# Generative scene models with analytical path-tracing

Nima Keivan and Gabe Sibley

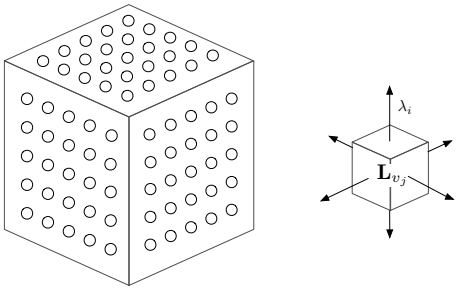


Fig. 1. Left: the voxel grid used to represent potential light sources in the scene. Right: The parameters used to represent each voxel consist of $\mathbf{L}_{v_j} \in \mathbb{R}^3$ which is the RGB intensity of the voxel and $\lambda_1, \ldots, \lambda_6$ which are the modulating intensities in the 6 principal directions. These parameters allow the representation of directional lights.

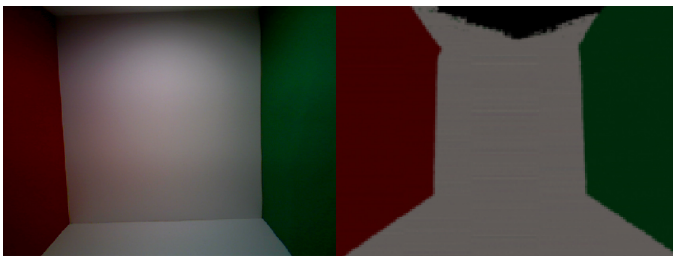

Fig. 2. Left: An RGB image captured from an RGBD sequence of an experimental Cornell box. Right: A rendering of the model with back-projected textures followed by clustering and albedo initialization. Note that the effects of direct and indirect lighting (including color bleed from the walls) has been removed.

## I. Introduction

Computer vision is often referred to as the "inverse graphics" problem. This is because many of the equations and relations used in computer vision find their roots in the understanding of image formation and light interaction. However, the complete process of image formation is mainly ignored in most applications of computer vision. In the case of localization and mapping, brightness constancy is assumed from multiple viewpoints and robust estimation is used to reduce or ignore the influence of any non-cooperative observations. While this approach works well for surfaces exhibiting Lambertian reflectivity, specular and transparent surfaces are often treated as outliers. This assumption is also the cause for the complexity of localization from a completely dense 3D map. While brightness constancy may apply for small baselines, it generally does not in the case of widely varying baselines.

With increased interest in augmented reality applications, estimating the lighting and materials in a scene finds new purpose beyond localization and mapping. Realistic rendering of virtual objects onto a scene requires knowledge of lighting for believable shadows and shading, as well as materials to properly show the interaction between the virtual object and the scene [10][6] [14]. Finally, including shading and lighting information in the estimation of geometry can lead to improvements and refinements which are not possible with more traditional multiple-view geometry methods [20].

The light transport equation (LTE) forms the basis by which physically based renders of synthetic scenes are generated in computer graphics [12]. Computationally, images are generated by evaluating the integral of the LTE over the scene using Monte Carlo integration techniques. This is known as "path tracing". Given the geometry of a scene, illumination sources, and the bidirectional reflection distribution function (BRDF) of surfaces which describe their reflective behavior, a rendered image can be obtained by tracing and integrating the LTE from illumination sources to the camera. This evaluation requires knowledge of all three components of the scene: geometry, materials and lighting.

The purpose of this work is to estimate a generative model for the scene including lighting, materials and geometry using the LTE as part of the objective function. The integration of the LTE is a *simulation* of the interaction of light with the scene. As such, any estimation performed with the LTE as part of the objective is affected by the random sampling performed in the Monte Carlo integration. The problem of estimation performed on objective functions with random coefficients is known as *simulation optimization* and has received much attention in the literature [3]. Following the notation in [3], the objective function is written as

$$\min_{x \in X} \hat{g}\left(x, \hat{\xi}_1, \ldots, \hat{\xi}_2\right), \tag{1}$$

where $\hat{g}$ is the estimate of the objective function, given the random coefficients $\hat{\xi}_1, \ldots, \hat{\xi}_2$, and $x$ is the parameter space over which the optimization takes place.

