

Shape From Tracing Report

Loudon Cohen

loudon.cohen@brown.edu

Advisor: Daniel Ritchie

Forward & Abstract

The following is a supplemental research report regarding the individual contribution of the present author to the paper Shape from Tracing: Reconstructing 3D Object Geometry and SVBRDF Material from Images via Differentiable Path Tracing [1]. As this work is, as of yet, unpublished, the following includes abbreviated elements for context and older results. One may refer to the footnotes for details regarding individual contribution which mainly entail material reconstruction.

In this preliminary work, we explore the capability of differentiable path tracing for inverse rendering. We present a full pipeline for reconstruction of 3D geometry and texture from a series of 2D images all while capturing complex physically-based phenomena through global illumination.

1. Introduction

High-quality, realistic 3D content is a critical asset across several industries—film, games/interactive VR, AV simulation, to name a few—where its functions range from imparting a sense of visual realism to allowing a user to perform destructive operations that would not be possible in the real world. Creating such content by hand requires skill and is labor intensive, often requiring large teams of artists to be done at even moderate scale. For these reasons, acquiring it directly from real-world objects is an attractive alternative.

There are many approaches for doing this, including MVS, volumetric fusion using time-of-flight sensors, etc. All share the same core approach: taking multiple views of the object/scene via some sensor, and using that information to infer the underlying geometry and material properties.

Most 3D reconstruction systems do this bottom up: fusing observed sensor information together to construct a 3D model. However, there is another approach inspired by the alternative school of thought of “vision as inverse graphics”: generate a 3D model that, when rendered, is consistent with the 2D observations.

While this view has a long history and is theoretically el-

egant, it is difficult to realize in practice and its applications have been limited. Now, new tools are available which merit revisiting these ideas in detail. In particular, differentiable path tracing allows a full global illumination renderer to be added to any gradient-based optimization pipeline, making it possible to optimize for renderer output.

In this paper, we present the first system which performs multi-view 3D reconstruction via differentiable path tracing. Given multiple calibrated views of an object under known lighting, our method reconstructs both the 3D geometry of the object (as a surface mesh model) as well as the material of the object.

For material modeling, the system supports a Torrance-Sparrow SVBRDF model, a physically-based “microfacet” model that is a standard within the film and games industries. Using such a model allows our system to capture a wide range of materials, from highly specular plastic to diffuse drywall.

Through simulation studies, we demonstrate that the use of a full global illumination renderer results in better reconstructions in settings with non-local illumination effects (i.e. effectively disentangling surface albedo from illumination). Moreover, our pipeline acts as a refinement upon input fed from other reconstruction approaches.

2. Related Work

Given the age and prevalence of the shape-versus-shading problem, there are many approaches for inverse modeling. Classic approaches generally consist of multi-view stereo [14] and/or space carving [5] techniques in the multi-image domain. Results in this area have also been furthered with deep learning versions of these same techniques [4].

Also with the advent of deep learning-based approaches, another mode of reconstruction has emerged in the form of neural rendering. These approaches tackle the act of learning to render: here meaning converting a representation such as (but not limited to) 3D into a 2D space. As these representations are designed to be naturally differentiable, it becomes a simple training process to optimize with respect to real targets. Li et al. use this approach to capture

material but not geometry from a single mobile phone image [7]. Likewise, Peterson et al. use neural rendering to capture geometry with no texture variation [12]. Most recently, Lombardi et al. use a voxel-based method with a differentiable raymarching operator to recover both shape and texture [8].

While neural rendering is a promising field of study for inverse graphics, creating such systems can take considerable amounts of data and training resources to produce reasonable results. An alternative is more traditional renderers with differentiable rendering integration [9]. Generally these techniques use some amount of jitter, blur, or explicit sampling to smooth out edge discontinuities. Among these renderers, two utilize path tracing for lighting estimation: Redner [6] and Mitsuba 2 [11]. As global illumination allows for rendering of physically based phenomena, these renderers allow for the most realistic inverse rendering. We build off of the Redner framework for our work.

3. Method

Our strategy can be broken up roughly into three components—setup, geometry and texture initialization, and geometry and texture optimization—which are described in detail in their respective subsections.

3.1. Setup

In simulation, we begin with a set of rendered images of our target object, simulating real-world photographs. We use a Fibonacci sphere distribution to determine the positions of 32 cameras, each with two different zoom levels, giving us 64 distinct views of the object in total. We set the object against a completely black background that contributes no illumination. The illumination is introduced in a controlled way via a single point light, placed at a constant offset to the camera. This setup represents the best case scenario for our system to determine the best hypothetical reconstruction quality.

As for real world results, we take a high-resolution video circling the hemisphere around the object in an unconstrained environment. As an approximation to the lighting conditions, we also use a 360 degree camera co-located with the object’s original position.

3.2. Geometry & Texture Initialization

We have experimented with many different methods of constructing an initial guess for input (Figure 1). We have found that voxel carving generally performs best for simulated data and that COLMAP [13] performs best for real world data ¹.

¹COLMAP/Redner interoperation was a significant group effort. In addition, I was responsible for creating appropriate dense model reconstructions as initialization to our pipeline.

