Rich Intrinsic Image Separation for Multi-View Outdoor Scenes

Pierre-Yves Laffont, Adrien Bousseau, George Drettakis

Rich Intrinsic Image Separation for Multi-View Outdoor Scenes

Pierre-Yves Laffont, Adrien Bousseau, George Drettakis

RESEARCH REPORT

N° 7851

December 2011

Project-Teams REVES

ISSN 0249-6399 ISRN INRIA/RR--7851--FR+ENG

Rich Intrinsic Image Separation for Multi-View Outdoor Scenes

Pierre-Yves Laffont, Adrien Bousseau, George Drettakis

Project-Teams REVES

Research Report n° 7851 — December 2011 — 28 pages

Abstract: Intrinsic images aim at separating an image into its reflectance and illumination components to facilitate further analysis or manipulation. This separation is severely ill-posed and the most successful methods rely on user indications or precise geometry to resolve the ambiguities inherent to this problem. In this paper we propose a method to estimate intrinsic images from multiple views of an outdoor scene without the need for precise geometry or involved user intervention. We use multiview stereo to automatically reconstruct a 3D point cloud of the scene. Although this point cloud is sparse and incomplete, we show that it provides the necessary information to compute plausible sky and indirect illumination at each 3D point. We then introduce an optimization method to estimate sun visibility over the point cloud. This algorithm compensates for the lack of accurate geometry and allows the extraction of precise shadows in the final image. We finally propagate the information computed over the sparse point cloud to every pixel in the photograph using image-guided propagation. Our propagation not only separates reflectance from illumination, but also decomposes the illumination into a sun, sky and indirect layer. This rich decomposition allows novel image manipulations as demonstrated by our results.

Key-words: Intrinsic images, image-guided propagation, multi-view stereo, mean-shift algorithm

RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Décomposition en images intrinsèques riches à partir de plusieurs vues d'une scène extérieure

Résumé : Nous présentons une méthode capable de décomposer les photographies d'une scène en trois composantes intrinsèques — la réflectance, l'illumination due au soleil, l'illumination due au ciel, et l'illumination indirecte. L'extraction d'images intrinsèques à partir de photographies est un problème difficile, généralement résolu en utilisant des méthodes de propagation guidée par l'image nécessitant de multiples indications utilisateur. Des méthodes récentes en vision par ordinateur permettent l'acquisition facile mais approximative d'informations géométriques d'une scène à l'aide de plusieurs photographies selon des points de vue différents. Nous développons un nouvel algorithme qui nous permet d'exploiter cette information bruitée et peu fiable pour automatiser et améliorer les algorithmes d'estimation d'images intrinsèque par propagation. En particulier, nous développons une nouvelle approche par optimisation afin d'estimer les ombres portées dans l'image, en peaufinant une estimation initiale obtenue à partir des informations géométriques reconstruites. Dans une dernière étape nous adaptons les algorithmes de propagation guidée par l'image, en remplaçant les indications utilisateurs manuelles par les données d'ombre et de réflectance déduite du nuage de points 3D par notre algorithme. Notre méthode permet l'extraction automatique des images intrinsèques à partir de multiples points de vue, permettant ainsi de nombreux types de manipulations d'images.

Mots-clés : Images intrinsèques, propagation guidée par l'image, stereo multi-vues, algorithm mean-shift

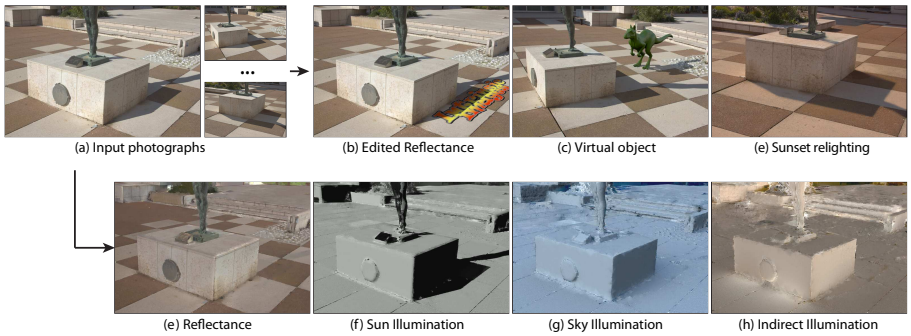


Figure 1: Starting from multiple views of the scene (a), our method decomposes photographs into four intrinsic components — the reflectance (e), the illumination due to sun (f), the illumination due to sky (g) and the indirect illumination (h). Each intrinsic component can then be manipulated independently for advanced image editing applications (b-d).

1 Introduction

Editing material properties and lighting conditions is a common image manipulation task that requires significant expertise to achieve consistent results. The main difficulty in such manipulations resides in the fact that a pixel color aggregates the effect of both material and lighting, so that standard color manipulations are likely to affect both components.

A typical usage scenario would be to take an interesting photograph, and manipulate its content *after* the capture, e.g., to change the color of an object, modify the lighting of the scene, or add in virtual objects. In this paper we present a method to achieve this just by taking a few extra photographs, which represents a minimal additional cost. We focus on outdoor scenes and separate photographs into a material layer (also called *reflectance*) and several *illumination* layers that describe the contributions of sun, sky and indirect lighting (Fig. 1e-h). This *intrinsic images* decomposition [1] allows easy editing of each component separately, and subsequent compositing with consistent lighting. We illustrate applications of our decomposition in Fig. 1. We first alter the floor material with a graffiti while maintaining consistent shadows (b); we then add a virtual object in the scene with consistent lighting (c); and we finally change the lighting color and blur the shadows to simulate sunset (d). Please also see the video which illustrates the image manipulation process.

Estimating intrinsic images from photographs is an ill-posed problem that has been traditionally addressed with restrictive assumptions on reflectance and illumination variations [22], significant user intervention [2], or multiple images under varying lighting [40] captured over several hours for outdoor scenes. In addition, most existing methods only generate a single illumination layer, grouping the effects of sun, sky and indirect lighting. Inverse global illumination methods [42, 9] also estimate the material properties of a scene but require very accurate geometric models, either built by hand or acquired with complex laser scanners.

In our approach we address the above shortcomings by providing an illu-

mination decomposition with minimal user intervention, resulting in intrinsic image layers for sun, sky and indirect illumination. We achieve this by combining sparse geometric reconstruction [37, 15] with image-guided propagation [23], thus leveraging their respective strengths. We exploit the automatically reconstructed 3D information to compute lighting information for a subset of pixels, and use image-guided propagation to decompose the photographs into intrinsic images.

Our algorithm takes as input a small number of photographs of the scene captured *at a single time of the day*, along with an environment map. From this lightweight capture we use recent computer vision algorithms to reconstruct a sparse 3D point cloud of the scene. The point cloud only provides an imprecise and incomplete representation of the scene. However, we show that this is sufficient to compute plausible sky and indirect illumination at each reconstructed 3D point, without complex inverse global illumination computations. The coarse geometry is however unreliable for sun illumination that typically contains high-frequency features such as cast shadows. We introduce a new parameterization of reflectance with respect to sun visibility that we integrate in an optimization algorithm to robustly identify the 3D points that are in shadow. We developed an optimization inspired by mean shift [7] where we use asymmetric regions of influence and constrain the evolution of the estimates.

Image-guided propagation algorithms are typically used to propagate user scribbles [23, 2]; we show how to use these algorithms to propagate the illumination information computed at 3D points over the image pixels. Our approach generates intrinsic images of similar quality as scribble-based approaches, with only a small amount of user intervention for capture and calibration. In addition, our ability to separate sun, sky and indirect illumination opens the door for advanced image manipulations, as demonstrated in Fig. 1b-d.

To summarize, this paper makes the following contributions:

- We show how to compute sky, indirect, and sun (ignoring cast shadows) illumination at automatically reconstructed 3D points, using incomplete and imprecise geometry and a small set of input images.

- We introduce an algorithm to reliably identify points in shadow based on a new parameterization of the reflectance with respect to sun visibility. Our algorithm compensates for the lack of accurately reconstructed and complete 3D information.

- We show how to propagate reflectance, sun, sky and indirect illumination to all pixels in an image, without user intervention or involved inverse global illumination computation. We achieve this by using the illumination values computed at 3D points as constraints for image propagation algorithms.

After a presentation of previous work, the definition of our image formation model and a description of capture, the structure of our paper follows the three contributions described above.

2 Related Work

Intrinsic images. Several methods have been proposed to estimate reflectance and illumination from a single image. This decomposition is severely ill-posed and can only be solved with additional knowledge or assumptions about the scene content. The Retinex algorithm [22, 18] assumes smooth illumination and piece-wise constant reflectance, while Li and Yeo [25] assume that neighboring pixels with similar chromaticity have the same reflectance and that the image is composed of a small set of reflectances. Tappen et al. [39] train a classifier to discriminate image derivatives due to reflectance and illumination, and Shen et al. [35] introduce texture constraints to ensure that pixels with similar texture share the same reflectance. These various approaches produce encouraging decompositions on isolated objects, as evaluated by the ground-truth dataset of Grosse et al. [16]. However, the automatic decomposition of complex outdoor images remains an open challenge, in part because most existing methods assume monochrome lighting while outdoor scenes are often lit by a mixture of colored sun and sky light.

Weiss [40] demonstrates how multiple images of a scene under different illuminations can be factored into a reflectance image and a set of illumination images. Sunkavalli et al. [38] decompose similar image sequences of outdoor scenes into a shadow mask and images illuminated only by skylight or sunlight. By also capturing an environment map at multiple times of the day, Matusik et al. [28] estimate the reflectance field of an outdoor scene that can then be used for relighting. These methods assume a fixed viewpoint and varying illumination (i.e., *timelapse* sequences), while our method relies on images captured under different viewpoints and fixed illumination. The main advantage of our capture approach is to reduce the acquisition time to a few minutes while timelapses typically require at least several hours of capture to cover as many lighting directions as possible. Most related to our capture strategy is the system of Melendez et al. [29] that relights buildings reconstructed from multiple photographs. However their method necessitates the additional acquisition of flash / no-flash image pairs to capture a *material exemplar* for every material of a building. Users then need to associate a material exemplar to each texture region of the reconstructed building.