## II. Light Estimation

The problem of light estimation has received much attention in the literature. The approaches can generally be grouped into two categories: methods which require the light sources to be visible and reconstructed [10][6], and those which do not [19][16][5][1]. In general most solutions either assume that there are a small number of single point lights, or an array of lights that are infinitely distant. In this work, a voxel based lighting system is proposed which allows the flexibility of positioning lights anywhere within the scene, without the requirement to have the light be visible. Prior knowledge of the number of lights in the scene is also not required. Figure

1 shows the structure of the voxel grid as well as the parameterization of voxels. This particular parameterization allows both omnidirectional and directional lights to be modeled.

Given this parameterization, an optimization can be formulated with a cost per pixel defined as :

$$\mathbf{r}_i = \mathbf{I}_i - \int_{A_{film}} \int_{\omega} \dots \int_{\omega} \gamma \mathbf{L}_{e_u} f_j \left| \cos\theta_{j-1} \right| \tag{2}$$

where $\mathbf{I}_i \in \mathbf{R}^3$ is the RGB intensity of the $i$th pixel in the image, and the integral is the compact notation describing the Monte Carlo integration resulting in the predicted RGB intensity for the image. The first integral is evaluated by taking multiple samples per pixel, while the following integrals are sampled at every intersection of rays with the scene. For more information please refer to [12].

This cost can then be minimized by optimizing the parameters of every light voxel. An interesting property of the cost is that the geometric terms are unchanged as the voxel parameters are optimized. This allows the caching of the expensive intersections associated with path tracing into coefficients allowing both the re-computation of the required derivatives, and evaluation the cost itself. Performing this on the GPU allows fast optimization of a large number of light voxel parameters using first order gradient descent techniques.

An issue worth noting is that in this case the path trace operation is being performed without knowledge of the correct probability distribution function governing the integral. This is because the correct lighting parameters (which affect this distribution) have not yet been found. The initial sampling of the integrals may then be ineffective. In order to address this problem, multiple samples over the voxels are taken after each intersection (referred to as splitting) in order to sample as many voxels as possible. As the voxel values evolve, the sampling also becomes more focused on the voxels which contribute to the integral, improving the quality of the objective function, and consequently the lighting estimate.

Due to the potential over-parameterization of lighting and the large number of voxels, an activation penalty function is used as an additional cost. This logarithmic cost penalizes the initial activation of a voxel but then plateaus as the intensity of the voxel increases. Increasing the weight of this cost favors solutions with fewer voxels enabled.

## III. Material Estimation

A generative model for the scene would require high resolution texture information. Texture and geometry often have much different spatial frequencies. While it is possible to obtain very high resolution texture information by finely discretizing the geometry and coloring every vertex individually [18], the resulting mesh is needlessly complex and renders methods like path-tracing ineffective. The alternative solution is texture mapping. A great deal of research interest has been placed on how to best map 2D textures onto geometry [8] [15] [13]. However, since whether the texture can be represented on a 2D grid or not is not a significant factor in this work, the Mesh Colors [17] method is used for texture parameterization instead. The advantage of Mesh Colors is that it provides per-triangle texture mapping and discontinuities between the triangles are completely handled, resulting in a minimal parameterization of texture which is decoupled from the discretization of geometry. The parameterization is in the form of an array of RGB texels per triangle, with separate arrays for edges and vertices. The entire mesh is textured by back-projecting and averaging RGB keyframes onto the texels on a mesh fused using KinectFusion [11] and transformed into a triangle mesh using a GPU variant of the Marching Cubes [9] algorithm which produces a fully connected mesh.

