimization. Descriptions of these priors and implementation details can be found in the supplemental file.

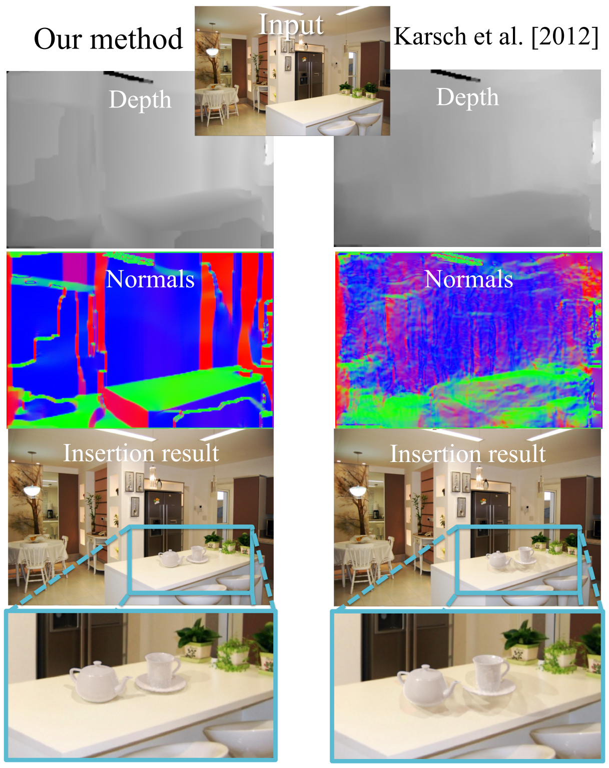

Fig. 4. Comparison of different depth estimation techniques. Although the depth maps of our technique and of Karsch et al. [2012] appear roughly similar, the surface orientations provide a sense of how distinct the methods are. Our depth estimation procedure (aided by geometric reasoning) is crucial in achieving realistic insertion results, as the noisy surface orientations from Karsch et al. can cause implausible cast shadows and lighting effects.

Figure 3 shows the pipeline of our depth estimation algorithm, and Fig 4 illustrates the differences between our method and the depth transfer approach; in particular, noisy surface orientations from the depth of Karsch et al. [2012] lead to unrealistic insertion and relighting relights.

In supplemental material, we show additional results, including state-of-the-art results on two benchmark datasets using our new depth estimator.

Camera parameters. It is well known how to compute a simple pinhole camera (focal length, f and camera center, (c_0^x, c_0^y)) and extrinsic parameters from three orthogonal vanishing points [Hartley and Zisserman 2003] (computed during depth estimation), and we use this camera model at render-time.

Surface materials. We use Color Retinex (as described in [Grosse et al. 2009]), to estimate a spatially-varying diffuse material albedo for each pixel in the visible scene.

5. ESTIMATING ILLUMINATION

We categorize luminaires into *visible* sources (sources visible in the photograph), and *out-of-view* sources (all other luminaires). Visible sources are detected in the image using a trained "light classifier" (Sec 5.1). Out-of-view sources are estimated through a data-driven procedure (Sec 5.2) using a large dataset of annotated spherical panoramas (SUN360 [Xiao et al. 2012]). The resulting lighting

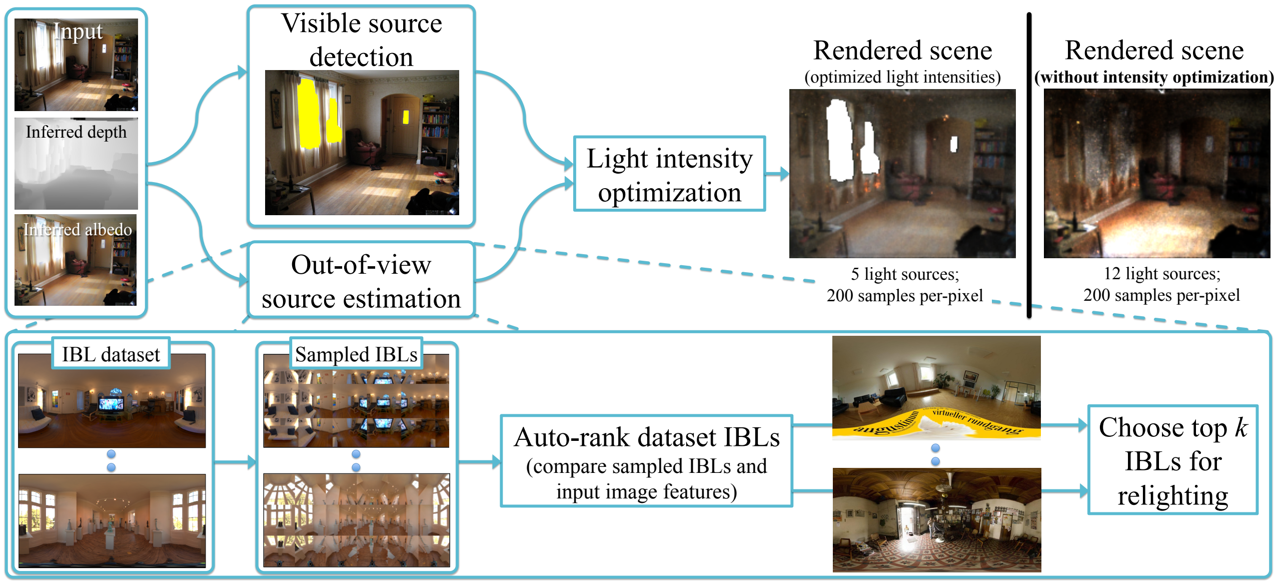
Input
Inferred depth
Inferred albedo
Visible source detection
Out-of-view source estimation
Light intensity optimization
Rendered scene
(optimized light intensities)
5 light sources; 200 samples per-pixel
Rendered scene
(without intensity optimization)
12 light sources; 200 samples per-pixel
IBL dataset
Sampled IBLs
Auto-rank dataset IBLs
(compare sampled IBLs and input image features)
Choose top k IBLs for relighting

Fig. 5. Overview of our lighting estimation procedure. Light sources are first detected in the input image using our light classifier. To estimate light outside of the view frustum, we use a data-driven approach (utilizing the SUN360 panorama dataset). We train a ranking function to rank IBLs according to how well they "match" the input image's illumination (see text for details), and use the top k IBLs for relighting. Finally, source intensities are optimized to produce a rendering of the scene that closely matches the input image. Our optimization not only encourages more plausible lighting conditions, but also improves rendering speed by pruning inefficient light sources.

model is a hybrid of area sources and spherical emitting sources. Finally, light source intensities are estimated using an optimization procedure which adjusts light intensities so that the rendered scene appears similar to the input image (Sec 5.3). Figure 5 illustrates this procedure.

Dataset. We have annotated light sources in 100 indoor and 100 outdoor scenes from the SUN360 dataset. The annotations also include a discrete estimate of distance from the camera ("close": 1-5m, "medium": 5-50m, "far": $>$50m, "infinite": reserved for sun). Since SUN360 images are LDR and tonemapped, we make no attempt to annotate absolute intensity, only the position/direction of sources. Furthermore, the goal of our classifier is only to predict location (not intensity). This data is used in both our in-view and out-of-view techniques below.

5.1 Illumination visible in the view frustum

To detect sources of light in the image, we have developed a new light classifier. For a given image, we segment the image into superpixels using SLIC [Achanta et al. 2012], and compute features for each superpixel. We use the following features: the height of the superpixel in the image (obtained by averaging the 2D location of all pixels in the superpixel), as well as the features used by Make3D[1] [Saxena et al. 2009].

Using our annotated training data, we train a binary classifier to predict whether or not a superpixel is emitting/reflecting a significant amount of light using these features. For this task, we do not use the discrete distance annotations (these are however used in Sec 5.2). A classification result is shown in Figure 5. In supplemental material, we show many more qualitative and quantitative results of our classifier, and demonstrate that our classifier significantly outperforms baseline detectors (namely thresholding).