Bousseau et al. [2] and Shen et al. [34] rely on user scribbles to disambiguate reflectance from illumination while Okabe et al. [31] propagate a sparse set of user-specified normals over the image. Given an intrinsic image decomposition, Carroll et al. [5] propose a user-assisted decomposition to isolate the indirect contribution of each colored material in the scene. Our method shares similarities with these user-assisted approaches but we propagate illumination values automatically computed from a sparse point cloud instead of user indications.

Inverse rendering. Inverse rendering methods [43, 42, 27, 9] recover the reflectance and illumination of a scene by inverting the rendering equation. These methods require an accurate 3D model of the scene that is either modeled manually or acquired with expensive laser scanners and then solve an often costly global illumination system. In contrast our method is robust to incomplete geometry and handles sparse point clouds automatically reconstructed from a few photographs of the scene. In addition, we are able to estimate indirect illumination at every pixel without the need to solve inverse global illumination. Nonetheless, inverse rendering methods allow applications that are beyond the

scope of this paper, such as free viewpoint navigation and dynamic relighting.

Image-based propagation. Image guided interpolation methods have been introduced by Levin and Lischinski to propagate colors [23] and tonal adjustments [26]. In this paper we use the propagation algorithm of Bousseau et al. [2] that was originally designed to propagate user indications for intrinsic image decompositions. This algorithm is inspired by the *matting Laplacian* of Levin and colleagues [24] that has been used to decompose an image into a foreground and background layer, and to recover white balanced images under mixed lighting [19].

Shadow removal. Our work is also related to shadow removal methods [13, 12, 30, 41, 36] that aim at identifying and removing cast shadows in an image, either automatically or with user assistance. While our method is also able to identify cast shadows, our main goal is to extract a reflectance image, as well as smooth illumination variations. We also separate the contribution of sun, sky and indirect illumination, which enables novel image manipulations.

3 Image Formation Model

We assume Lambertian surfaces and model the image values at each pixel as the product between the incident illumination and the object reflectance $\mathbf{R}$. Formally, the radiance $\mathbf{I}$ towards the camera at each non-emissive, visible point corresponding to a pixel is given by the rendering equation [20]:

$$\mathbf{I} = \mathbf{R} * \int_{\Omega} \cos\theta_{\omega} \mathbf{L}(\omega) d\omega \tag{1}$$

where we integrate over the entire hemisphere Ω at the visible point, $\mathbf{L}(\omega)$ is the incoming radiance in direction ω, θ_{ω} is the angle between the normal at the visible point and direction ω. Capital bold letters represent RGB color values and $*$ denotes per-channel multiplication.

For our purposes, we will separate out the incoming radiance into three components: the radiance due to the sun, that due to the sky and that due to indirect lighting. To simplify notation, we define two subsets of the hemisphere: Ω_{sky}, i.e., the subset of directions in which the visible point sees the sky, and Ω_{ind} the subset of directions in which another object is visible, and thus contributes to indirect lighting. We however explicitly represent the sun visibility v_{sun}, first because precise computation of v_{sun} is necessary to capture sharp shadows, and second because estimating v_{sun} robustly is one of our main contributions.

We can now re-write Equation 1:

$$\mathbf{I} = \mathbf{R} * \Big(v_{\text{sun}} \max(0, \cos\theta_{\text{sun}}) \mathbf{L}_{\text{sun}} + \int_{\Omega_{\text{sky}}} \cos\theta_{\omega} \mathbf{L}_{\text{sky}}(\omega) d\omega + \int_{\Omega_{\text{ind}}} \cos\theta_{\omega} \mathbf{L}_{\text{ind}}(\omega)\ d\omega \Big)$$

where $\mathbf{L}_{\text{sun}}$, $\mathbf{L}_{\text{sky}}$ and $\mathbf{L}_{\text{ind}}$ are radiance from the sun, the sky and indirect lighting respectively, θ_{sun} is the angle between the normal at the visible point and the sun modeled as a directional light source, and θ_{ω} is the angle between the normal and the direction of integration ω over the corresponding hemisphere. The scalar $v_{\text{sun}} \in [0, 1]$ models the visibility of the sun (0 for completely hidden, 1 for fully visible).

We next define the following simplified quantities at each pixel:

$$\mathbf{S}_{\text{sun}} = v_{\text{sun}} \max(0, \cos\theta_{\text{sun}})\mathbf{L}_{\text{sun}} = v_{\text{sun}}\hat{\mathbf{S}}_{\text{sun}} \quad (2)$$

$$\mathbf{S}_{\text{sky}} = \int_{\Omega_{sky}} \cos\theta_\omega \mathbf{L}_{\text{sky}}(\omega)\, d\omega \quad (3)$$

$$\mathbf{S}_{\text{ind}} = \int_{\Omega_{ind}} \cos\theta_\omega \mathbf{L}_{\text{ind}}(\omega)\, d\omega. \quad (4)$$

where $\hat{\mathbf{S}}_{\text{sun}}$ corresponds to the sun illumination when cast shadows are ignored.

This allows us to define a simplified image formation model, which we use from now on:

$$\mathbf{I} = \mathbf{R} * (\mathbf{S}_{\text{sun}} + \mathbf{S}_{\text{sky}} + \mathbf{S}_{\text{ind}}) \quad (5)$$

$$= \mathbf{R} * \mathbf{S}_{\text{total}} \quad (6)$$

where $\mathbf{R}$ is the object RGB reflectance. $\mathbf{S}_{\text{sun}}$, $\mathbf{S}_{\text{sky}}$ and $\mathbf{S}_{\text{ind}}$ are the RGB incident illumination (or shading) from the sun, sky and indirect lighting respectively.

Our first goal is to extract the reflectance $\mathbf{R}$ and illumination $\mathbf{S}_{\text{total}}$ from this image formation model. We demonstrate how to make this problem tractable by leveraging the sparse geometric information generated by multiview stereo algorithms to compute $\hat{\mathbf{S}}_{\text{sun}}$, $\mathbf{S}_{\text{sky}}$ and $\mathbf{S}_{\text{ind}}$ at each reconstructed 3D point (Section 5). We then introduce a new algorithm to estimate the sun visibility v_{sun} precisely despite the approximate available geometry (Section 6), which will allow us to obtain all illumination components using image propagation (Section 7).

4 Capture and Reconstruction

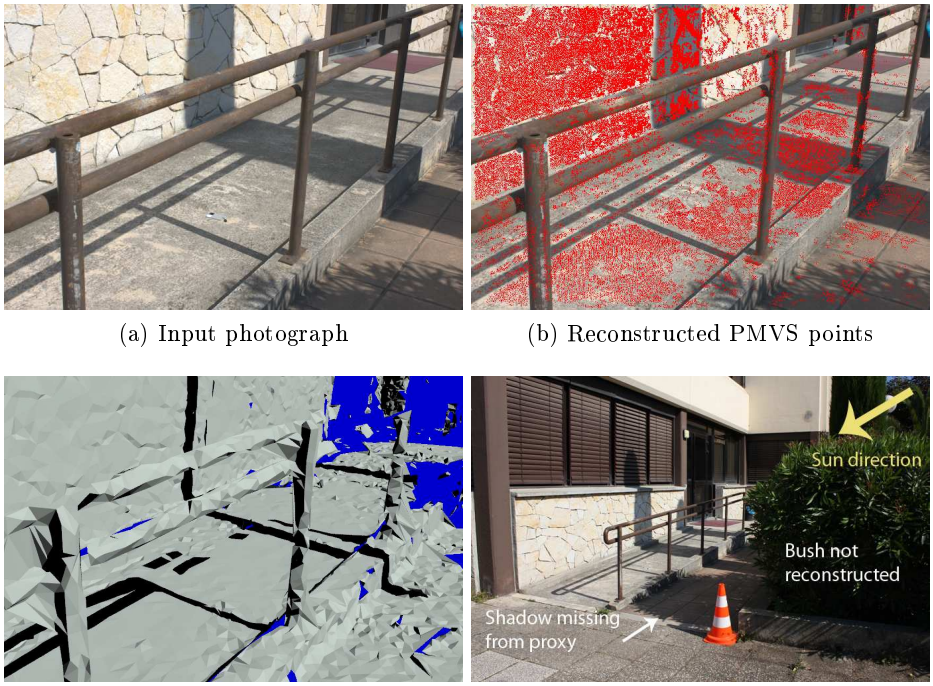
Our method relies on a relatively lightweight capture setup composed of a digital camera (preferably on a tripod), a photographer's gray card and a simple reflective sphere[1] to capture the environment map (see inset). No other special capture or measurement hardware is required.

4.1 Photography

We first capture a few ordinary low-dynamic range photographs (LDR) which we use to perform approximate geometric reconstruction of the scene. This set of photographs should have a good coverage of the scene from different viewpoints and sufficient overlap between neighboring viewpoints to facilitate multiview stereo reconstruction. The number of photographs required to obtain an acceptable reconstruction depends on the complexity of the scene and the presence of image features. We captured between 10 and 31 LDR photographs for each of the scenes presented in our results.

We then capture two high-dynamic range (HDR) images of the front and side of the reflective sphere, placed in the scene, to obtain an angular environment

[1] The reflective sphere does not need to be as accurate as in the apparatus described in [9]; in practice we used an inexpensive *pétanque* ball.



(a) Input photograph (b) Reconstructed PMVS points

(c) Proxy shadows (d) Shadow configuration

Figure 2: Multiview geometry provides us with a sparse oriented point cloud (b, red pixels). Compared to the original image (a), the initial guess of the sun shadows computed using the proxy reconstructed from the point cloud is very inaccurate due to geometric errors, and incomplete (c). In particular, a bush casting a large shadow on the floor (d) is not reconstructed.

map of the scene. We do this using the standard HDR assembly technique of Debevec et al. [10].