A problem with this approach is that direct and indirect lighting incident on the surfaces will also be back-projected and ”baked into” the texels. Estimating lighting with this texture will result in uniformly lit surfaces, which is an incorrect generative model. This problem is somewhat remedied for specular materials, as the specular highlight can be detected and removed, however the effects of indirect and direct lighting will still be present in all but the most perfectly reflective of materials. The problem of splitting an image into its albedo and shading components has received much attention in the literature [2] [7]. However, most approaches have focused on images or video. In this work, the aim is to obtain the albedo values for texels on the surface of a 3D triangle mesh. The method by Garces et al. [4] can be expanded to 3D to obtain color clusters over the texels. The cluster centers can then be assigned to the texels as the initial albedo value.

Figure 2 shows the results of simple clustering in the color space followed by assignment of texels to the closest color, effectively removing the direct and indirect components of illumination and obtaining an initial guess for the albedo.

## IV. Geometry Estimation

Similar to the case of light estimation, Equation 2 can be used to estimate the 3D position of vertices. Unlike the case for light estimation, however, changes in vertex positions have downstream effects on the geometry and reflection terms, meaning that as vertex positions are updated, an entirely new path-trace operation has to be performed in order to compute derivatives. Furthermore, the derivatives themselves are non-trivial, as random sampling of the scene will involve multiple vertex parameters. In order to compute these derivatives, a sparse auto-differentiator was implemented which operates in parallel with the path tracer, carrying forward derivatives and involving new vertex terms as required.

An important design distinction is the treatment of normals. In computer graphics, vertex normals are an extra set of parameters which are then interpolated to obtain the normal of any point on a polygon. Rather than optimizing this extra set of parameters, the normal of each vertex is computed as the average of the normals of all triangles which involve that vertex, weighted by triangle area. This allows changes in vertex positions to directly affect surface normals, which is a an expected property of smooth surface functions. In order to compute the derivatives of the vertex positions on the normal

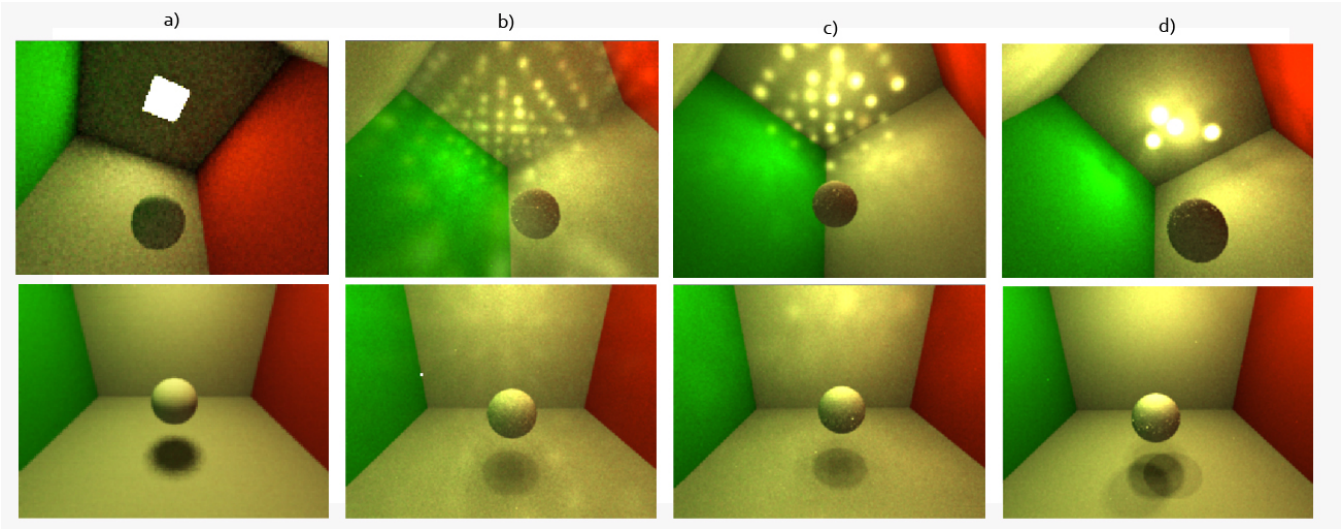


Fig. 3. Results of the light estimation experiment. Column a shows the reference lighting (top) and one of the 3 reference images (bottom) used to form the objective function. Columns b-d (top) show the result of the optimization with increasing activation penalties. Columns b-d (bottom) show the generated images using the lighting solutions in the top row.