For each detected source superpixel (and corresponding pixels), we find their 3D position using the pixel's estimated depth and the projection operator K. Writing D as the estimated depth map and (x, y) as a pixel's position in the image plane, the 3D position of the pixel is given by:

$$\mathbf{X} = D(x, y)K^{-1}[x, y, 1]^T, \tag{3}$$

where K is the intrinsic camera parameters (projection matrix) obtained in Section 4. We obtain a polygonal representation of each light source by fitting a 3D quadrilateral to each cluster (oriented in the direction of least variance). Notice that this only provides the position of the visible light sources; we describe how we estimate intensity in Section 5.3. Figure 5 (top) shows a result of our light detector.

One might wonder why we train using equirectangular images and test using rectilinear images. Dror et al. [2004] showed that many image statistics computed on equirectangular images follow the same distributions as those computed on rectilinear images; thus features computed in either domain should be roughly the same. We have also tested our method on both kinds of images (more results in the supplemental file), and see no noticeable differences.

5.2 Illumination outside of the view frustum

Estimating lighting from behind the camera is arguably the most difficult task in single-image illumination estimation. We use a data-driven approach, utilizing the extensive SUN360 panorama

[1] 17 edge/smoothing filter responses in YCbCr space are averaged over the superpixel. Both the energy and kurtosis of the filter responses are computed (second and fourth powers) for a total of 34 features, and then concatenated with four neighboring (top,left,bottom,right) superpixel features. This is done at two scales (50% and 100%), resulting in $340 (= 34 \times 5 \times 2)$ features per superpixel.

dataset of Xiao et al. [2012]. However, since this dataset is not available in HDR, we have annotated the dataset to include light source positions and distances.

Our primary assumption is that if two photographs have similar appearance, then the illumination environment beyond the photographed region will be similar as well. There is some empirical evidence that this can be true (e.g. recent image-extrapolation methods [Zhang et al. 2013]), and studies suggest that people hallucinate out-of-frame image data by combining photographic evidence with recent memories [Intraub and Richardson 1989].

Using this intuition, we develop a novel procedure for matching images to luminaire-annotated panoramas in order to predict out-of-view illumination. We sample each IBL into N rectilinear projections (2D) at different points on the sphere and at varying fields-of-view, and match these projections to the input image using a variety of features described below (in our work, $N = 10$ stratified random samples on the sphere with azimuth $\in [0, 2\pi)$, elevation $\in [-\frac{\pi}{6}, \frac{\pi}{6}]$, FOV $\in [\frac{\pi}{3}, \frac{\pi}{2}]$). See the bottom of Fig 5 for an illustration. These projections represent the images a camera with that certain orientation and field of view would have captured for that particular scene. By matching the input image to these projections, we can effectively "extrapolate" the scene outside the field of view and estimate the out-of-view illumination.

Given an input image, our goal is to find IBLs in our dataset that emulate the input's illumination. Our rectilinear sampled IBLs provide us with ground truth data for training: for each sampled image, we know the corresponding illumination. Based on the past success of rank prediction for data-driven geometry estimation [Satkin et al. 2012], we use this data and train an IBL rank predictor (for an input image, rank the dataset IBLs from best to worst).

Features. After sampling the panoramas into rectilinear images, we compute seven features for each image: geometric context [Hoiem et al. 2005a], orientation maps [Lee et al. 2009], spatial pyramids [Lazebnik et al. 2006], HSV histograms (three features total), and the output of our light classifier (Sec 5.1).

We are interested in ranking pairs of images, so our final feature describes how well two particular images match in feature space. The result is a 7-dimensional vector where each component describes the similarity of a particular feature (normalized to [0,1], where higher values indicate higher similarity). Similarity is measured using the histogram intersection score for histogram features (spatial pyramid – sp – and HSV – h,s,v – histograms), and following Satkin et al. [2012], a normalized, per-pixel dot product (averaged over the image) is used for for other features (geometric contact – gc, orientation maps – om, light classifier – lc).

More formally, let F_i, F_j be the features computed on images i and j, and x_j^i be the vector that measures similarity between the features/images:

$$\begin{aligned} x_j^i = [&nd(F_i^{\text{gc}}, F_j^{\text{gc}}), nd(F_i^{\text{om}}, F_j^{\text{om}}), nd(F_i^{\text{lc}}, F_j^{\text{lc}}), \\ &hi(F_i^{\text{sp}}, F_j^{\text{sp}}), hi(F_i^{\text{h}}, F_j^{\text{h}}), hi(F_i^{\text{s}}, F_j^{\text{s}}), hi(F_i^{\text{v}}, F_j^{\text{v}})]^T, \end{aligned} \quad (4)$$

where $nd(\cdot), hi(\cdot)$ are normalized dot product and histogram intersection operators respectively. In order to compute per-pixel dot products, images must be the same size. To compute features on two images with unequal dimension, we downsample the larger image to have the same dimension as the smaller image.

Training loss metric. In order to discriminate between different IBLs, we need to define a distance metric that measure how similar one IBL is to another. We use this distance metric as the loss for our rank training optimization where it encodes the additional margin when learning the ranking function (Eq 6).

A naïve way to measure the similarity between two IBLs is to use pixel-wise or template matching [Xiao et al. 2012]. However, this is not ideal for our purposes since it requires accurate correspondences between elements of the scene that may not be important from a relighting aspect. Since our primary goal is re-rendering, we define our metric on that basis of how different a set of canonical objects appear when they are illuminated by these IBLs. In particular, we render nine objects with varying materials (diffuse, glossy, metal, etc.) into each IBL environment (see supplemental material). Then, we define the distance as the mean L2 error between the renderings.

One caveat is that our IBL dataset isn't HDR, and we don't know the intensities of the annotated sources. So, we compute error as the minimum over all possible light intensities. Define $\mathbf{I}_i$ and $\mathbf{I}_j$ as two IBLs in our dataset, and $I_i = [I_i^{(1)}, \ldots, I_i^{(n)}], I_j = [I_j^{(1)}, \ldots, I_j^{(m)}]$ as column-vectorized images rendered by the IBLs for each of the IBLs' sources (here $\mathbf{I}_i$ has n sources, and $\mathbf{I}_j$ has m). Since a change in the intensity of the IBL corresponds to a change of a scale factor for the rendered image, we define the distance as the minimum rendered error over all possible intensities (y_i and y_j):

$$d(\mathbf{I}_i, \mathbf{I}_j) = \min_{y_i, y_j} ||I_i y_i - I_j y_j||, \text{ s.t. } ||[y_i^T, y_j^T]|| = 1. \quad (5)$$

The constraint is employed to avoid the trivial solution, and we solve this using SVD.

Training the ranking function. Our goal is to train a ranking function that can properly rank images (and their corresponding IBLs) by how well their features match the input's. Let w be a linear ranking function, and x_j^i be features computed between images i and j. Given an input image i and any two images from our dataset (sampled from the panoramas) j and k, we wish to find w such that $w^T x_j^i > w^T x_k^i$ when the illumination of j matches the illumination of i better than the illumination of k.

To solve this problem, we perform a standard 1-slack, linear SVM-ranking optimization [Joachims 2006]:

$$\operatorname*{argmin}_{w,\xi} ||w||^2 + C\xi, \text{ s.t. } w^T x_j^i \geq w^T x_k^i + \delta_{j,k}^i - \xi, \; \xi \geq 0, \quad (6)$$

where x are pairwise image similarity features (Eq 4), and $\delta_{j,k}^i = \max(d(\mathbf{I}_i, \mathbf{I}_k) - d(\mathbf{I}_i, \mathbf{I}_j), 0)$ is a hinge loss to encourage additional margin for examples with unequal distances (according to Eq 5, where $\mathbf{I}_i$ is the IBL corresponding to image i).

Inference. To predict the illumination of a novel input image (i), we compute the similarity feature vector (Eq 4) for all input-training image pairs (x_j^i, $\forall j$), and sort the prediction function responses ($w^T x_j^i$) in decreasing order. Then, we use choose the top k IBLs (in our work, we use $k = 1$ for indoor images, and $k = 3$ outdoors to improve the odds of predicting the correct sun location). Figure 5 (bottom) shows one indoor result using our method (where $k = 2$ for demonstration).

5.3 Intensity estimation through rendering

Having estimated the location of light sources within and outside the image, we must now recover the relative intensities of the sources. Given the exact geometry and material of the scene (including light source positions), we can estimate the intensities of the sources by adjusting them them until a rendered version of the scene matches the original image. While we do not know exact geometry/materials, we assume that our automatic estimates are good enough, and could apply the above rendering-based optimization to recover the light source intensities; Fig 5 shows this process. Our optimization has two goals: match the rendered image to the input, and differing from past techniques [Boivin and Gagalowicz 2001; Karsch et al. 2011], ensure the scene renders efficiently.