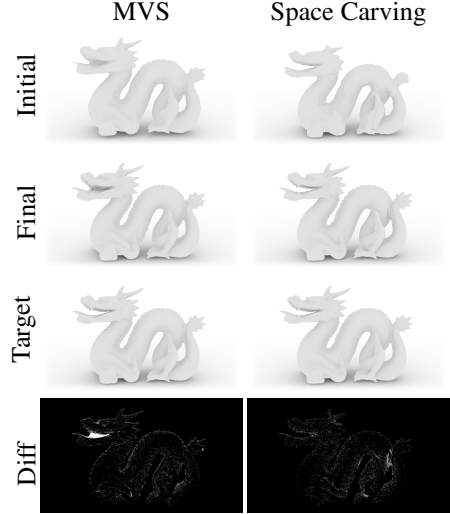


Figure 1. Visualization of different initialization strategies for our reconstruction system. Row one shows initializations produced by COLMAP’s multi-view stereo reconstruction (top left), voxel carving (top right), Row two shows the results of geometry-only optimization starting from each corresponding initialization. Row four shows a difference (2x magnified) between the final reconstruction result and the target ground truth model.

3.3. Geometry & Texture Optimization

After choosing an initial guess, the next step is to refine the position of its vertices with respect to the target images. This way, we can capture the target’s desired shape and topology. Through gradient descent, we shift the positions of the vertices and calculate a loss value by comparing the rendered views of the intermediate mesh with the initial target images. In simulation, the varying zoom levels of the target images encapsulate both coarse and fine surface detail to capture. In order to automatically get finer detail where relevant, we periodically perform subdivision and quadric simplification on the intermediate mesh. This results in adaptive remeshing, where vertices that remain after simplification are pushed towards areas of greater detail. Lastly, since gradient descent over vertex positions can result in noisy mesh interiors and backwards-facing triangles, we resample the surface of the mesh by raycasting from the original viewpoints. By performing a Poisson reconstruction on the resulting pointcloud, we continue the optimization with a new, clean mesh.

As traditional UV texture mapping doesn’t lend itself to simultaneous optimization with significantly changing geometry, we employ a per-triangle color representation. Specifically, we use a differentiable version of the Mesh Colors [15] extension of vertex colors, which defines a color for vertices, edges, and faces of triangles. To extract texture information for objects, we simply optimize with respect to the Mesh Color texels per-triangle in order to match the

Object	Condition	F1	Full	Diffuse	Roughness
<i>Dragon</i>					
	Local + Shadows	87.51	21.922, 0.8591	27.815, 0.8884	21.450 , 0.7564
	Global	87.51	27.546 0.9003	28.150 , 0.8968	21.418, 0.7633

Table 1. Quantitative results for Figure 2. Geometry is kept the same when global illumination is introduced so therefore F1 score is the same for both lighting conditions. F1 score is computed with a tolerance of 0.01. Average PSNR (left value) and SSIM (right value) are computed as the render accuracy for each material acquisition.

same target images. For further details see Appendix A.²

As joint shape and texture optimization introduces a large parameter space, we currently alternate between texture and geometry optimization steps. In simulation, loss for both of these optimization steps is the summed L2 distance of gaussian pyramids across all camera views.³

4. Results & Evaluation

4.1. Full System

Here, a single example from an early iteration of [1] is provided: the Stanford dragon with a spatially varying wood-grain texture which can be seen in Figure 2. The improvement of material reconstruction for this case is shown in Table 1.⁴

4.2. Ablations

For sake of brevity in this report, ablations of the pipeline are omitted. However, for evaluations regarding number of input views as well as local illumination vs. local illumination + shadows vs. global illumination for geometry and texture, one may reference the paper [1].⁵

4.3. Comparisons

While this work is somewhat unique in its simultaneous capture of texture and geometry, parts of the pipeline can be compared against other methods.

For instance, the quality of material reconstruction can be evaluated against that of SVBRDF-only techniques [2, 3]. To compare with these methods, we rendered a flat plane orthographically in simulation and ran just the material capture section of our pipeline. Notably, this optimization also included normal maps for a full comparison. Due to the dif-

ferences in material model, results are compared between each respective ground truth and not each other.

Preliminary results show that, while the neural approaches have a significant advantage in material smoothness due to their learned prior, our approach can get increased re-render accuracy. At the very least, we show that hundreds of images are not necessary for an accurate material reconstruction [2]. For conclusions on this comparison as well as refinement of neural geometric systems, one may refer to [1].⁶

5. Conclusion & Future Work

We have found that our method works well under simulation. In controlled environments, where we simulate photographs using rendered images, and we assume the object’s materials are diffuse, our method can reconstruct the shape and texture of 3D objects. One may refer to [1] for our results regarding real world data.

The main limitation of our system is falling into local minima when optimizing from a poor initialization. As such, we present this work, not as a standalone method, but as a refinement step for existing methods.

In the future, this pipeline may be extended to include materials other than the diffuse and partially specular objects shown, such as mirrors and glass. However, through just these preliminary results, we show that differentiable rendering is a useful tool for reconstructing 3D scenes from a set of images via joint lighting, shape, and texture estimation.