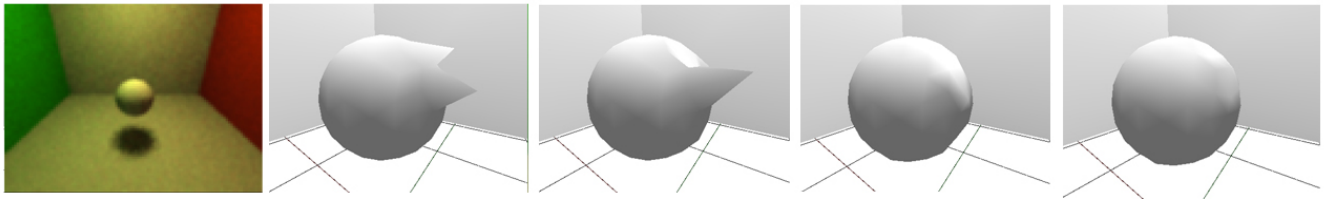

Fig. 4. Results of the geometry estimation experiment. The color image on the left is the single reference image at $106 \times 80$ pixels resolution. The images on the right show the shading-only rendering of the sphere with perturbed vertices during iterations 0-3.

vector, the sparse auto-differentiator is run in parallel with the normal compute operation. This operation is run at every iteration, before the auto-differentiated path-tracer, to update the normal derivatives.

Since constraints on the vertices are due to random sampling of the LTE in the scene, the 3D vertex position parameters are regularized with a weak prior to remove any null spaces due to sparse sampling.

## V. Results

The results of the system are focused on the estimation of voxel lights and vertex positions to demonstrate the feasibility of using the full LTE to optimize generative scene models. A simulated Cornell box is used in the experiments. The first experiment involves using the voxel light parameterization to estimate lighting in a scene which is illuminated by a square area light on the ceiling, as shown in Figure 3 a (top). To obtain the reference images, a GPU path tracer is used to render the scene from 3 known camera poses. The area light is then removed, and the optimization described in section III is run in order to iteratively estimate the voxel parameters and reduce the error between the reference image, and the images generated through voxel lighting. The optimization was performed on a grid of $7 \times 7 \times 7$ voxels using 3000 random pixels per image to form the residual. The top row shows the reference lighting (a) and the generated voxel lighting solution with increasing values of the activation penalty function (b-d). The bottom row shows one of the reference images (a) and the corresponding generated images for the voxel lightings in the top row (b-d). It can be seen that increasing the penalty function creates lighting solutions that are increasingly more different than the reference image, but fewer voxels are activated. In all cases, the generated voxel lighting produces images which are very similar to the reference image.

The second experiment focuses on the feasibility of estimating 3D vertex positions using the LTE as part of the objective function. In this case, a single low resolution reference view is rendered using the reference geometry as shown in Figure 4 (left). Two vertices of the sphere are then perturbed as shown in the Figure, and the geometry is iteratively optimized to reduce the error between the reference image and the one generated from the scene as described in section IV. The result of each iteration is shown in Figure 4. It can be seen that even though the 3D position of each vertex is optimized, the perturbations of the sphere are able to be corrected using only a single view.

## VI. Conclusions and Future Work

It has been demonstrated that using the full LTE equations for the purposes of lighting and geometry optimization is feasi-

ble. A number of new concepts have been introduced, such as directional voxel lights, and analytical path-tracing. However, the results have so far been in simulation environments where only one of the three facets (lighting, materials, geometry) of the problem were optimized, and the other two were known. Ongoing work is focused on performing inference on real data, as well as on simulation data where more than one facet of the problem is unknown.

## REFERENCES


[1] Xi Ning Steven McDonagh Peter Sandilands Robert Fisher Bastiaan J. Boom, Sergio Orts-Escolano. Point light source estimation based on scenes recorded by a rgb-d camera. In *Proceedings of the British Machine Vision Conference*. BMVA Press, 2013.