Define L_i as the intensity of the i^{th} light source, I as the input image, and $R(\cdot)$ as a scene "rendering" function that takes a set of light intensities and produces an image of the scene illuminated by those lights (i.e., $R(L)$). We want to find the intensity of each light source by matching the input and rendered images, so we could solve $\operatorname{argmin}_L \sum_{i \in \text{pixels}} ||I_i - R_i(L)||$. However, this optimization can be grossly inefficient and unstable, as it requires a new image to be rendered for each function evaluation, and rendering in general is non-differentiable. However, we can use the fact that light is additive, and write $R(\cdot)$ as a linear combination of "basis" renders [Schoeneman et al. 1993; Nimeroff et al. 1994]. We render the scene (using our estimated geometry and diffuse materials) using only one light source at a time (i.e., $L_k = 1, L_j = 0 \; \forall j \neq k$, implying $L = e_k$). This results in one rendered image per light source, and we can write a new render function as $R'(w) = C\left(\sum_k w_k R(e_k)\right)$, where C is the camera response function, and $R(e_i)$ is the scene rendered with only the i^{th} source. In our work, we assume the camera response can be modeled as an exponent, i.e., $C(x) = x^\gamma$. This allows us to rewrite the matching term above as

$$Q(w,\gamma) = \sum_{i \in \text{pixels}} \left|\left| I_i - \left[\sum_{k \in \text{sources}} w_k R_i(e_k) \right]^\gamma \right|\right|. \quad (7)$$

Since each basis render, $R(e_k)$, can be precomputed prior to the optimization, Q can be minimized more efficiently than the originally described optimization.

We have hypothesized a number of light source locations in Secs 5.1 and 5.2, and because our scene materials are purely diffuse, and our geometry consists of a small set of surface normals, there may exist an infinite number of lighting configurations that produce the same rendered image. Interestingly, user studies have shown that humans cannot distinguish between a range of illumination configurations [Ramanarayanan et al. 2007], suggesting that there is a family of lighting conditions that produce the same perceptual response. This is actually advantageous, because it allows our optimization to choose only a small number of "good" light sources and prune the rest. In particular, since our final goal is to relight the scene with the estimated illumination, we are interested in lighting configurations that can be rendered faster. We can easily detect if a particular light source will cause long render times by rendering the image with the given source for a fixed amount of time, and checking the variance in the rendered image (i.e., noise); we incorporate this into our optimization. By rendering with fewer sources and sources that contribute less variance, the scenes produced by our method render significantly faster than without this optimization (see Fig 5, right).

Specifically, we ensure that only a small number of sources are used in the final lighting solution, and also prune problematic light sources that may cause inefficient rendering. We encode this with a sparsity prior on the source intensities and a smoothness prior on the rendered images:

$$P(w) = \sum_{k \in \text{sources}} \left[||w_k||_1 + w_k \sum_{i \in \text{pixels}} ||\nabla R_i(e_k)|| \right]. \quad (8)$$

Intuitively, the first term coerces only a small number of nonzero elements in w, and the second term discourages noisy basis renders from having high weights (noise in a basis render typically indicates an implausible lighting configuration, making the given image render much more slowly).

Combining the previous equations, we develop the following optimization problem:

$$\underset{w,\gamma}{\operatorname{argmin}} \;\; Q(w,\gamma) + \lambda_P P(w) + \lambda_\gamma ||\gamma - \gamma_0||, \qquad \text{s.t. } w_k \geq 0 \; \forall k, \; \gamma > 0, \quad (9)$$

where $\lambda_P = \lambda_\gamma = 0.1$ are weights, and $\gamma_0 = \frac{1}{2.2}$. We use a continuous approximation to the absolute value ($|x| \approx \sqrt{x^2 + \epsilon}$), and solve using the active set algorithm [Nocedal and Wright 2006]. The computed weights (w) can be directly translated into light intensities (L), and we now have an entire model of the scene (geometry, camera, and materials from Sec 4, and light source positions/intensities as described above).

Our method has several advantages to past "optimization-through-rendering" techniques [Karsch et al. 2011]. First, our technique has the ability to discard unnecessary and inefficient light sources by adding illumination priors (Eq 8). Second, we estimate the camera response function jointly during our optimization, which we do not believe has been done previously. Finally, by solving for a simple linear combination of pre-rendered images, our optimization procedure is much faster than previous methods that render the image for each function evaluation (e.g., as in [Karsch et al. 2011]). Furthermore, our basis lights could be refined or optimized with user-driven techniques guided by aesthetic principles (e.g. [Boyadzhiev et al. 2013]).

6. PHYSICALLY GROUNDED IMAGE EDITING

One of the potential applications of our automatically generated 3D scene models is physically grounded photo editing. In other words, we can use the approximate scene models to facilitate physically-based image edits. For example, one can use the 3D scene to insert and composite objects with realistic lighting into a photograph, or even adjust the depth-of-field and aperture as a post-process.

There are many possible interactions that become available with our scene model, and we have developed a prototype interface to facilitate a few of these. Realistically inserting synthetic objects into legacy photographs is our primary focus, but our application also allows for post-process lighting and depth-of-field changes. To fully leverage our scene models, we require physically based rendering software. We use LuxRender[2], and have built our application on top of LuxRender's existing interface. Figure 6 illustrates the possible uses of the interface, and we refer the reader to the accompanying video for a full demonstration.

Drag-and-drop insertion. Once a user specifies an input image for editing, our application automatically computes the 3D scene as described in sections 4 and 5. Based on the inserted location, we also add additional geometric constraints so that the depth is flat in a small region region around the base of the inserted object. For a given image of size 1024×768, our method takes approximately five minutes on a 2.8Ghz dual core laptop to estimate the 3D scene model, including depth and illumination (this can also be precomputed or computed remotely for efficiency, and only occurs once per image). Next, a user specifies a 3D model and places it in the scene by simply clicking and dragging in the picture (as in Fig 1). Our scene reconstruction facilitates this insertion: our estimated perspective camera ensures that the object is scaled properly

[2]`http://www.luxrender.net`

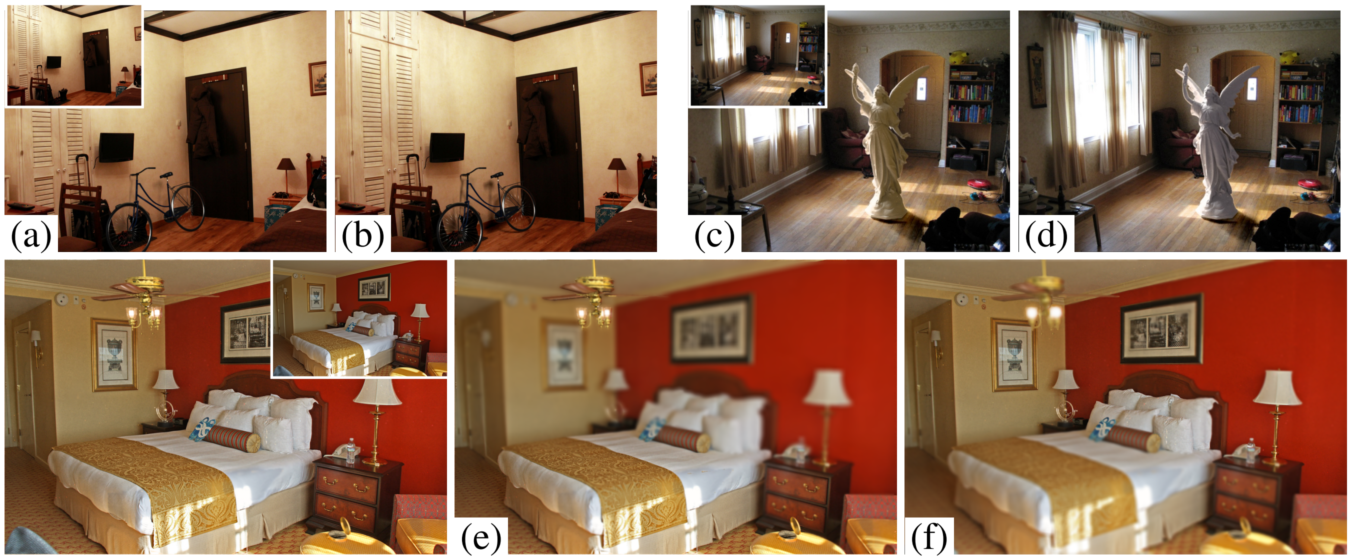

Fig. 6. Our user interface provides a drag-and-drop mechanism for inserting 3D models into an image (input image overlaid on our initial result). Our system also allows for real-time illumination changes without re-rendering. Using simple slider controls, shadows and interreflections can be softened or hardened (a,b), light intensity can be adjusted to achieve a different mood (c,d), and synthetic depth-of-field can be applied to re-focus the image (e,f). See the accompanying video for a full demonstration. Best viewed in color at high-resolution. *Photo credits: ©Rachel Titiriga (top) and Flickr user ©"Mr.TinDC" (bottom).*

as it moves closer/farther from the camera, and based on the surface orientation of the clicked location, the application automatically re-orients the inserted model so that its up vector is aligned with the surface normal. Rigid transformations are also possible through mouse and keyboard input (scroll to scale, right-click to rotate, etc).