We finally capture linear HDR images of the viewpoints for which we want to estimate the intrinsic images. Recall that we capture all images in one session, at a single time of day.

4.2 Geometry and Illuminant Calibration

We apply Structure from motion using Bundler [37] and the patch-based multi-view stereo (PMVS) algorithm [15] on the set of LDR+HDR photographs – using the respective authors' publicly available implementations[2]. The result of this process is an oriented point cloud (3D positions and normals), and calibrated cameras (extrinsic and intrinsic parameters). The process also returns whether each point is visible from each camera. We rectify the images for radial distortion, and we consider only visible PMVS points in each image.

We found that the PMVS normals were too noisy for illumination computation. We thus estimate normals at each 3D point by fitting a plane on the

[2] Available at http://phototour.cs.washington.edu/bundler/ and http://grail.cs.washington.edu/software/pmvs/, as of 2011/10/10

local 3D point cloud using the PCA method of Hoppe et al. [17]. We discard 3D points in too sparse or too degenerate local neighborhoods. We consider a neighborhood as degenerate when the first singular value of the PCA is twice greater than the second singular value.

We recover a geometric proxy from the oriented point cloud using the scale space meshing method of Digne et al. [11]. This automatic approach produces detailed accurate meshes in regions where the point cloud is dense, while leaving holes in areas where the point cloud is irregularly sampled. We use the automatic hole filling tool available in MeshLab [6] to further improve the reconstruction in those areas. We also experimented with the Poisson reconstruction of Kazhdan et al. [21]. Although this algorithm generates a closed surface with no holes, we found that it tends to produce bumpy surfaces due to the irregular sampling of the point cloud. We nevertheless used the Poisson reconstruction on the "Rocks" scene for which the point cloud is dense and uniform.

Some manual interaction is also required to calibrate the sun and sky illumination. We first identify the sun position, orient the environment map and label sky pixels. We also take two photographs of a gray card, in the sun and in shadow, which are used to estimate the color transfer function of the reflective sphere and the radiance $\mathbf{L}_{\text{sun}}$ of the sun. The details of this process are described in the Appendix.

The output of our capture is: a moderately dense point cloud reconstruction of the scene which we will refer to as the *PMVS points* (Fig. 2b), a very approximate geometric *proxy*, which often contains significant geometric errors (Fig. 2c), the direction and radiance of the sun and a correctly aligned and scaled HDR environment map containing the sky and distant indirect radiance (Fig. 4c).

5 Geometry-Based Computation

We describe here how to compute sun, sky and indirect illumination values for each PMVS point. These points have been generated using multiview stereo and also have normals (Section 4.2).

We first compute sun illumination $\hat{\mathbf{S}}_{\text{sun}}$ ignoring cast shadows, i.e., unoccluded sunlight. We already know the required quantities for this computation, i.e., the normal at the point and the sun radiance and direction. Treating sun visibility requires much higher precision than the one provided by the proxy, and is treated separately in Section 6.

We then compute sky and indirect illumination at each PMVS point. Fig. 3 illustrates this computation that we detail bellow. In a nutshell, we use the HDR environment map to compute both sky illumination and *distant* indirect illumination, while we compute the *near-field* indirect illumination from the proxy geometry. Note however that we do not need to know the reflectance of the proxy for this step, we instead use the captured photographs to recover the necessary outgoing radiance over the reconstructed geometry.

Assigning radiance to the proxy geometry. We first assign radiance to each PMVS point by looking up its pixel values in each HDR image where it appears (Fig. 3). We assign the average value as the outgoing radiance of the point, which is assumed constant in all directions. We then assign the radiance of the closest PMVS point to each vertex of the proxy, and interpolate these

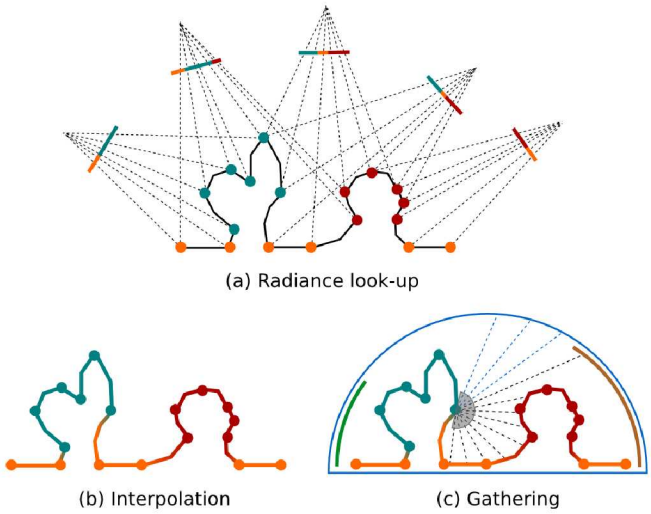


Figure 3: Evaluation of sky and indirect illumination at each PMVS point. We look up the radiance value of the PMVS points in each image and average the values over the images where they are visible (a). We then project and interpolate this radiance over the proxy geometry (b). Finally we gather the sky and indirect illumination at each PMVS point by shooting rays which sample the hemisphere (c, dotted lines). Rays that reach sky pixels in the environment map (c, blue lines) contribute to sky illumination, while other rays contribute to indirect illumination (c, dark lines).

values over the faces of the mesh. This step yields a colored proxy that stores the outgoing radiance of the scene as captured in the photographs (Fig. 4b).

Sky and indirect illumination. We separate the environment map into two regions (see Fig. 4c), one for the sky and the other one for distant objects. At every PMVS point we cast a set of rays towards the hemisphere. If a ray reaches a sky pixel of the environment map, the resulting radiance contributes to sky illumination $\mathbf{S}_{\text{sky}}$. Otherwise the ray contributes to indirect illumination $\mathbf{S}_{\text{ind}}$, either from an intersection with the proxy or from a non-sky pixel of the environment map. If the ray hits the proxy, it uses the radiance assigned to the geometry as described in the previous paragraph.

We rewrite Equations 3 and 4 to express this computation, changing the integration domain to the entire hemisphere Ω for both integrals, and introducing a visibility term $v_{\text{sky}}(\omega)$ which is 1 when the ray in direction ω reaches the sky, and 0 otherwise (it contributes to indirect illumination):

$$\mathbf{S}_{\text{sky}} = \int_{\Omega} v_{\text{sky}}(\omega) \cos\theta_{\omega} \mathbf{L}_{\text{sky}}(\omega)\ d\omega \tag{7}$$

$$\mathbf{S}_{\text{ind}} = \int_{\Omega} \big(1 - v_{\text{sky}}(\omega)\big) \cos\theta_{\omega} \mathbf{L}_{\text{ind}}(\omega)\ d\omega. \tag{8}$$

The computation of Equations 7 and 8 is robust to the coarse geometry of the proxy since the integration over Ω averages the values over the entire hemisphere. We implement this computation with a custom renderer in the PBRT stochastic raytracer [33].

(a) Input photograph

(b) Colored proxy

(c) Environment map

Figure 4: We estimate sky and indirect illumination at PMVS points using (b) a colored proxy, obtained by projecting and interpolating the radiance of PMVS points on the reconstructed geometry, and (c) an environment map, shown in a latitude-longitude parameterization where the red curve separates pixels contributing to sky illumination and those contributing to distant indirect illumination.

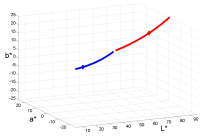
Our approach shares similarities with the techniques described by Yu and Malik [43], but while their method was designed for accurate geometry constructed manually, we handle sparse incomplete geometry reconstructed automatically.

Finally we compute an initial guess of the sun visibility $v_{\text{sun}}^{\text{init}}$ at each PMVS point by tracing a ray against the proxy geometry in the direction of the sun. Note however that this visibility test is very sensitive to errors in the reconstructed proxy, as illustrated in Fig. 2c. This fact underlines the importance of accurately estimating v_{sun} and we next show how to refine this initial estimate.

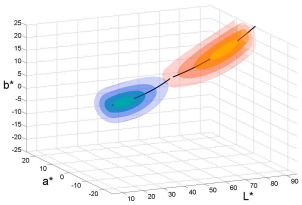
6 Estimating Sun Visibility at PMVS Points

One key contribution of our approach is a novel algorithm for identifying visibility v_{sun} with respect to the sun for each PMVS point. From our image formation model in Equations 2 and 5 we express the reflectance at each PMVS point as

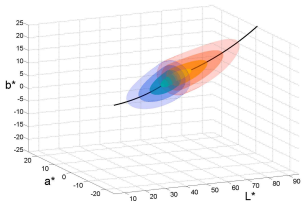
(a) Selected PMVS points



(b) Candidate reflectance curves



(c) Regions of influence at initialization



(d) Regions of influence after one iteration

Figure 5: Multiple PMVS points sharing the same reflectance will generate intersecting curves in color space. (a) We selected two PMVS points with the same reflectance but different illuminations (red and blue squares). (b) The corresponding candidate reflectance curves (nearly) intersect at one end, which corresponds to the reflectance of both points. Diamond markers on the curves correspond to the initial guess for visibility (randomly set in this example). (c) Each curve affects a region of the surrounding color space, according to Equation 10 . Regions of influence at initialization are illustrated as isosurfaces with varying opacity. Regions closer to the curve and the current visibility estimate are more affected. (d) After one iteration, the visibility estimates have moved towards the intersecting end of the curves, increasing the overlap between the regions of influence (i.e., the energy E_{total} in Equation 12).

a function of the visibility term:

$$\mathbf{R}(v_{\text{sun}}) = \frac{\mathbf{I}}{(v_{\text{sun}}\hat{\mathbf{S}}_{\text{sun}} + \mathbf{S}_{\text{sky}} + \mathbf{S}_{\text{ind}})} \tag{9}$$

where $\mathbf{I}$ is the RGB pixel value in the image we wish to process, $\mathbf{S}_{\text{sky}}$, $\mathbf{S}_{\text{ind}}$ and $\hat{\mathbf{S}}_{\text{sun}}$ are the illumination values of the corresponding PMVS point computed in Section 5, and the division is per-channel. With this parameterization, varying v_{sun} in $[0, 1]$ generates a curve of candidate reflectances in RGB space. Our goal in this section is to find for each PMVS point, the position on its *candidate curve* corresponding to its reflectance $\mathbf{R}$ (or correspondingly visibility v_{sun}).