References

- [1] Loudon Cohen and anon. Shape from tracing: Reconstructing 3d object geometry and svbrdf material from images via differentiable path tracing. 2020.
- [2] Valentin Deschaintre, Miika Aittala, Frédo Durand, George Drettakis, and Adrien Bousseau. Flexible svbrdf capture with a multi-image deep network. *Computer Graphics Forum (Eurographics Symposium on Rendering Conference Proceedings)*, 38(4):13, jul 2019.
- [3] Duan Gao, Xiao Li, Yue Dong, Pieter Peers, Kun Xu, and Xin Tong. Deep inverse rendering for high resolution SVBRDF estimation from an arbitrary number of images. *ACM Transactions on Graphics*, 37(4), July 2019.
- [4] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. *CoRR*, abs/1804.00650, 2018.
- [5] Kiriakos N. Kutulakos and Steven M. Seitz. A theory of shape by space carving. *Int. J. Comput. Vision*, 38(3):199–218, July 2000.
- [6] Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. Differentiable monte carlo ray tracing through edge

²I was solely responsible for this implementation which required direct modification to the Redner framework. As such, I created a unique indexing scheme which is addressed in Appendix A.

³While the optimization cycle was a large group effort of tuning, I specifically authored the main optimization script, Google Colab tools for online editing, and pyramidal loss.

⁴I generated the content (not the renderings) for all of the included diagrams [1, 2] and evaluated the material results in Table 1.

⁵I was responsible for an ablation on the number of camera view inputs.

⁶I was solely responsible for material comparison work which involved running both systems on the same input and evaluating each upon their respective ground truth renders.








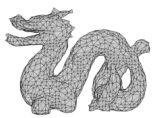

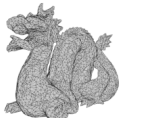














Object	Render	Target	Init	1 bounce		2 bounce	
Dragon	Full						
	Geo						
	Albedo						
	Roughness						

Figure 2. Our system’s reconstruction of the Stanford dragon model, textured with a spatially-varying glossy wood material. The optimization used 32 cameras distributed in a Fibonacci sphere, each with 2 light angles (one offset along the tangent and one aligned with the camera), making for a total of 64 images. The image resolution was 128×128 , except for the final texture optimization cycle. This uses an image resolution of 256×256 , and optimizes a UV map texture representation, to facilitate rendering under novel illumination conditions. The final target images and optimization images were rendered with a sample count of 512 spp and 64 spp respectively.

- sampling. *ACM Trans. Graph.*, 37(6):222:1–222:11, Dec. 2018.
- [7] Zhengqin Li, Kalyan Sunkavalli, and Manmohan Chandraker. Materials for masses: SVBRDF acquisition with a single mobile phone image. *CoRR*, abs/1804.05790, 2018.
- [8] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Trans. Graph.*, 38(4):65:1–65:14, July 2019.
- [9] Matthew M. Loper and Michael J. Black. OpenDR: An approximate differentiable renderer. In *Computer Vision – ECCV 2014*, volume 8695 of *Lecture Notes in Computer Science*, pages 154–169. Springer International Publishing, Sept. 2014.
- [10] Ian Mallett, Larry Seiler, and Cem Yuksel. Patch textures: Hardware implementation of mesh colors. In *High-Performance Graphics (HPG 2019)*. The Eurographics Association, 2019.
- [11] Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and Wenzel Jakob. Mitsuba 2: A retargetable forward and inverse renderer. *Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 38(6), Nov. 2019.
- [12] Felix Petersen, Amit H. Bermano, Oliver Deussen, and Daniel Cohen-Or. Pix2vex: Image-to-geometry reconstruction using a smooth differentiable renderer. *CoRR*, abs/1903.11149, 2019.
- [13] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016.
- [14] Steven M. Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1*, CVPR ’06, pages 519–528, Washington, DC, USA, 2006. IEEE Computer Society.
- [15] Cem Yuksel, John Keyser, and Donald H. House. Mesh colors. *ACM Trans. Graph.*, 29(2):15:1–15:11, Apr. 2010.

Appendices

A. Mesh Colors

In our implementation, a mesh colors texture is stored in a one-dimensional array \mathbf{a} , the size of which is determined by the number of triangles in the mesh and an integer resolution level r . Given the barycentric coordinates (α, β) of a point in mesh triangle number t , the texel at that can point is defined by

$$\begin{aligned}\mathbf{m}(r, t, \alpha, \beta) &= \mathbf{a}[k] \\ k &= \frac{t \cdot (r + 1) \cdot (r + 2) + i \cdot (2r - i + 3)}{2} + j \\ (i, j) &= \lfloor r \cdot (\alpha, \beta) \rfloor\end{aligned}$$

Note that this storage scheme duplicates edge and vertex detail for parallelism at the slight cost of additional memory. This approach more closely matches that of the newer Patch Textures [10], rather than the original mesh colors. To compute derivatives for optimization, we use finite differences (necessary for all discretely-sampled representations of texture, including UV maps).