Once the user is satisfied with the object placement, the object is rendered into the image[3]. We use the additive differential rendering method [Debevec 1998] to composite the rendered object into the photograph, but other methods for one-shot rendering could be used (e.g. Zang et al. [2012]). This method renders two images: one containing synthetic objects I_{obj}, and one without synthetic objects I_{noobj}, as well as an object mask M (scalar image that is 0 everywhere where no object is present, and (0, 1] otherwise). The final composite image I_{final} is obtained by

$$I_{final} = M \odot I_{obj} + (1 - M) \odot (I_b + I_{obj} - I_{noobj}), \tag{10}$$

where I_b is the input image, and $\odot$ is the entry-wise product. For efficiency (less variance and overhead), we have implemented this equation as a surface integrator plugin for LuxRender. Specifically, we modify LuxRender's bidirectional path tracing implementation [Pharr and Humphreys 2010] so that pixels in I_{final} are intelligently sampled from either the input image or the rendered scene such that inserted objects and their lighting contributions are rendered seamlessly into the photograph. We set the maximum number of eye and light bounces to 16, and use LuxRender's default Russian Roulette strategy (dynamic thresholds based on past samples).

[3]Rendering time is clearly dependent on a number of factors (image size, spatial hierarchy, inserted materials, etc), and is slowed by the fact that we use an unbiased ray-tracer. The results in this paper took between 5 minutes and several hours to render, but this could be sped up depending on the application and resources available.

The user is completely abstracted from the compositing process, and only sees I_{final} as the object is being rendered. The above insertion method also works for adding light sources (for example, inserting and emitting object). Figure 1 demonstrates a drag-and-drop result.

Lighting adjustments. Because we have estimated sources for the scene, we can modify the intensity of these sources to either add or subtract light from the image. Consider the compositing process (Eq 10 with no inserted objects; that is, $I_{obj} = I_{noobj}$, implying $I_{final} = I_b$. Now, imagine that the intensity of a light source in I_{obj} is increased (but the corresponding light source in I_{noobj} remains the same). I_{obj} will then be brighter than I_{noobj}, and the compositing equation will reflect a physical change in brightness (effectively rendering a more intense light into the scene). Similarly, decreasing the intensity of source(s) in I_{obj} will remove light from the image. Notice that this is a physical change to the lighting in the picture rather than tonal adjustments (e.g., applying a function to an entire image).

This technique works similarly when there are inserted objects present in the scene, and a user can also manually adjust the light intensities in a scene to achieve a desired effect (Fig 6, top). Rather than just adjusting source intensities in I_{obj}, if sources in both I_{obj} and I_{noobj} are modified equally, then only the intensity of the inserted object and its interreflections (shadows, caustics, etc) will be changed (i.e. without adding or removing light from the rest of the scene).

By keeping track of the contribution of each light source to the scene, we only need to render the scene once, and lighting adjustments can be made in real time without re-rendering (similar to [Loscos et al. 1999; Gibson and Murta 2000]). Figure 6 shows post-process lighting changes; more can be found in supplemental material.

Synthetic depth-of-field. Our prototype also supports post-focusing the input image: the user specifies a depth-of-field and an aperture size, and the image is adaptively blurred using our predicted depth map ($\mathbf{D}$). Write I_{final} as the composite image with inserted objects (or the input image if nothing is inserted), and $G(\sigma)$ as a Gaussian kernel with standard deviation σ. We compute the blur at the i^{th} pixel as

$$I_{dof,i} = I_{final,i} \star G(a|\mathbf{D}_i - d|), \tag{11}$$

where d is the depth of field, and a corresponds to the aperture size. Figures 1 and 6 (bottom) shows a post-focus result.

7. EVALUATION

We conducted two user studies to evaluate the "realism" achieved by our insertion technique. Each user study is a series of two-alternative forced choice tests, where the subject chooses between a pair of images which he/she feels looks the most realistic.

In the first study (Sec 7.1), we use real pictures. Each pair of images contains one actual photograph with a small object placed in the scene, and the other is photo showing a similar object inserted synthetically into the scene (without the actual object present).

The second study (Sec 7.2) is very similar, but we use highly realistic, synthetic images rather than real pictures. For one image, a synthetic object is placed into the full 3D environment and rendered (using unbiased, physically-based ray tracing); in the other, the same 3D scene is rendered without the synthetic object (using the same rendering method), and then synthetically inserted *into the photograph* (rather than the 3D scene) using our method.

Finally, we visualize the quantitative accuracy achieved by our estimates in Section 7.3.

7.1 Real image user study

Experimental setup. We recruited 30 subjects for this study, and we used a total of 10 synthetic objects across three unique, real scenes. Each subject saw each inserted object only once, and viewed seven to nine side-by-side trial images; one was a real photograph containing a real object, and the other is a synthetic image produced by our method (containing a synthetic version of the real object in the real photo). Subjects were asked to choose the image they felt looked most realistic. Trials and conditions were permuted to ensure even coverage. An example trial pair is shown in Fig 7, and more can be found in supplemental material.

Conditions. We tested six binary conditions that we hypothesized may contribute to a subject's performance: **expert** subject; **realistic shape**; **complex material**; **automatic** result; **multiple objects** inserted; and whether the trial occured in the **first half** of a subject's study (to test whether there was any learning effect present). Subjects were deemed experts if they had graphics/vision experience (in the study, this meant graduate students and researchers at a company lab), and non-experts otherwise. The authors classified objects into realistic/synthetic shape and complex/simple material beforehand. We created results with our automatic drag-and-drop procedure as one condition ("automatic"), and using our interface, created an improved (as judged by the authors) version of each result by manually adjusting the light intensities ("refined").

Results. On average, the result generated by our **automatic** method was selected as the real image in 34.1% of the 232 pairs that subjects viewed. The refined condition achieved 35.8% confusion, however these distributions do not significantly differ using a two-tailed t-test. An optimal result would be 50%. For comparison, Karsch et al. [2011] evaluated their semiautomatic method with a similar user study, and achieved 34% confusion, although their study (scenes, subjects, etc) were not identical to ours.

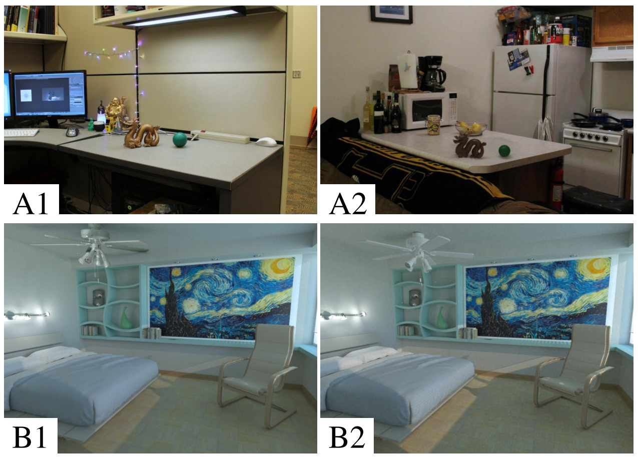

Fig. 7. Example trials from our "real image" user study (top) and our "synthetic image" user study (bottom). In the studies, users were shown two side-by-side pictures; one photograph is real (or rendered photorealistically with an exact 3D scene), and the other has synthetic objects inserted into it with our method. Users were instructed to choose the picture from the pair that looked the most realistic. For each row, which of the pair would you choose? Best viewed in color at high-resolution. *Scene modeling credits (bottom): Matthew Harwood.*

A1,B2: real/fully rendered; A2, B1: created with our method

We also tested to see if any of the other conditions (or combination of conditions) were good predictors of a person's ability to perform this task. We first used logistic regression to predict which judgements would be correct using the features, followed by a Lasso (L_1 regularized logistic regression [Tibshirani 1996], in the Matlab R2012b implementation) to identify features that predicted whether the subject judged a particular pair correctly. We evaluated deviance (mean negative log-likelihood on held out data) with 10-fold cross validation. The lowest deviance regression ignores all features; incorporating the "expert" and "realistic shape" features causes about a 1% increase in the deviance; incorporating other features causes the deviance to increase somewhat. This strongly suggests that none of the features actually affect performance.