The intuition of our approach is that multiple PMVS points sharing the same reflectance will generate intersecting curves in color space; their (shared) reflectance will be the color where the candidate curves intersect. This is illustrated on Fig. 5a-b. By finding these intersections we can deduce the value of v_{sun} for the PMVS point corresponding to each curve.

However, imprecision in the capture process and in the geometry-based computation prevent the curves from perfectly intersecting in color space. In addition, multiple intersections can occur along a curve, which gives multiple candidates for the visibility term. We address these issues with a robust iterative procedure inspired by the *mean shift* algorithm.

Overview. Mean shift [14, 7] is a non-parametric mode-seeking algorithm that aims to locate the maxima of a density function, given a set of *data points*. First, a *kernel* (or window) is placed at each data point; it represents the region

of influence of this point. In an iterative process, each kernel is then moved in a direction that increases the local density, computed as the weighted average of nearby data points. The process stops when all kernels have reached a stationary point (or mode).

In our approach, we define an asymmetric region of influence for each candidate curve. We use *mean shift iterations* to maximize an energy that measures the overlap among pairs of curves, and iteratively update the estimated reflectances while constraining them to lie on their candidate curves. After convergence, for each curve we obtain the reflectance (and corresponding visibility) that tends to maximize the number of PMVS points sharing a similar reflectance, and correspondingly tends to minimize the number of reflectances in the scene.

This algorithm assumes that the scene is composed of a sparse set of reflectances shared by multiple points, which is a common assumption recently used in image segmentation [32] and white balance algorithms [19].

Region of influence. Equation 9 defines the candidate reflectances of a PMVS point as a rational curve that is parameterized non-uniformly by v_{sun}. In order to obtain a uniform parametrization we first approximate each curve c as a piecewise linear curve in CIE L*a*b* space, and parameterize it by arc length from the shadowed end of the curve: t in $[0,1]$ so that $\mathbf{R}(v_{\text{sun}}=0) = \mathbf{R}(t=0)$. We use this uniform parametrization to compute distances along the curve. We chose to work in CIE L*a*b* space because it defines a perceptually uniform distance metric.

We then define the influence of curve c on a point $\mathbf{x}$ of color space as:

$$A_{\mathbf{x}c} = h_{\perp}\left(\frac{d_{\perp}^2(\mathbf{x},c)}{\sigma_{\perp}^2}\right) h_{\parallel}\left(\frac{d_{\parallel}^2(\mathbf{x},c)}{\sigma_{\parallel}^2}\right) \tag{10}$$

where

- $d_{\perp}^2(\mathbf{x},c) = \|\mathbf{x} - proj_{3D}(\mathbf{x},c)\|^2$ is the squared distance in color space between $\mathbf{x}$ and its projection on curve c (i.e., the distance perpendicular to the curve).

- $d_{\parallel}^2(\mathbf{x},c) = (t_c - proj_t(\mathbf{x},c))^2$ is the squared difference between the position of the current reflectance estimate along the curve t_c and the arc length $proj_t(\mathbf{x},c)$ corresponding to the projection of $\mathbf{x}$ on the curve (i.e., the distance along the curve).

- $h_{\perp}$ and $h_{\parallel}$ are Gaussian kernel profiles with the form $h(x) = e^{-x}$ controlled by the standard deviations $\sigma_{\perp}^2$ and $\sigma_{\parallel}^2$ (larger σ values correspond to a wider region of influence).

The first term in Equation 10 compensates for curves that do not exactly intersect, by defining a region of influence around the curve with a Gaussian falloff orthogonal to the curve. The second factor makes our algorithm robust to "false intersections", i.e., intersections of curves that in fact do not share the same reflectance. The Gaussian kernel $h_{\parallel}$ reduces the influence of the false intersections far from t_c as t_c converges toward the most likely reflectance.

The regions of influence of two curves are illustrated in Fig. 5c.

Energy Formulation. We then define an energy that measures the overlap between the regions of influence of pairs of curves:

$$E = \int_V \left(\sum_{c \in C} \sum_{c' \neq c} A_{\mathbf{x}c} A_{\mathbf{x}c'} \right) d\mathbf{x} \tag{11}$$

where we integrate over the entire color space V.

We evenly discretize the 3D color space into a set S of samples, and rewrite this energy as a discrete sum:

$$E_{\text{total}} = \sum_{\mathbf{s} \in S} E_{\text{sample}}(\mathbf{s}). \tag{12}$$

where the energy of a sample $E_{\text{sample}}(\mathbf{s})$ accumulates the contribution of each pair of curves intersecting nearby:

$$E_{\text{sample}}(\mathbf{s}) = \sum_{c \in C} \sum_{c' \neq c} A_{\mathbf{s}c} A_{\mathbf{s}c'}$$

With this formulation, two curves will contribute to the energy of a sample only if they (almost) intersect near this sample.

Derivation of the energy gradient. We seek the positions of the reflectance estimates along the curves t_c for $c \in C$ that maximize E_{total}:

$$\underset{t_c}{\text{argmax}}\, E_{\text{total}} \tag{13}$$

i.e. that are located at the zeros of the gradient function.

The derivative of E_{total} with respect to t_c is given by:

$$\frac{\partial E_{\text{total}}}{\partial t_c} = \frac{4}{\sigma_\parallel^2} \sum_{\mathbf{s} \in S} \sum_{c' \neq c} \left(proj_t(\mathbf{s}, c) - t_c \right) A_{\mathbf{s}c} A_{\mathbf{s}c'} \tag{14}$$

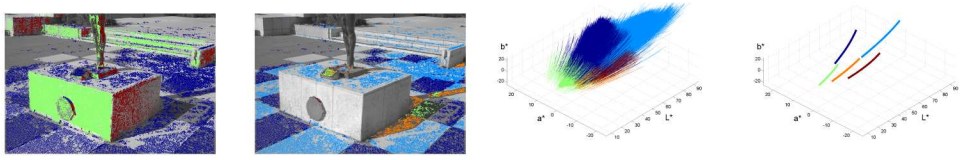
Setting Equation 14 to 0 for all $c \in C$ gives:

$$\begin{aligned} &\frac{\partial E_{\text{total}}}{\partial v_c} = 0 \quad \Leftrightarrow \\ &\left(\sum_{\mathbf{s} \in S} \sum_{c \neq c'} A_{\mathbf{s}c} A_{\mathbf{s}c'} \right) \\ &\left(t_c - \frac{\sum_{\mathbf{s} \in S} proj_t(\mathbf{s}, c) \left(\sum_{c' \neq c} A_{\mathbf{s}c} A_{\mathbf{s}c'} \right)}{\sum_{\mathbf{s} \in S} \sum_{c' \neq c} A_{\mathbf{s}c} A_{\mathbf{s}c'}} \right) = 0 \end{aligned}$$

which is analogous to the form obtained in the mean-shift algorithm [7].

Iterative process. We define our iterative procedure recursively by computing at iteration i the weights $A^i_{\mathbf{s}c}$ (i.e., the regions of influence) using the estimates t^i_c for all curves $c \in C$, then updating the estimates t^{i+1}_c for each curve c using a weighted average of the projection of nearby samples on c:

$$t_c^{i+1} = \frac{\sum_{\mathbf{s} \in S} proj_t(\mathbf{s}, c) \left(\sum_{c' \neq c} A^i_{\mathbf{s}c} A^i_{\mathbf{s}c'} \right)}{\sum_{\mathbf{s} \in S} \sum_{c' \neq c} A^i_{\mathbf{s}c} A^i_{\mathbf{s}c'}} \tag{15}$$



(a) Principal orientations (b) Clusters for vertical orientation (c) Candidate curves before clustering (d) Candidate curves after clustering

Figure 6: Orientation separation and curve clustering. (a) PMVS points are separated into three groups according to the orientation of their normals. Each group is then processed independently. (b-c) PMVS points are clustered according to the endpoints of their candidate curves. (d) After clustering, pairs of curves intersect as expected (orange and light blue, green and dark blue), since each pair represents PMVS points sharing the same reflectance. Reflectances that are only observed in shadow or sun light form a single curve (brown) that inherits the sun visibility computed from the proxy.

We initialize t_c^0 using the initial guess of the sun visibility $v_{\text{sun}}^{\text{init}}$, obtained in Section 5 by casting the shadow of the geometric proxy. We ensure that this iterative process converges to a maximizer by checking that the energy does not decrease after each iteration [8]; if $E_{\text{total}}^{i+1} < E_{\text{total}}^{i}$ we set $t_c^{i+1} = (t_c^{i+1} + t_c^i)/2$ for all c and iterate. The algorithm stops when $|t_c^{i+1} - t_c^i|$ is small enough for all c.

As a result of this process, the reflectance estimates converge to regions of space where many curves intersect, while being constrained to lie on their candidate curves due to our parameterization. This is illustrated in Fig. 5d.

Orientation separation. We observed that false intersections sometimes occur when two curves of different reflectances have different orientations, sky or indirect illumination. In order to further reduce the presence of such intersections, we separate PMVS points into three groups based on the orientation of their normals. We apply mean-shift clustering [7] where the 3D feature vectors contain the normals of PMVS points, and group the points into three classes: two corresponding to the modes with the two highest counts, and one that contains all the remaining points (Fig. 6a). We then run our optimization independently on each group.