[2] Qifeng Chen and V. Koltun. A simple model for intrinsic image decomposition with depth cues. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 241–248, Dec 2013.

[3] Tito Homem de Mello and Gzin Bayraksan. Monte carlo sampling-based methods for stochastic optimization. *Surveys in Operations Research and Management Science*, 19(1):56 – 85, 2014.

[4] Elena Garces, Adolfo Munoz, Jorge Lopez-Moreno, and Diego Gutierrez. Intrinsic images by clustering. *Computer Graphics Forum (Proceedings of the Eurographics Symposium on Rendering)*, 31(4), 2012.

[5] K. Hara, K. Nishino, and K. Ikeuchi. Multiple light sources and reflectance property estimation based on a mixture of spherical distributions. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1627–1634 Vol. 2, Oct 2005.

[6] S.B. Knorr and D. Kurz. Real-time illumination estimation from faces for coherent rendering. In *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 113–122, Sept 2014.

[7] KyongJoon Lee, Qi Zhao, Xin Tong, Minmin Gong, Shahram Izadi, SangUk Lee, Ping Tan, and Stephen Lin. Estimation of intrinsic image sequences from image+depth video. In *Computer Vision ECCV 2012*, volume 7577 of *Lecture Notes in Computer Science*, pages 327–340. Springer Berlin Heidelberg, 2012.

[8] Bruno Levy, Sylvain Petitjean, Nicolas Ray, and Jerome Maillot. Least squares conformal maps for automatic texture atlas generation. *ACM Trans. Graph.*, 21(3):362–371, July 2002.

[9] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *ACM siggraph computer graphics*, volume 21, pages 163–169. ACM, 1987.

[10] M. Meilland, C. Barat, and A. Comport. 3d high dynamic range dense visual slam and its application to real-time object re-lighting. In *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*, pages 143–152, Oct 2013.

[11] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality*, ISMAR '11, pages 127–136, Washington, DC, USA, 2011. IEEE Computer Society.

[12] Matt Pharr and Greg Humphreys. *Physically Based Rendering, Second Edition: From Theory To Implementation*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2nd edition, 2010.

[13] Nicolas Ray, Vincent Nivoliers, Sylvain Lefebvre, and Bruno Lévy. Invisible seams. In *Computer Graphics Forum*, volume 29, pages 1489–1496. Wiley Online Library, 2010.

[14] K. Rohmer, W. Buschel, R. Dachselt, and T. Grosch. Interactive near-field illumination for photorealistic augmented reality on mobile devices. In *Mixed and Augmented Reality (ISMAR), 2014 IEEE International Symposium on*, pages 29–38, Sept 2014.

[15] Alla Sheffer, Bruno Levy, Maxim Mogilnitsky, and Alexander Bogomyakov. Abf++: Fast and robust angle based flattening. *ACM Trans. Graph.*, 24(2):311–330, April 2005.

[16] T. Takai, A. Maki, and T. Matsuyama. Self shadows and cast shadows in estimating illumination distribution. In *Visual Media Production, 2007. IETCVMP. 4th European Conference on*, pages 1–10, Nov 2007.

[17] Cem Yuksel, John Keyser, and Donald H. House. Mesh colors. *ACM Trans. Graph.*, 29(2):15:1–15:11, April 2010.

[18] Qian-Yi Zhou and Vladlen Koltun. Color map optimization for 3d reconstruction with consumer depth cameras. *ACM Trans. Graph.*, 33(4):155:1–155:10, July 2014.

[19] Wei Zhou and Chandra Kambhamettu. Estimation of illuminant direction and intensity of multiple light sources. In *Computer Vision ECCV 2002*, volume 2353 of *Lecture Notes in Computer Science*, pages 206–220. Springer Berlin Heidelberg, 2002.

[20] Michael Zollhöfer, Angela Dai, Matthias Innmann, Chenglei Wu, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Shading-based refinement on volumetric signed distance functions. *ACM Transactions on Graphics (TOG)*, 2015.