7.2 Synthetic image user study

Our "real image" study provides encouraging initial results, but it is limited due to the nature of data collection. Collecting corresponding real and synthetic objects with high-quality geometry and reflectance properties can be extremely difficult, so we confined our real image study to small, tabletop accessories (household items and 3D-printable items).

In this follow-up study, we utilize highly realistic, synthetic 3D scenes in order to more extensively evaluate our method (objects with larger-pixel coverage and varying materials/geometry, diverse lighting conditions, indoors/outdoors).

Experimental setup. We collected four synthetic (yet highly realistic[4]) scenes – three indoor, one outdoor. For each scene, we inserted three realistic 3D models using conventional modeling soft-

[4]We pre qualified these images as highly-confusable with real pictures in a preliminary study; see supplemental material.

ware (Blender, `http://www.blender.org/`), and rendered each scene using LuxRender under three lighting conditions (varying from strongly directed to diffuse light), for a total of 36 (= 4 scenes $\times$ 3 objects $\times$ 3 lighting conditions) unique and varied scenes (viewable in supplemental material). Next, we used our method to insert the same 3D models in roughly the same location into the empty, rendered images, resulting in 36 "synthetic" insertion results corresponding to the 36 ground truth rendered images.

For the study, each participant viewed 12 pairs[5] of corresponding images, and was asked to select which image he/she felt looked the most realistic (two-alternative forced choice). For example, a subject might see two identical bedroom scenes with a ceiling fan, except in one picture, the fan had actually been inserted using our method (see Fig 7).

We polled 450 subjects using Mechanical Turk. In an attempt to avoid inattentive subjects, each study also included four "qualification" image pairs (a cartoon picture next to a real image) placed throughout the study in a stratified fashion. Subjects who incorrectly chose any of the four cartoon picture as realistic were removed from our findings (16 in total, leaving 434 studies with usable data).

At the end of the study, we showed subjects two additional image pairs: a pair containing rendered spheres (one a physically plausible, the other not), and a pair containing line drawings of a scene (one with proper vanishing point perspective, the other not). For each pair, subjects chose the image they felt looked most realistic. Then, each subject completed a brief questionnaire, listing demographics, expertise, and voluntary comments.

Methods tested. We are primarily interested in determining how well our method compares to the ground truth images, but also test other illumination estimation methods as baselines. We generate results using the method of Khan et al. [2006] (projecting the input image onto a hemisphere, duplicating it, and using this as the illumination environment), the method of Lalonde et al. [2009] (for outdoor images only), and a simplified IBL-matching technique that finds a similar IBL to the input image by template matching (similar to [Xiao et al. 2012]; we use this method indoors only).

For each method, only the illumination estimation stage of our pipeline changes (depth, reflectance, and camera estimation remain the same), and all methods utilize our lighting optimization technique to estimate source intensity (Sec 5.3).

Conditions. Because we found no condition in our initial study to be a useful predictor of people's ability to choose the real image, we introduce many new (and perhaps more telling) conditions in this study: **indoor/outdoor** scene, **diffuse/direct lighting** in the scene, **simple/complex material** of inserted object, **good/poor composition** of inserted object, and if the inserted object has **good/poor perspective**. We also assigned each subject a set of binary conditions: **male/female**, **age** < 25 / ≥ 25, **color-normal / not color-normal**, whether or not the subject correctly identified both the physically accurate sphere *and* the proper-perspective line drawing at the end of the study (**passed/failed perspective-shading (p-s) tests**), and also **expert/non-expert** (subjects were classified as experts only if they passed the perspective-shading tests *and* indicated that they had expertise in art/graphics).

Results and discussion. Overall, our synthetic image study showed that people confused our insertion result with the true rendered image in over 35% of 1776 viewed image pairs (an optimal result would be 50%). We also achieve better confusion rates than the methods we compared to, and Table I shows these results.

Table I. Fraction of times subjects chose a synthetic insertion result over the ground truth insertion in our user study

Condition	N	ours	Khan	Lalonde/matching
indoor	1332	**.377**	.311	.279
outdoor	444	.288	**.297**	.240
diffuse light	1184	**.377**	.332	.238
directional light	592	**.311**	.257	.330
simple material	1036	**.353**	.301	.248
complex material	740	**.357**	.315	.299
large coverage	1332	**.355**	.324	.277
small coverage	444	**.356**	.253	.245
good composition	1036	**.356**	.312	.265
poor composition	740	**.353**	.300	.274
good perspective	1036	**.378**	.316	.287
poor perspective	740	**.322**	.295	.244
male	1032	**.346**	.295	.267
female	744	**.367**	.319	.272
age ($\leq$25)	468	**.331**	.294	.234
age (>25)	1164	**.363**	.312	.284
color normal	1764	**.353**	.308	.267
not color normal	12	**.583**	.167	.361
passed p-s tests	1608	**.341**	.297	.260
failed p-s tests	168	.482	**.500**	.351
non-expert	1392	**.361**	.312	.275
expert	384	**.333**	.288	.252
overall	1776	**.355**	.307	.269

Highlighted blocks indicate that there are significant differences in confusion when a particular condition is on/off (p value < 0.05 using a 2-tailed t-test). For example, in the top left cell, the "ours indoor" distribution was found to be significantly different from the "ours outdoor" distribution. For the Lalonde/matching column, the method Lalonde et al. is used for outdoor images, and the template matching technique is used indoors (see text for details). The best method for each condition is shown in bold. N is the total number of samples. Overall, our confusion rate of 35.5% better than the baselines (30.7% and 26.9%) by statistically significant margins.

Although the differences between our method and the other methods might appear smaller (~5-10 percentage points), these differences are statistically significant using a two-tailed test. Furthermore, we are only assessing differences in light estimation among these method (since it is our primary technical contribution); every other part of our pipeline remains constant (e.g. camera and depth estimation, as well as the light intensity optimization). In other words, we do not compare directly to other techniques, rather these other lighting techniques are aided by bootstrapping them with the rest of our pipeline.

We also analyze which conditions lead to significant differences in confusion for each method, indicated by highlighted cells in the table. As expected, our method works best indoors and when the lighting is not strongly directed, but still performs reasonably well otherwise (over 31% confusion). Also, our method looks much more realistic when inserted objects have reasonable perspective.

Perhaps most interestingly, our method is almost confusable at chance with ground truth for people who have a hard time detecting shading/perspective problems in photographs (failed p-s tests), and the population that passed is significantly better at this task than those that failed. This could mean that such a test is actually a very good predictor for whether a person will do well at this task, or it could be viewed as a secondary "qualification" test (i.e. similar yet possibly more difficult than the original cartoon-real image qualifi-

[5]Image pairs are randomly permuted/selected so that each of the 12 objects appears exactly once to each subject.

cation tests). Either way, for the population that passed these tests, the confusion rates are very similar to the overall mean (i.e. results are very similar even if this data is disregarded).

All other conditions did not have a significant impact on the confusion rate for our method. Again, we observe that the "expert" subjects did not perform significantly better (p value = 0.323) than "non-expert" subjects (by our classification), although they did choose the ground truth image more times on average.

7.3 Ground truth comparison

We also measure the accuracy of our scene estimates by comparing to ground truth. In Figure 8, we show three images from our "synthetic" user study: rendered images (ground truth) are compared with insertion results using our technique as well as results relit using the method of Khan et al. [2006] (but our method is used for all other components, e.g. depth, light optimization, and reflectance).