Curve clustering. Even within one group of orientations, the candidate curves corresponding to the numerous PMVS points (up to dozens of thousands, in our scenes) tend to intersect each other at several positions along the curves (Fig. 6c), since many similar (but not equal) reflectances might be present in the scene. We enforce sparsity in our algorithm by grouping all the curves corresponding to a similar reflectance and illumination (Fig. 6b). Note that this grouping is only used to identify the sun visibility, we do not enforce sparsity in the final reflectance image.

We use mean-shift clustering to perform this grouping, representing each curve as a 6D feature vector that contains the L*a*b* reflectances of its endpoints. Clusters that contain few curves (less than 1% of the largest cluster) are discarded, and the corresponding PMVS points are ignored in the remaining of the algorithm. This leads to the clustering shown in Fig. 6b-c.

Finally, we replace all the curves belonging to each cluster by one *representative curve*: the curve closest to the median of this cluster's feature vectors. The initial guess of the sun visibility $v_{\text{sun}}^{\text{init}}$ is assigned to the median visibility of each cluster. This process greatly simplifies and cleans up the set of curves, and lowers the computational cost of our algorithm. As shown in Fig. 6d, the curves corresponding to groups of PMVS points with similar reflectances (nearly) intersect after clustering.

After clustering, we run the optimization to maximize E_{total}. Upon convergence, for each cluster we obtain the position of the estimated reflectance along its representative curve t^{final}. This position value is assigned to all curves belonging to this cluster, from which we can deduce the final estimated reflectance of each PMVS point corresponding to these curves.

At the end of this process, we obtain a list of PMVS points for which the position along the curve t, the sun visibility v_{sun} and the reflectance $\mathbf{R}$ have been estimated.

Implementation. In practice we use a truncated kernel profile $h_\perp$ (Equation 10) so that $h_\perp(x) = 0$ when $x > \lambda$ (we use $\lambda = 3$). This means that each curve will only influence a limited number of samples; for each curve c we can precompute the indices of these samples, as well as their orthogonal distance $\|\mathbf{s} - proj_{3D}(\mathbf{s}, c)\|$ and position of their projection along the curve $proj_t(\mathbf{s}, c)$.

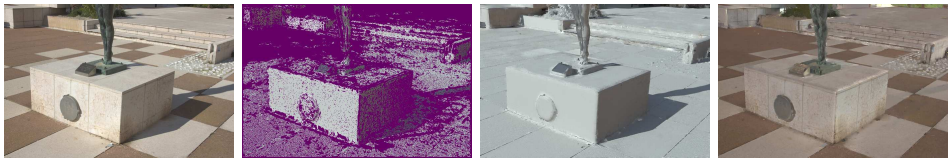
For all our results, we evenly discretized the CIE L*a*b* color space into $60 \times 36 \times 36$ samples, with L* in $[5, 95]$, a* in $[-25, 25]$, b* in $[-25, 25]$. We used a fixed bandwidth for the curve clustering using 6D mean-shift clustering, as well as for the 3D mean-shift clustering for orientation separation.

The clustering is the most costly part of the algorithm and takes from 25 seconds to a few minutes with our Matlab implementation, depending on the number of PMVS points. Once the clustering has been performed, the iterative optimization takes around 10 seconds, which allowed us to test many parameters for $\sigma_\perp^2$ and $\sigma_\parallel^2$.

We found that the algorithm produces good results for a wide range of parameters, and that the best values were scene-dependent. In our experiments we found that large values of $\sigma_\perp^2$ can compensate for calibration errors that prevent reflectance curves to perfectly intersect, while large values of $\sigma_\parallel^2$ can compensate for the erroneous initialization of $v_{\text{sun}}^{\text{init}}$ provided by the approximate proxy. However, σ values should remain small enough to prevent curves of different materials to influence one another. Table 1 summarizes the parameter values used for the examples in this paper.

	Statue	Rocks	Ramp	Stairs
$(\sigma_\parallel^2, \sigma_\perp^2)$	$(0.1, 1)$	$(0.01, 1)$	$(0.3, 1)$	$(0.01, 20)$
	$(0.1, 5)$	$(0.01, 5)$	$(0.3, 5)$	

Table 1: Sets of region of influence parameters used on the four scenes. When specified, the two sets of parameters produced equally good results.



(a) Input photograph (b) Illumination constraints (c) Estimated total illumination (d) Estimated reflectance

Figure 7: Separation of Reflectance and Total Illumination. The optimization procedure described in Equation 17 enforces illumination constraints at PMVS points (b) and propagates them to all pixels, in order to separate an input photograph (a) into total illumination (c) and reflectance (d).

7 Estimating Illumination at Each Pixel

In previous steps, we have used multiview stereo methods to generate a sparse set of 3D points on which we compute the illumination values $\hat{\mathbf{S}}_{\text{sun}}$, $\mathbf{S}_{\text{sky}}$ and $\mathbf{S}_{\text{ind}}$ (Section 5) along with the visibility of the sun v_{sun} (Section 6). We next show how to leverage image-guided propagation methods to assign reflectance and illumination values to all pixels in the input photographs. We first show how to propagate the total illumination $\mathbf{S}_{\text{total}}$ and then describe a method to subsequently separate the contribution of each lighting component (i.e., sun, sky and indirect).

7.1 Image Guided Propagation

We use the intrinsic images algorithm of Bousseau et al. [2] that was designed to propagate user indications for separating reflectance and illumination in a single image. This algorithm makes the intrinsic image decomposition tractable by assuming that the reflectance values in a pixel neighborhood lie in a plane in color space. This planar reflectance assumption translates to a set of linear equations so that the illumination image is expressed as the minimizer of a least-square energy

$$\underset{\bar{\mathbf{S}}_{\text{total}}}{\operatorname{argmin}}\, \bar{\mathbf{S}}_{\text{total}}^T \mathbf{M} \bar{\mathbf{S}}_{\text{total}} \tag{16}$$

where the vector $\bar{\mathbf{S}}_{\text{total}}$ stacks the pixels of the estimated illumination image and the matrix $\mathbf{M}$ encodes the planar reflectance assumption (see the paper by Bousseau et al. [2] for the complete derivation). For colored illumination the optimization is solved for each color channel separately.

In their original paper, Bousseau et al. constrain the least-square system with user indications. Users can specify the value of $\bar{\mathbf{S}}_{\text{total}}$ over a few pixels or indicate that several pixels share the same illumination or reflectance. In our approach we use instead the illumination and visibility estimated at PMVS points to constrain the optimization. We express these constraints as an additional quadratic penalty

$$\underset{\bar{\mathbf{S}}_{\text{total}}}{\operatorname{argmin}}\, \bar{\mathbf{S}}_{\text{total}}^T \mathbf{M} \bar{\mathbf{S}}_{\text{total}} + w \sum_{p \in \mathcal{P}} (\bar{\mathbf{S}}_{\text{total}}^p - \mathbf{S}_{\text{total}}^p)^2 \tag{17}$$

where $\mathcal{P}$ is the set of pixels covered by PMVS points and $\mathbf{S}^p_{\text{total}} = v^p_{\text{sun}}\hat{\mathbf{S}}^p_{\text{sun}} + \mathbf{S}^p_{\text{sky}} + \mathbf{S}^p_{\text{ind}}$ their illumination values computed in Sections 5 and 6. The weight w controls the importance of the constraints, in practice we use $w = 1$ to give equal importance to the constraints and the propagation model.

Fig. 7 shows the results of the image guided propagation. This result shows the power of our approach. We exploit the information provided by the sparse and imprecise geometry to provide constraints *automatically* to the algorithm of Bousseau et al. [2], thus eliminating the need for user scribbles. A visualization of the constraints for all scenes can be found in the supplemental material.

7.2 Light Source Separation

Given the estimated illumination image $\bar{\mathbf{S}}_{\text{total}}$, we wish to separate the contribution of each illumination component $\bar{\mathbf{S}}_{\text{sun}}$, $\bar{\mathbf{S}}_{\text{sky}}$ and $\bar{\mathbf{S}}_{\text{ind}}$. Inspired by previous work on white balance under mixed lighting [19], we show how to express our light source separation as two successive matting problems. In a first step we decompose the illumination into a sun component and a *diffuse* component $\bar{\mathbf{S}}_{\text{diff}}$ that includes the contribution of both sky and indirect lighting. In a second step we decompose the diffuse component into its two terms.

We first express each illumination term $\bar{\mathbf{S}}$ as the product between a scalar intensity $\mathbf{s} = \|\bar{\mathbf{S}}\|$ and a chromaticity $\mathbf{C} = \bar{\mathbf{S}}/\|\bar{\mathbf{S}}\|$:

$$\begin{aligned}\bar{\mathbf{S}}_{\text{total}} &= \mathbf{s}_{\text{sun}}\mathbf{C}_{\text{sun}} + \mathbf{s}_{\text{sky}}\mathbf{C}_{\text{sky}} + \mathbf{s}_{\text{ind}}\mathbf{C}_{\text{ind}} \\ &= \mathbf{s}_{\text{sun}}\mathbf{C}_{\text{sun}} + \mathbf{s}_{\text{diff}}\mathbf{C}_{\text{diff}}. \end{aligned} \tag{18}$$

Denoting $\alpha = \mathbf{s}_{\text{sun}}/(\mathbf{s}_{\text{sun}} + \mathbf{s}_{\text{diff}})$ we express the illumination image values at each pixel as a mixture between two values weighted by α:

$$\begin{aligned}\bar{\mathbf{S}}_{\text{total}} &= \alpha(\mathbf{s}_{\text{sun}} + \mathbf{s}_{\text{diff}})\mathbf{C}_{\text{sun}} \\ &\quad +(1-\alpha)(\mathbf{s}_{\text{sun}} + \mathbf{s}_{\text{diff}})\mathbf{C}_{\text{diff}} \qquad (19) \\ &= \alpha\mathbf{F} + (1-\alpha)\mathbf{B} \qquad (20)\end{aligned}$$