Figure 9 shows our inverse rendering estimates (depth, reflectance, and illumination) for the same three scenes as in Figure 8. These images illustrate typical errors produced by our algorithm: depth maps can be confused by textured surfaces and non-planar regions, reflectance estimates may fail in the presence of hard shadows and non-Lambertian materials, and illumination maps can contain inaccurate lighting directions or may not appear similar to the actual light reflected onto the scene. These downfalls suggest future research directions for single image inference.

Notice that while in some cases our estimates differ greatly from ground truth, our insertion results can be quite convincing, as demonstrated by our user studies. This indicates that the absolute physical accuracy of inverse rendering is not a strong indicator of how realistic an inserted object will look; rather, we believe that relative cues are more important. For example, inserted objects typically cast better shadows in regions of planar geometry and where reflectance is constant and relatively correct (i.e. erroneous up to a scale factor); strong directional illuminants need not be perfectly located, but should be consistent with ambient light. Nonetheless, 3D object insertion will likely benefit from improved estimates.

For quantitative errors and additional results, we refer the reader to supplemental material.

8. RESULTS AND CONCLUSION

We show typical results produced by our system in Fig 10. Our method is applicable both indoors and outdoors and for a variety of scenes. Although a Manhattan World is assumed at various stages of our algorithm, we show several reasonable results even when this assumption does not hold. Failure cases are demonstrated in Fig 11. Many additional results (varying in quality and scene type) can be found in the supplemental document. The reader is also referred to the accompanying video for a demonstration of our system.

As we found in our user study, our method is better suited for indoor scenes or when light is not strongly directional. In many cases, people confused our insertion results as real pictures over one third of the time. For outdoor scenes, we found that simpler illumination methods might suffice (e.g. [Khan et al. 2006]), although our geometry estimates are still useful for these scenes (to estimate light intensity and to act as shadow catchers).

We would like to explore additional uses for our scene models, such as for computer gaming and videos. It would be interesting to try other physically grounded editing operations as well; for example, deleting or moving objects from a scene, or adding physically-based animations when inserting objects (e.g., dragging a table cloth over a table). Extending our method to jointly infer a

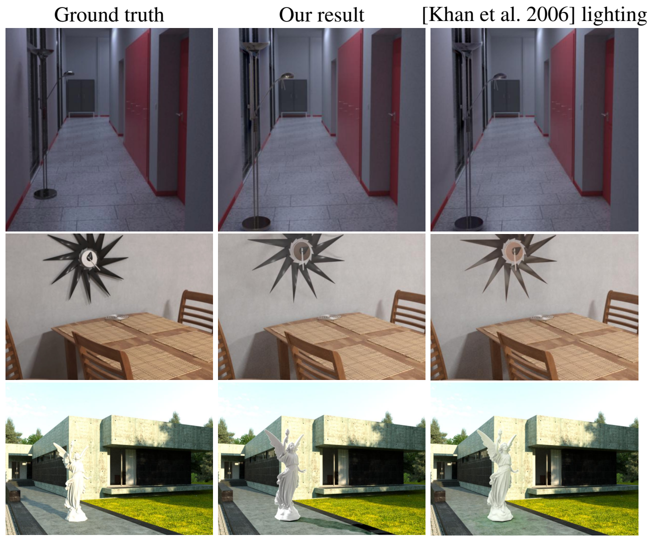

Fig. 8. Example results used in our synthetic user study (corresponding to the scenes shown in Figure 9). We compared the ground truth renderings (left) to our insertion method (middle) and others (e.g. using the method of Khan et al. for relighting, right). *Scene modeling credits (top to bottom): ©Simon Wendsche, ©Peter Sandbacka, ©Eduardo Camara.*

scene all at once (rather than serially) could lead to better estimates. Our depth estimates might also be useful for inferring depth order and occlusion boundaries.

Judging by our synthetic image user study, our method might also be useful as a fast, incremental renderer. For example, if a 3D modeler has created and rendered a scene, and wants to insert a new object quickly, our method could be used rather than re-rendering the full scene (which, for most scenes, should incur less render time since our estimated scene may contain significantly fewer light sources and/or polygons).

We have presented a novel image editor that, unlike most image editors that operate in 2D, allows users to make physically meaningful edits to an image with ease. Our software supports realistic object insertion, on-the-fly lighting changes, post-process depth of field modifications, and is applicable to legacy, LDR images. These interactions are facilitated by our automatic scene inference algorithm, which encompasses our primary technical contributions: single image depth estimation, and data-driven illumination inference. Results produced by our system appear realistic in many cases, and a user study provides good evidence that objects inserted with our fully automatic technique look more realistic than corresponding real photographs over one third of the time.

ACKNOWLEDGMENTS

This work was funded in part by the NSF GRFP, NSF Award IIS 0916014, ONR MURI Award N00014-10-10934, and an NSF Expeditions Award, IIS-1029035. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or of the Office of Naval Research.

REFERENCES

ACHANTA, R., SHAJI, A., SMITH, K., LUCCHI, A., FUA, P., AND SSSTRUNK, S. 2012. Slic superpixels compared to state-of-the-art su-

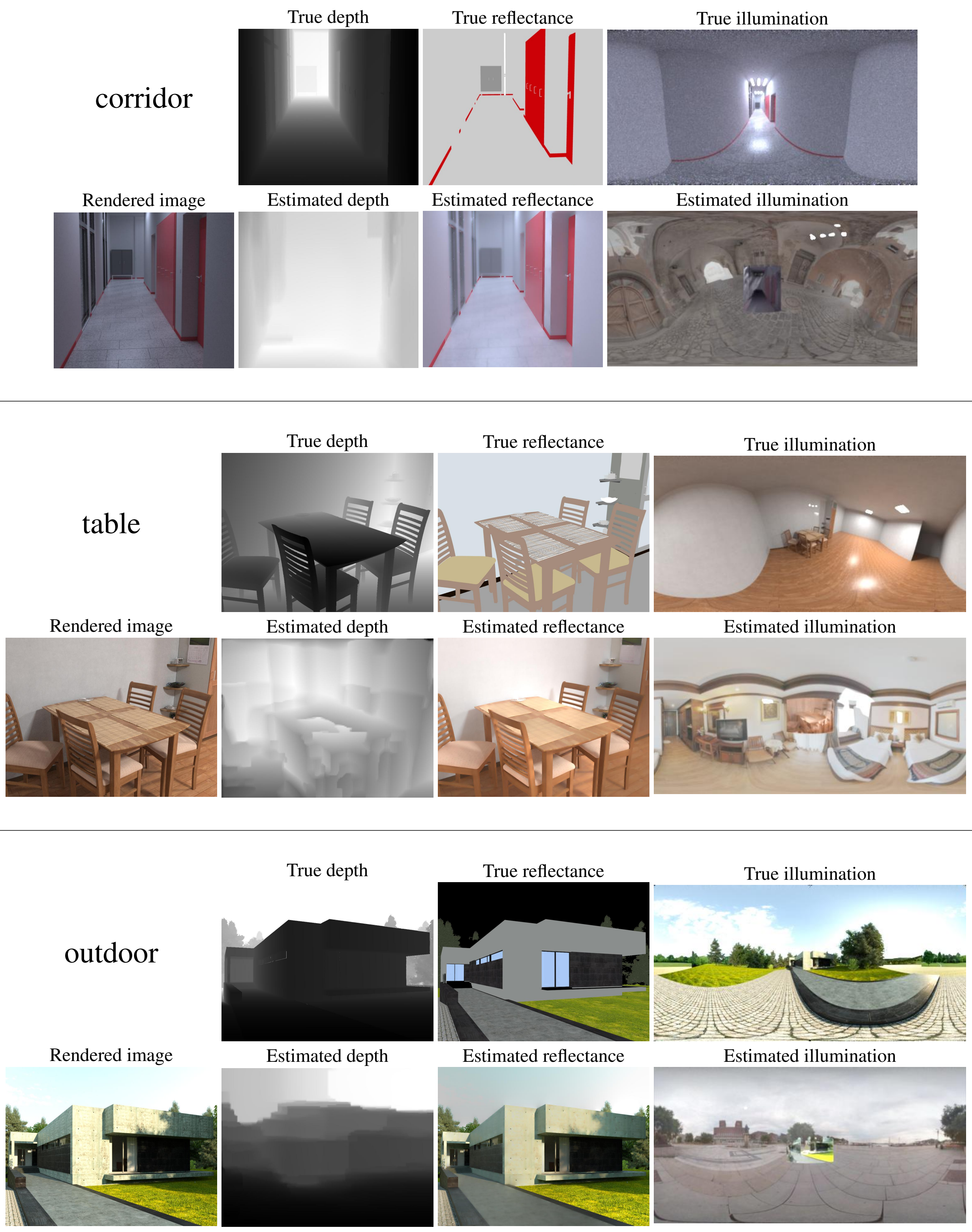

Fig. 9. Comparison of ground truth scene components (depth, diffuse reflectance, and illumination) with our estimates of these components. Illumination maps are tonemapped for display. See Figure 8 and the supplemental document for insertion results corresponding to these scenes. *Scene modeling credits (top to bottom): ©Simon Wendsche, ©Peter Sandbacka, ©Eduardo Camara.*

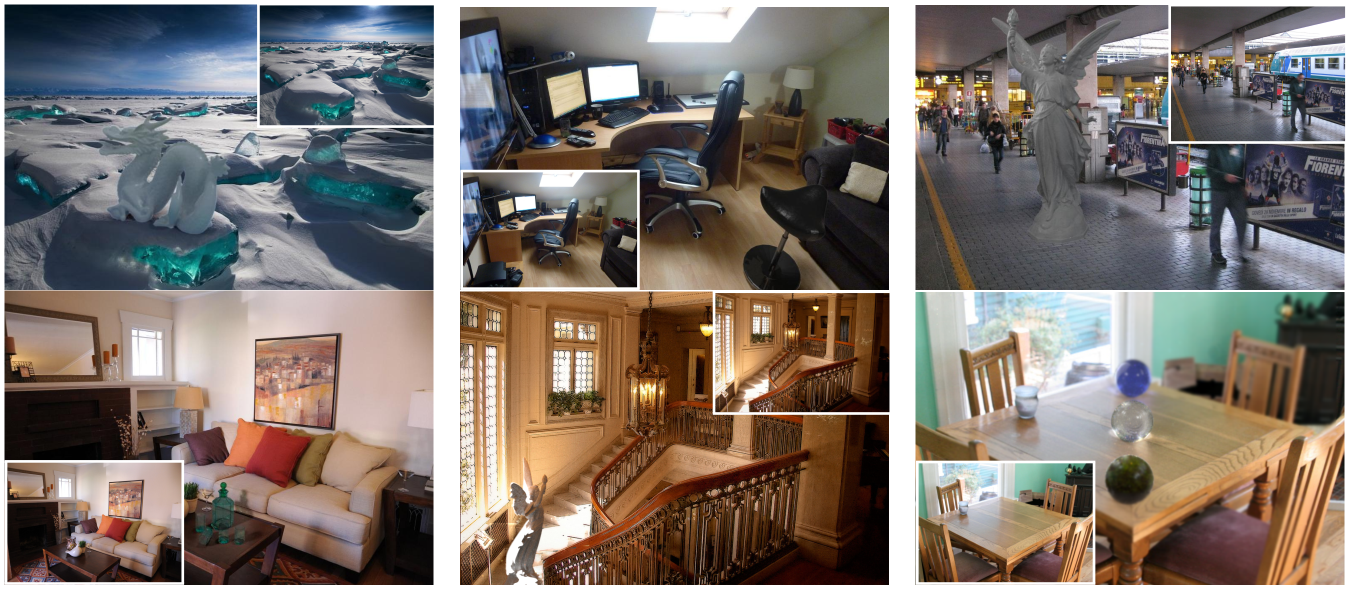
Fig. 10. Additional results produced by our method (original picture is overlaid on top of the result). Our method is capable of producing convincing results both indoors and outdoors, and can be robust to non-manhattan scenes. Our method can be applied to arbitrary images, and makes no explicit assumptions about the scene geometry; virtual staging is one natural application of our system. Best viewed in color at high-resolution. *Photo credits (left to right, top to bottom): ©Alexey Trofimov, ©Sean MacEntee, Flickr users ©"AroundTuscany," ©"Wonderlane," ©"PhotoAtelier," and ©Brian Teutsch.*

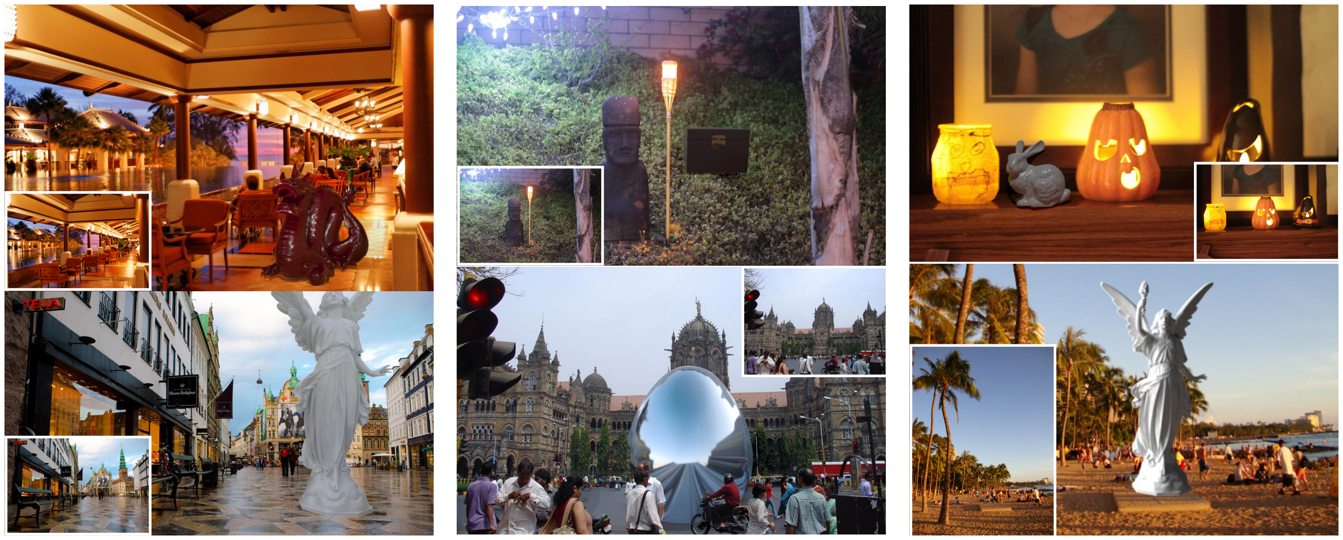
Fig. 11. Failure cases. In some images, placement can be difficult due to inaccurate geometry estimates (top row), e.g. the dragon's shadow (top left) is not visible due to poor depth estimates, and the chest (top middle) rests on a lush, slanted surface, and our depth is flat in the region of the chest. Insertions that affect large portions of the image (e.g. statue; bottom left) or reflect much of the estimated scene (e.g. curved mirrors; bottom middle) can sometimes exacerbate errors in our scene estimates. Light estimates and color constancy issues lead to less realistic results (top right), especially outdoors (bottom right). Note that the people in the bottom right image have been manually composited over the insertion result. *Photo credits (left to right, top to bottom): ©Dennis Wong, ©Parker Knight, ©Parker Knight, ©Kenny Louie, Flickr user ©"amanderson2," and ©Jordan Emery.*

perpixel methods. *IEEE TPAMI 34,* 11, 2274–2282.

Barron, J. T. and Malik, J. 2013. Intrinsic scene properties from a single rgb-d image. *CVPR*.

Boivin, S. and Gagalowicz, A. 2001. Image-based rendering of diffuse, specular and glossy surfaces from a single image. *ACM SIGGRAPH*.

Boyadzhiev, I., Paris, S., and Bala, K. 2013. Example-based synthesis of 3d object arrangements. In *SIGGRAPH*.

Criminisi, A., Reid, I., and Zisserman, A. 2000. Single view metrology. *International Journal of Computer Vision 40,* 2 (Nov), 123–148.

Debevec, P. 1998. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. *ACM SIGGRAPH*.