We can now recover $\bar{\mathbf{S}}_{\text{sun}} = \alpha\mathbf{F}$ and $\bar{\mathbf{S}}_{\text{diff}} = (1-\alpha)\mathbf{B}$ by solving a standard matting problem. We compute α at each PMVS point from the illumination values estimated in Sections 5 and 6. We then propagate α over the image $\bar{\mathbf{S}}_{\text{total}}$ using the *matting Laplacian* algorithm of Levin et al. [24]. Finally, given α at every pixel and the known $\mathbf{S}_{\text{sun}}$ and $\mathbf{S}_{\text{diff}}$ at each PMVS point, we solve for the sun and diffuse illumination images with the following least-square optimization:

$$\begin{aligned}\underset{\mathbf{F},\mathbf{B}}{\operatorname{argmin}} \sum_{i\in\mathcal{I}} \Bigg(\Big(\bar{\mathbf{S}}^i_{\text{total}} - \big(\alpha^i\mathbf{F}^i + (1-\alpha^i)\mathbf{B}^i\big)\Big)^2 \\ + \lambda\Big((\mathbf{F}^i_x)^2 + (\mathbf{F}^i_y)^2 + (\mathbf{B}^i_x)^2 + (\mathbf{B}^i_y)^2\Big)\Bigg) \\ + w\sum_{p\in\mathcal{P}} \big(\alpha^p\mathbf{F}^p - \mathbf{S}^p_{\text{sun}}\big)^2 + \big((1-\alpha^p)\mathbf{B}^p - \mathbf{S}^p_{\text{diff}}\big)^2\end{aligned}$$

where $\mathcal{I}$ is the image domain, $\mathcal{P}$ is the set of pixels covered by PMVS points and $\mathbf{F}_x$, $\mathbf{F}_y$, $\mathbf{B}_x$ and $\mathbf{B}_y$ are the x and y derivatives of $\mathbf{F}$ and $\mathbf{B}$ computed with finite differences.

(a) Before inpainting (b) After inpainting

Figure 8: The reflectance, sky illumination (here, close up) and indirect illumination estimated by our algorithm can contain residual variations along hard shadow boundaries (a). We use inpainting to remove these artifacts (b).

The first term of this functional ensures that the decomposition explains the input illumination $\bar{\mathbf{S}}_{\text{total}}$ and follows the estimated ratio α. The second term adds a smoothness regularization on each component while the third term constrains the solution to agree with the illumination values computed at PMVS points. We used $\lambda = 0.1$ and $w = 0.01$ for all our results.

As a second step we apply the same matting approach to further separate the diffuse illumination $\bar{\mathbf{S}}_{\text{diff}}$ as the sum of the sky illumination $\bar{\mathbf{S}}_{\text{sky}}$ and the indirect illumination $\bar{\mathbf{S}}_{\text{ind}}$. We show in Fig. 9 and supplemental materials the results of this decomposition. The overall decomposition takes around 90 seconds to compute for a 3.2 megapixel image, where 30 seconds are necessary to compute $\bar{\mathbf{S}}_{\text{total}}$ and 60 seconds to perform the two subsequent separations.

Inpainting This overall process gives a satisfactory decomposition in the scenes we have tested. However, a small residual border can remain around the hard shadow boundary (see Fig. 8 (left)), a common artifact of shadow removal [12, 41]. We identify these shadow boundaries as pixels located on sun illumination discontinuities but not on normal discontinuities. We first propagate normals from the PMVS points over the image using the method of Okabe et al. [31]. We then run an edge detector over the sun illumination image and label edge pixels that do not correspond to edges in the normal image. We remove the labeled pixels and their immediate neighbors from the reflectance, sky and indirect illumination images and use inpainting to fill in the holes. This post-process takes 60 seconds on average. Fig. 8 illustrates the result of this process.

8 Results and Discussion

In Fig. 9 we show results on four different scenes. For all results, we show reflectance, sun, sky and indirect illumination layers. The number of photographs used for each scene is shown in Table 2. The estimated illumination at PMVS points and additional views for each scene are shown in the supplemental material.

The first scene shows the base of a statue on a square; we took HDR images

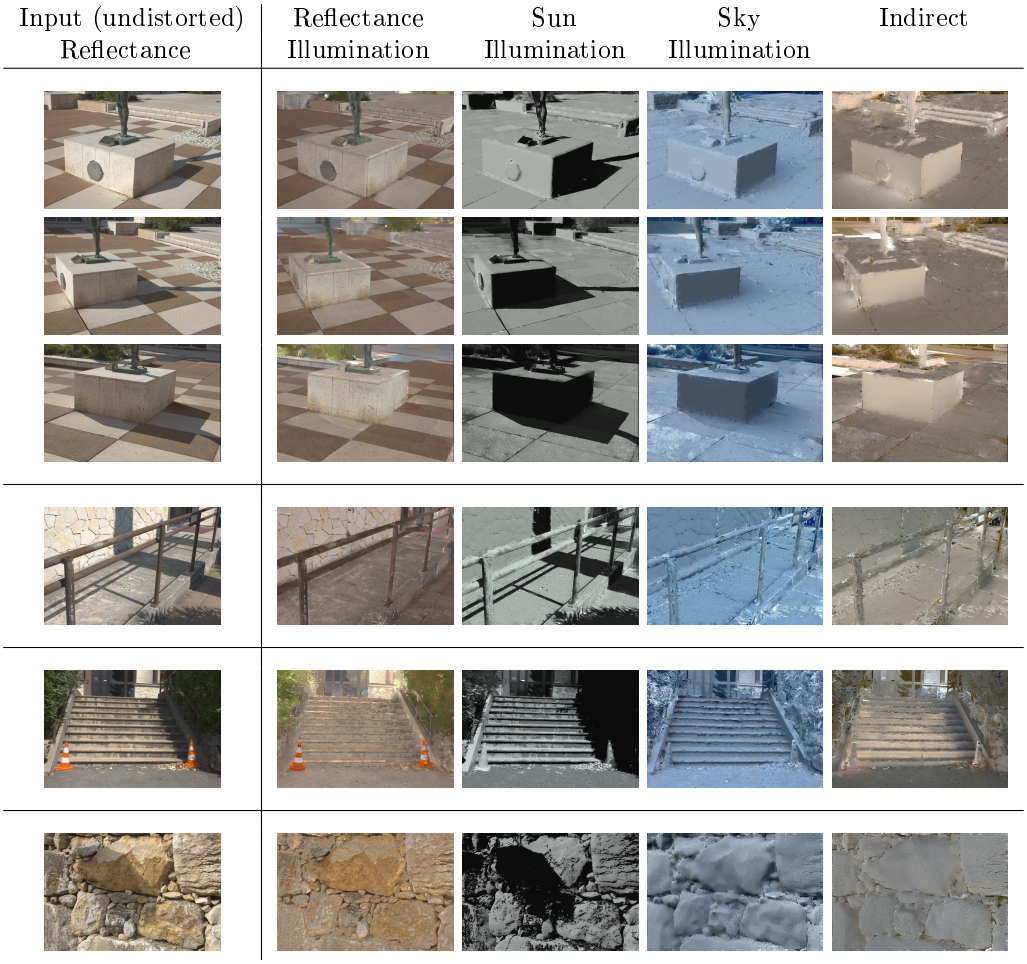


Figure 9: Results of our decomposition. We adjusted the brightness of images for illustration purpose (the scaling factors used can be found in the supplementary materials). For each scene, the sun illumination is usually much more intense than sky illumination on average, and the sky illumination is more intense than indirect illumination on average.

for three different views and we show results for all three. As we can see the results are plausible for the reflectance layer in all views. The sun and sky layers have been successfully separated in all cases, however the third view shows a slight color shift of the reflectance in the shadow area; this is due to the insufficient number of PMVS points on the ground. Please refer to the supplemental material to see an illustration of the distribution of 3D points. The indirect layer is particularly interesting, since it clearly shows the indirect light bouncing off the front of the base (first and second views) which is in direct sunlight.

The second scene is challenging, since the reconstruction process is unable to capture details of the railings and does not reconstruct the vegetation casting the main shadows (see also Fig. 2d). In this view, despite the intricate geometric

configuration, shadows are successfully removed from the reflectance layer, and the three other layers show good results. In the third scene, a staircase is shown. There are some residual artifacts at the shadow boundaries because the vegetation moved in the breeze during HDR acquisition. In the fourth scene, a rock wall is shown. Notice how the indirect layer well represents lighting in the cracks between rocks where neither sun nor skylight is present. The third and fourth scene also show that our method succeeds in creating plausible reflectance, sun, sky and indirect layers.

Statue	Rocks	Ramp	Stairs
30	11	31	10

Table 2: Number of photographs captured for the geometric reconstruction, for each scene in this paper.

Next we compare our approach to three state-of-the-art methods in Fig. 10. All these methods take a single image as input. The user-assisted approach of Bousseau et al. [2] produces result of a similar quality to ours, but requires a significant number of user indications (between 25 and 105 scribbles). The automatic method of Shen et al. [35] is able to extract most of the illumination variations but colored shadows remain in the reflectance image. These colored residuals are due to the variation in color between sun and sky illumination, which violates the gray illumination assumption of this method. Residual shadows are also present in the reflectance estimated with the automatic method of Shen et al. [34], as well as reflectance residuals in the illumination image (tiled floor in the statue scene). Although this method can support user scribbles, the authors reported that scribbles did not improve the result significantly in these examples. All the results of these comparisons have been kindly provided by the respective authors.

We also provide in Fig. 11 a comparison to the result obtained simply by using the proxy to compute sun, sky and indirect illumination using PBRT, and then inverting Equation 5 to obtain reflectance. As we can see, for a given image, there are many holes due to the incomplete proxy. In addition, the sun illumination is completely erroneous, due to the lack of reconstruction of the surrounding objects. In contrast, our method correctly captures these sun shadows, and removes them in the reflectance layer as well.

We illustrate applications of our decomposition in Fig. 1. We first alter the reflectance of the ground to insert a graffiti while maintaining consistent shadows (b). We then add a virtual object in the scene with consistent lighting and shadows (c). We used PBRT to render the dinosaur surrounded by the captured environment map, and its shadow cast on a horizontal plane. Finally we simulate a sunset by changing the color and intensity of each illumination component separately; in addition, our decomposition allows us to blur shadows without affecting the reflectance of the scene (d). All these manipulations can be performed easily in image-editing software with layer support; the supplementary video shows how we created the images in Fig. 1b-d with Adobe Photoshop.

Discussion A drawback of our current method is the need for the reflective sphere to capture the environment map. We could investigate fitting a sky model to sky pixels visible in the other input photographs, similar to [43], and extracting information about non-sky surroundings in a similar manner. We

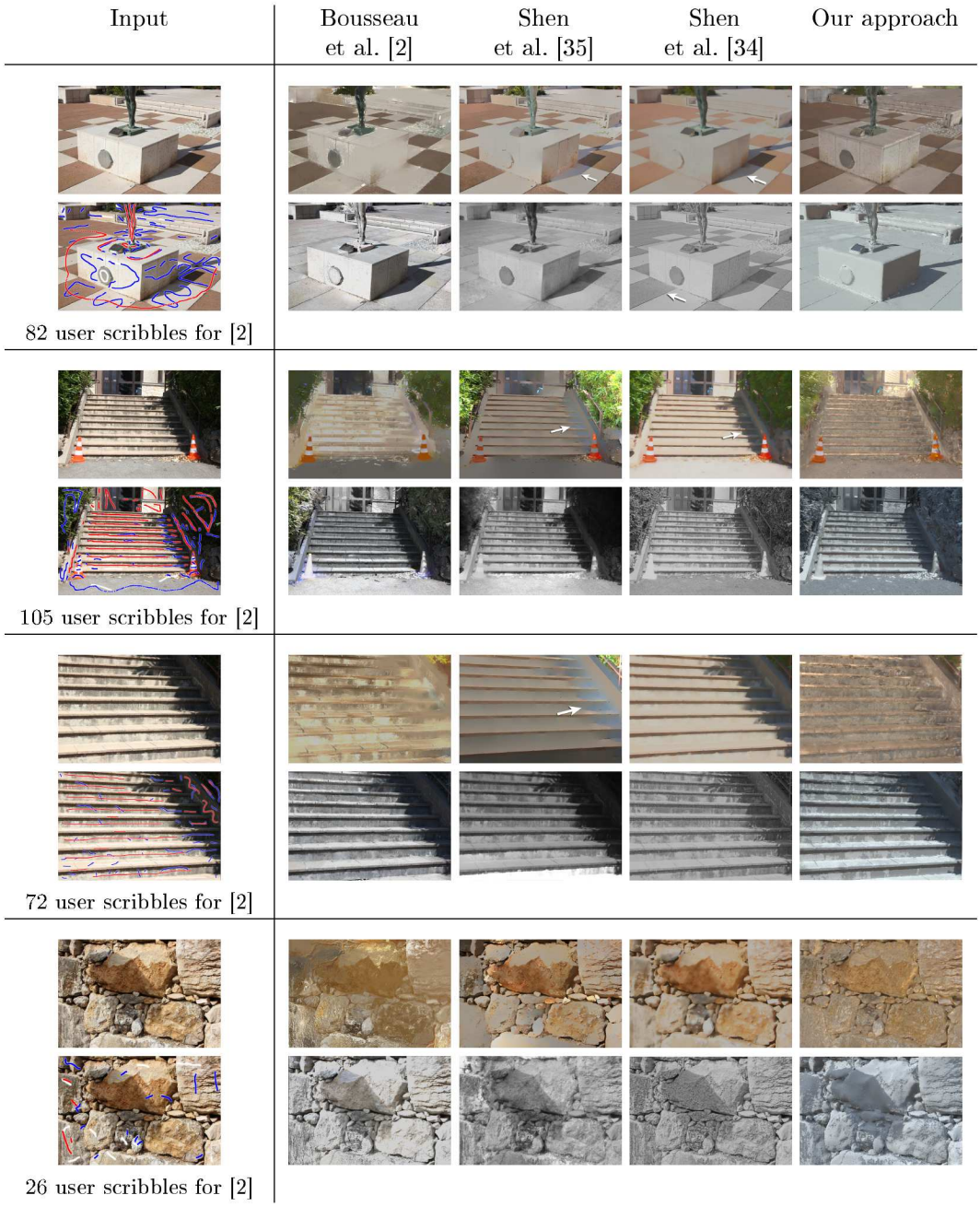


Figure 10: Comparison of our method with a user-assisted method [2] and two automatic algorithms [35, 34]. In the first column, the user scribbles used for the method of Bousseau et al. [2] are shown under the input image. In the next columns, the first row contains the estimated reflectance while the second one corresponds to the total illumination. Our results are shown in the last column. Our multiview approach outperforms single-image automatic algorithms and achieves results of comparable quality to the user-assisted approach with significantly less user intervention. White arrows point to residual reflectance or erroneous shading variations. The brightness of images has been adjusted for comparison.

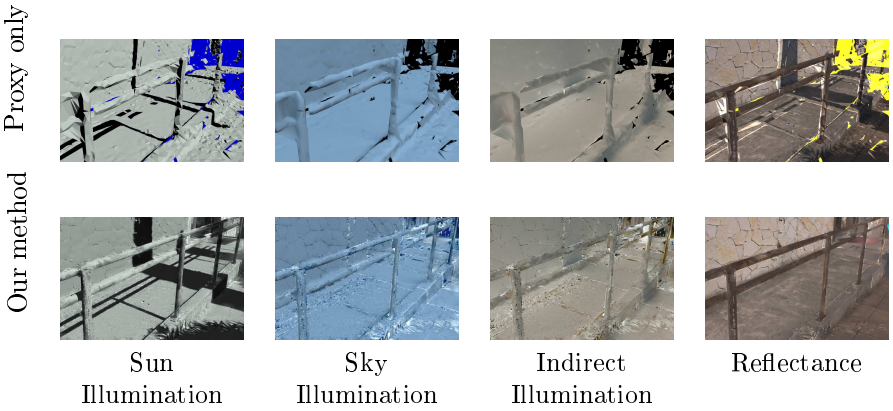


Figure 11: Comparison between the decomposition estimated directly from the geometric proxy (first row), and our results (second row). We obtain the proxy reflectance by dividing the input image by the sum of the illumination components. Holes and inaccuracies in the proxy translate to artifacts and residual shadows in the reflectance.

will investigate ways to simplify the calibration process of the sphere so that it can be done once and subsequently used for any scene.

Since we were interested in separating sun lighting from other sources, we have not shown overcast scenes. There is no fundamental reason that our method would not work with overcast scenes; the decomposition would simply rely on the $\mathbf{S}_{\text{sky}}$ and $\mathbf{S}_{\text{ind}}$ values and ignore the sun illumination component.

Our method is robust to holes in the proxy, since in all visibility calculations rays that do not hit the proxy will hit the environment map and get a plausible color. However, our method can fail if the initial guess for sun visibility is completely wrong. In the supplemental material we show a case where a spurious object appeared in a single view, and was thus not reconstructed at all, resulting in this type of failure. Similarly, if objects have very dark reflectance the PMVS reconstruction procedure does not provide a sufficient number of points, resulting in errors. Our optimization for sun visibility exploits the redundant information provided by points of the same reflectance that are in sun light *and* shadow. Nevertheless our method can also handle reflectances that are only in light *or* shadow. Most often in such case the proxy initialization will result in the correct answer. The only case which could potentially cause errors is when other reflectance curves incorrectly intersect with or influence the curve of this material. Finally, the candidate curves in our sun visibility method may not intersect if the illuminant calibration is too imprecise. We are investigating ways to improve calibration as well as a way to automate the process.

9 Conclusion

We have presented a method to estimate rich intrinsic images for outdoor scenes. In addition to reflectance, our algorithm generates a separate image for the sun, sky and indirect illumination. Our method relies on a lightweight capture (10-31 photographs in the scenes shown here) to estimate a coarse geometric

representation of the scene. This geometric information allows us to estimate illumination terms over a sparse 3D sampling of the scene. We then introduce an optimization algorithm to refine the inaccurate estimations of sun visibility. While incomplete, we demonstrate that this sparse information provides the necessary constraints for an image-guided propagation algorithm that recovers the reflectance and illumination components at each pixel of the input photographs.

Our intrinsic image decomposition allows users of image manipulation tools to perform consistent editing of material and lighting in photographs. An interesting direction of future work is to adapt the method of [5] to provide an alternative way of computing the illumination components. It will be interesting to investigate the tradeoffs between complexity, speed and quality obtained from such an adaptation compared to our approach.

Our current method works independently for each view. Enforcing consistent intrinsic image properties between views is an interesting future research direction. With such consistency, our method will open the way for dynamically relightable environments and free-viewpoint navigation for Image Based Rendering systems [3], in addition to the applications demonstrated in this paper. An important step for this goal is the generation of plausible shadow motion in the sun illumination layer in order to simulate moving light sources.

References

[1] H.G. Barrow and J.M. Tenenbaum. Recovering intrinsic scene characteristics from images. *Computer Vision Systems*, 1978.

[2] Adrien Bousseau, Sylvain Paris, and Frédo Durand. User-assisted intrinsic images. *ACM Trans. Graph.*, 28(5):1–10, 2009.

[3] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. *SIGGRAPH '01*, pages 425–432, 2001.

[4] Marcio Cabral, Nicolas Bonneel, Sylvain Lefebvre, and George Drettakis. Relighting photographs of tree canopies. *IEEE Transactions on Visualization and Computer Graphics*, 17:1459–1474, 2011.

[5] Robert Carroll, Ravi Ramamoorthi, and Maneesh Agrawala. Illumination decomposition for material recoloring with consistent interreflections. *ACM TOG (Proc. of SIGGRAPH 2011)*, August 2011.

[6] Paolo Cignoni, Massimiliano Corsini, and Guido Ranzuglia. Meshlab: an open-source 3d mesh processing system. *ERCIM News*, (73):45–46, 2008.