Debevec, P. 2005. Making "the parthenon". In *International Symposium* *on Virtual Reality, Archaeology, and Cultural Heritage*.

DELAGE, E., LEE, H., AND NG, A. Y. 2005. Automatic single-image 3d reconstructions of indoor manhattan world scenes. In *ISRR*. 305–321.

DIAZ, M. AND STURM, P. 2013. Estimating photometric properties from image collections. *Journal of Mathematical Imaging and Vision*.

DROR, R. O., WILLSKY, A. S., AND ADELSON, E. H. 2004. Statistical characterization of real-world illumination. *J Vis 4,* 9, 821–837.

FURUKAWA, Y., CURLESS, B., SEITZ, S. M., AND SZELISKI, R. 2009. Manhattan-world stereo. In *CVPR*. IEEE, 1422–1429.

GALLUP, D., FRAHM, J.-M., AND POLLEFEYS, M. 2010. Piecewise planar and non-planar stereo for urban scene reconstruction. In *CVPR*.

GIBSON, S. AND MURTA, A. 2000. Interactive rendering with real-world illumination. In *EGSR*. Springer-Verlag, London, UK, UK, 365–376.

GROSSE, R., JOHNSON, M. K., ADELSON, E. H., AND FREEMAN, W. 2009. Ground truth dataset and baseline evaluations for intrinsic image algorithms. *ICCV*.

HARTLEY, R. AND ZISSERMAN, A. 2003. *Multiple view geometry in computer vision*.

HEDAU, V., HOIEM, D., AND FORSYTH, D. 2009. Recovering the Spatial Layout of Cluttered Rooms. *ICCV*.

HOIEM, D., EFROS, A., AND HEBERT, M. 2005a. Geometric context from a single image. In *ICCV*. Vol. 1. 654–661.

HOIEM, D., EFROS, A. A., AND HEBERT, M. 2005b. Automatic photo pop-up. *ACM Trans. Graph. 24,* 3 (July), 577–584.

HORRY, Y., ANJYO, K.-I., AND ARAI, K. 1997. Tour into the picture: using a spidery mesh interface to make animation from a single image. In *SIGGRAPH*.

INTRAUB, H. AND RICHARDSON, M. 1989. Wide-angle memories of close-up scenes. *Journal of experimental psychology. Learning, memory, and cognition 15,* 2 (Mar.), 179–187.

JOACHIMS, T. 2006. Training linear svms in linear time. In *KDD*. 217–226.

JOHNSON, M. K. AND FARID, H. 2005. Exposing digital forgeries by detecting inconsistencies in lighting. In *Workshop on Multimedia and Security*.

JOHNSON, M. K. AND FARID, H. 2007. Exposing digital forgeries in complex lighting environments. *IEEE Transactions on Information Forensics and Security 2,* 3, 450–461.

KARSCH, K., HEDAU, V., FORSYTH, D., AND HOIEM, D. 2011. Rendering synthetic objects into legacy photographs. In *SIGGRAPH Asia*. 157:1–157:12.

KARSCH, K., LIU, C., AND KANG, S. B. 2012. Depth extraction from video using non-parametric sampling. In *ECCV*.

KHAN, E. A., REINHARD, E., FLEMING, R. W. W., AND BÜLTHOFF, H. H. 2006. Image-based material editing. In *ACM SIGGRAPH*.

LALONDE, J., EFROS, A. A., AND NARASIMHAN, S. 2009. Estimating Natural Illumination from a Single Outdoor Image. *ICCV*.

LALONDE, J., HOIEM, D., EFROS, A. A., AND ROTHER, C. 2007. Photo clip art. *ACM SIGGRAPH*.

LAZEBNIK, S., SCHMID, C., AND PONCE, J. 2006. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*. 2169–2178.

LEE, D. C., HEBERT, M., AND KANADE, T. 2009. Geometric reasoning for single image structure recovery. In *CVPR*. 2136–2143.

LIU, B., GOULD, S., AND KOLLER, D. 2010. Single image depth estimation from predicted semantic labels. In *CVPR*. 1253–1260.

LOMBARDI, S. AND NISHINO, K. 2012a. Reflectance and natural illumination from a single image. In *CVPR*.

LOMBARDI, S. AND NISHINO, K. 2012b. Single image multimaterial estimation. In *CVPR*.

LOPEZ-MORENO, J., HADAP, S., REINHARD, E., AND GUTIERREZ, D. 2010. Compositing images through light source detection. *Computers & Graphics 34,* 6, 698–707.

LOSCOS, C., FRASSON, M.-C., DRETTAKIS, G., WALTER, B., GRANIER, X., AND POULIN, P. 1999. Interactive virtual relighting and remodeling of real scenes. In *EGSR*. 329–340.

NIMEROFF, J. S., SIMONCELLI, E., AND DORSEY, J. 1994. Efficient Re-rendering of Naturally Illuminated Environments. In *EGSR*. 359–373.

NISHINO, K. AND NAYAR, S. K. 2004. Eyes for Relighting. *ACM Transactions on Graphics (also Proc. of SIGGRAPH) 23,* 3 (July), 704–711.

NOCEDAL, J. AND WRIGHT, S. J. 2006. *Numerical Optimization*, 2nd ed. Springer, New York.

OH, B. M., CHEN, M., DORSEY, J., AND DURAND, F. 2001. Image-based modeling and photo editing. In *SIGGRAPH*. 433–442.

OLIVA, A. AND TORRALBA, A. 2001. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision 42,* 3, 145–175.

PANAGOPOULOS, A., WANG, C., SAMARAS, D., AND PARAGIOS, N. 2011. Illumination estimation and cast shadow detection through a higher-order graphical model. In *CVPR*. 673 –680.

PERO, L. D., BOWDISH, J., HARTLEY, E., KERMGARD, B., AND BARNARD, K. 2013. Understanding bayesian rooms using composite 3d object models. In *CVPR*.

PHARR, M. AND HUMPHREYS, G. 2010. *Physically Based Rendering, Second Edition: From Theory To Implementation*, 2nd ed. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

RAMAMOORTHI, R. AND HANRAHAN, P. 2004. A signal-processing framework for reflection. *ACM Trans. Graph. 23,* 4 (Oct.), 1004–1042.

RAMANARAYANAN, G., FERWERDA, J. A., WALTER, B., AND BALA, K. 2007. Visual equivalence: towards a new standard for image fidelity. *ACM Trans. Graph. 26,* 3.

ROMEIRO, F., VASILYEV, Y., AND ZICKLER, T. 2008. Passive reflectometry. In *ECCV*.

ROMEIRO, F. AND ZICKLER, T. 2010. Blind reflectometry. In *ECCV*.

SATKIN, S., LIN, J., AND HEBERT, M. 2012. Data-driven scene understanding from 3D models. In *Proceedings of the 23rd British Machine Vision Conference*.

SAXENA, A., SUN, M., AND NG, A. Y. 2009. Make3D: Learning 3D Scene Structure from a Single Still Image. *IEEE Transactions on Pattern Analysis and Machine Intelligence 31,* 5, 824–840.

SCHOENEMAN, C., DORSEY, J., SMITS, B., ARVO, J., AND GREENBERG, D. 1993. Painting with light. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*. SIGGRAPH '93. ACM, New York, NY, USA, 143–146.

SCHWING, A. G. AND URTASUN, R. 2012. Efficient exact inference for 3d indoor scene understanding. In *ECCV (6)*. 299–313.

TIBSHIRANI, R. 1996. Regression shrinkage and selection via the lasso. *J. Roy. Stat. Soc. B 58,* 1, 267–288.

XIAO, J., EHINGER, K. A., OLIVA, A., AND TORRALBA, A. 2012. Recognizing scene viewpoint using panoramic place representation. In *CVPR*.

YU, Y., DEBEVEC, P., MALIK, J., AND HAWKINS, T. 1999. Inverse global illumination: recovering reflectance models of real scenes from photographs. In *ACM SIGGRAPH*.

ZANG, A. R., FELINTO, D., AND VELHO, L. 2012. Augmented reality using full panoramic captured scene light-depth maps. In *SIGGRAPH Asia 2012 Posters*. 28:1–28:1.

ZHANG, Y., XIAO, J., HAYS, J., AND TAN, P. 2013. Framebreak: Dramatic image extrapolation by guided shift-maps. In *CVPR*.