[7] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5):603 –619, may 2002.

[8] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 2, pages 142 –149 vol.2, 2000.

[9] Paul Debevec, Chris Tchou, Andrew Gardner, Tim Hawkins, Charis Poullis, Jessi Stumpfel, Andrew Jones, Nathaniel Yun, Per Einarsson, Therese Lundgren, Marcos Fajardo, and Philippe Martinez. Estimating surface reflectance properties of a complex scene under captured natural illumination. Technical report, USC Institute for Creative Technologies, 2004.

[10] Paul E. Debevec and Jitendra Malik. Recovering high dynamic range radiance maps from photographs. In *SIGGRAPH'97*, pages 369–378, 1997.

[11] Julie Digne, Jean-Michel Morel, Charyar-Mehdi Souzani, and Claire Lartigue. Scale space meshing of raw data point sets. *Computer Graphics Forum*, 6:1630–1642, 2011.

[12] Graham D. Finlayson, Mark S. Drew, and Cheng Lu. Intrinsic images by entropy minimization. In *ECCV*, pages 582–595, 2004.

[13] Graham D. Finlayson, Steven D. Hordley, and Mark S. Drew. Removing shadows from images. In *ECCV*, 2002.

[14] K Fukunaga and L Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21(1):32–40, 1975.

[15] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Trans. PAMI*, 32(8):1362–1376, 2009.

[16] Roger Grosse, Micah K. Johnson, Edward H. Adelson, and William T. Freeman. Ground-truth dataset and baseline evaluations for intrinsic image algorithms. In *International Conference on Computer Vision*, pages 2335–2342, 2009.

[17] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. *SIGGRAPH*, 26:71–78, 1992.

[18] Berthold K. Horn. *Robot Vision*. McGraw-Hill Higher Education, 1st edition, 1986.

[19] Eugene Hsu, Tom Mertens, Sylvain Paris, Shai Avidan, and Frédo Durand. Light mixture estimation for spatially varying white balance. *ACM TOG (proc. of SIGGRAPH)*, 27(3):70, 2008.

[20] James T. Kajiya. The rendering equation. *SIGGRAPH*, 20:143–150, August 1986.

[21] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proc. of Eurographics Symposium on Geometry Processing*, pages 61–70, 2006.

[22] Edwin H. Land and John J. McCann. Lightness and retinex theory. *Journal of the optical society of America*, 61(1), 1971.

[23] Anat Levin, Dani Lischinski, and Yair Weiss. Colorization using optimization. *ACM TOG (proc. of SIGGRAPH 2004)*, 23:689 – 694, 2004.

[24] Anat Levin, Dani Lischinski, and Yair Weiss. A closed-form solution to natural image matting. *IEEE Trans. PAMI*, 2008.

[25] Shen Li and Chuohao Yeo. Intrinsic image decomposition using a local and global sparse representation of reflectance. *CVPR*, 2011.

[26] Dani Lischinski, Zeev Farbman, Matt Uyttendaele, and Richard Szeliski. Interactive local adjustment of tonal values. *ACM TOG (proc. of SIGGRAPH)*, 25:646–653, 2006.

[27] Céline Loscos, George Drettakis, and Luc Robert. Interactive virtual relighting of real scenes. *IEEE Transactions on Visualization and Computer Graphics*, 6:289–305, 2000.

[28] Wojciech Matusik, Matthew Loper, and Hanspeter Pfister. Progressively-refined reflectance functions from natural illumination. In *Eurographics Symposium on Rendering*, pages 299–308, 2004.

[29] F. Melendez, M. Glencross, G. J. Ward, and R. J. Hubbold. Relightable Buildings from Images. In *Eurographics: Special Area on Cultural Heritage*, pages 33–40, 2011.

[30] Ankit Mohan, Jack Tumblin, and Prasun Choudhury. Editing soft shadows in a digital photograph. *IEEE Computer Graphics and Applications*, 27(2):23–31, 2007.

[31] Makoto Okabe, Gang Zeng, Yasuyuki Matsushita, Takeo Igarashi, Long Quan, and Heung yeung Shum. Single-view relighting with normal map painting. In *Proc. Pacific Graphics 2006*, pages 27–34, 2006.

[32] Ido Omer and Michael Werman. Color lines: Image specific color representation. *CVPR*, 2:946–953, 2004.

[33] Matt Pharr and Greg Humphreys. *Physically Based Rendering: From Theory to Implementation, second edition.* Morgan Kaufmann Publishers Inc., 2010.

[34] Jianbing Shen, Xiaoshan Yang, Yunde Jia, and Xuelong Li. Intrinsic images using optimization. *CVPR*, 2011.

[35] Li Shen, Ping Tan, and Stephen Lin. Intrinsic image decomposition with non-local texture cues. In *CVPR*, 2008.

[36] Yael Shor and Dani Lischinski. The shadow meets the mask: Pyramid-based shadow removal. *Comput. Graph. Forum*, 27(2):577–586, 2008.

[37] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3D. *ACM Trans. Graph.*, 25(3):835–846, 2006.

[38] Kalyan Sunkavalli, Wojciech Matusik, Hanspeter Pfister, and Szymon Rusinkiewicz. Factored time-lapse video. *ACM Trans. Graph.*, 26(3):101, 2007.

[39] Marshall F. Tappen, William T. Freeman, and Edward H. Adelson. Recovering intrinsic images from a single image. *IEEE Trans. PAMI*, 27(9), 2005.

[40] Yair Weiss. Deriving intrinsic images from image sequences. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, volume 2, page 68, 2001.

[41] Tai-Pang Wu, Chi-Keung Tang, Michael S. Brown, and Heung-Yeung Shum. Natural shadow matting. *ACM TOG*, 26(2):8, 2007.

[42] Yizhou Yu, Paul Debevec, Jitendra Malik, and Tim Hawkins. Inverse global illumination: recovering reflectance models of real scenes from photographs. In *SIGGRAPH '99*, pages 215–224, 1999.

[43] Yizhou Yu and Jitendra Malik. Recovering photometric properties of architectural scenes from photographs. In *SIGGRAPH '98*, pages 207–217, 1998.

Appendix: Illuminant Calibration

In this appendix, we describe the details of the illuminant calibration step for the sky and the sun.

First, because our model separates sun light from sky light, we need to remove sun pixels from the environment map. We use inpainting to fill-in the saturated sun pixels from their neighbors and define the sun position as the barycenter of the inpainted pixels. Since our model also separates sky light from indirect light, we use a standard color selection tool to label sky pixels that will contribute to the sky illumination, while other pixels (building, trees) will contribute to indirect lighting.

Second, the environment map only captures a scaled version of the scene lighting since the sphere is not perfectly specular. We need to compensate for this scaling factor.

In our system the environment map is used to compute both the sky illumination $\mathbf{S}_{\text{sky}}$ and part of the indirect illumination $\mathbf{S}_{\text{ind}}$, the other part being computed from the geometric proxy (see Section 5 for more details). We estimate the color transfer function of the reflective sphere $\mathbf{K}$ (represented as a RGB vector) by taking a photograph of a neutral gray card with known reflectance placed in sun shadow. We intentionally place the card at a position where we expect its geometry to be well reconstructed. From the image formation model we have

$$\begin{aligned} \mathbf{I} &= \mathbf{R} * (\mathbf{S}_{\text{sky}} + \mathbf{S}_{\text{ind}}) \\ &= \mathbf{R} * (\mathbf{K} * \mathbf{S}^{\text{env}}_{\text{sky}} + \mathbf{K} * \mathbf{S}^{\text{env}}_{\text{ind}} + \mathbf{S}^{\text{proxy}}_{\text{ind}}) \end{aligned} \tag{21}$$

where $\mathbf{S}^{\text{env}}$ denotes the illumination terms computed from the environment map and $\mathbf{S}^{\text{proxy}}$ the ones computed from the geometric proxy. We use this equation to solve for the unknown $\mathbf{K}$.

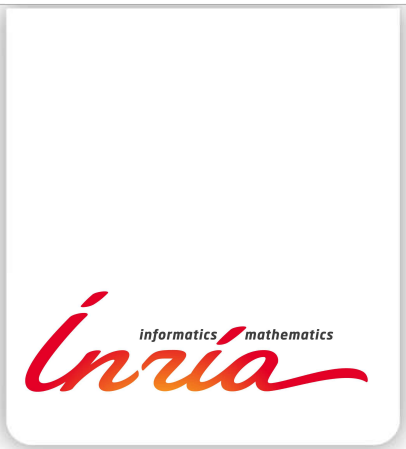
We similarly recover the sun radiance $\mathbf{L}_{\text{sun}}$ by taking a second picture of the gray card placed in sunlight ($v_{\text{sun}} = 1$). From this picture we have

$$\begin{aligned} \mathbf{I} &= \mathbf{R} * (\mathbf{S}_{\text{sun}} + \mathbf{S}_{\text{sky}} + \mathbf{S}_{\text{ind}}) \\ &= \mathbf{R} * (v_{\text{sun}} \cos\theta_{\text{sun}} \mathbf{L}_{\text{sun}} + \mathbf{S}_{\text{sky}} + \mathbf{S}_{\text{ind}}) \end{aligned} \tag{22}$$

where $\mathbf{L}_{\text{sun}}$ is the only unknown.

We finally align the environment map and sun with the reconstructed scene. To do so we manually mark a vertical edge of the reconstructed geometry and rotate the environment map and sun until the cast shadow of the virtual edge is aligned with that in the photograph.

We could envisage other approaches for this calibration. Cabral et al. [4] use a compass to compute the environment map orientation with respect to the view direction in each photograph. In Yu and Malik [42], a set of photographs of the horizon are taken, and a sky model is fitted. This involves taking several photographs, as opposed to the only two we take of the sphere, and also requires the use of two neutral lens filters to estimate sun radiance. In addition, fitting parameters of sky models can require specific tuning for relighting applications, as was the case for morning and evening sky in the work of Cabral et al. [4].



RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399