Supplement to Objects as Cameras: Estimating High-Frequency Illumination from Shadows

Tristan Swedish, Connor Henley, and Ramesh Raskar

Massachusetts Institute of Technology
Cambridge, MA, 02139, USA
{tswedish, co24401, raskar}@mit.edu

1. Regularization Parameters

For all experiments in the paper, we used the following values for the regularization parameters in Equation 6 and Equation 7 in the main text, λ_s: 600, λ_r: 100, λ_p: 1. We used a slightly different set of parameters for Figure 10 in the supplement, due to the challenge of model mismatch, λ_s: 1, λ_r: 200, λ_p: 1.

2. Additional Results

Teapot In addition to the results in Figure 4 in the main text, we show environment map estimates using a Utah Teapot in Figure 1. The quality of the reconstructions is not as good as the bunny results shown in the paper, but do appear to be successful. The teapot is rounded and is comparable to the sphere in Figure 7 of the main text, which we expect to produce lower resolution reconstructions than the bunny object. Interestingly, the same bright pixel artifact seen to the top right appears in both the teapot and bunny reconstructions. We hypothesize that this pixel corresponds roughly to the retro-reflective path from the perspective of the camera.

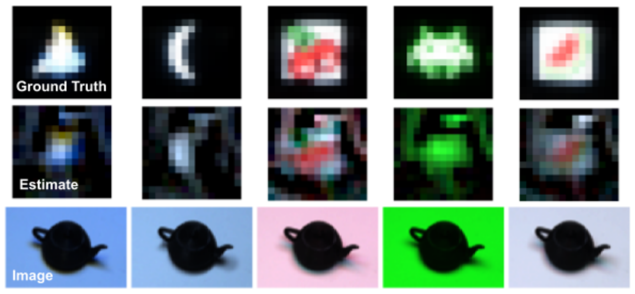

Figure 1. Estimated environment maps for a 3D printed teapot. In the top row, we show a cropped ground truth environment map centered on a display placed above the object. The middle row shows the estimated environment map with the same crop applied, and the bottom row shows the observed images captured using the camera.

Real Data using 3D Printed Object The main text shows environment map estimates from rendered models and realistic environment maps. In Figure 2, we show estimated environment maps captured using a 3D printed bunny on a white sheet of paper placed in various real world environments. For each estimate, we first determine the pose of the camera using a known printed pattern printed on the white paper near the bunny location, we obtain correspondences from the camera image to the known pattern and use the OpenGV [3] library implementation of the UPnP [4] algorithm to obtain the camera pose. Using the known intrinsic parameters of the camera, we mirror the scene in Mitsuba2 by matching camera parameters to Mitsuba's pinhole camera model. Since the 3D printed object was not affixed to the sheet of paper, we also perform a manual alignment step to ensure the 2D position and 1D rotation of the bunny match the captured image. We point out that much of this pipeline could be automated using standard object detection and pose estimation techniques. We also suggest that future work could utilize Equation 1 to refine the object position directly using a renderer capable of calculating derivatives of geometric parameters, such as the pose of a mesh.

Overall, the reconstructions are reasonable, and all major bright sources in the scene are recovered. It may not be surprising that the sun is recovered, but for the more indirectly illuminated scenes, the reconstructions contain more interesting detail. For example, in the second major row of Figure 2, most of the blue sky and a specular reflection of the sun from a building surface are recovered. In the first major row, the square shape of the window is recovered, as well as the light source in the upper left corner. There appear to be some artifacts near the horizon, as expected and discussed in relation to Figure 7 in the main text.

Notably, the reconstructions appear to fixate on the brightest sources, but when the brightest sources are blocked, as in the last row of Figure 2, the resulting environment map estimates contain less bright sources. It is a topic of future work to explore the effective dynamic range of estimated environment maps.

Differential Illumination Estimation One application of using objects as cameras could be for static surveillance cameras that observe a scene over a period of time. While most of the environment map will remain unchanged, can a person walking by be detected in the subtle changes to the observed image?

In Figure 4, we captured a sequence of four images of the bunny, each at a different point in time as a person was walking by. As seen in the top row ground truth environment maps, the person is moving from right to left from the perspective of the object. We preprocess the observed images by subtracting the first frame from the last three frames. As such, our observed images contain both positive and negative values, as seen in the bottom row of Figure 4. Given these observed images, we perform the environment map estimate without the non-negativity constraint, so the estimated environment map contains both positive and negative values. We can observe a moving negative region as the person blocks the sky behind them, and a brighter region where the sky was originally blocked in the first frame.

Such an approach could be used to detect subtle variations in the observed scene, and this experiment demonstrates the feasibility of estimating "differential environment maps" that explain how the observed scene changes as the surrounding illumination changes.

Albedo Estimation using Realistic Environment Maps In the main text we demonstrated the simultaneous estimation of albedo and illumination for somewhat simple localized light sources. We show a similar result in Figure 5, but in this case use more realistic environment maps. We found that letting the alternating update method run to convergence led to poor estimates of the individual environment maps, while the albedo appeared largely unchanged for much of the optimization. Instead, we show albedo and environment map estimates after only 3 iterations. The resulting albedo looks reasonable, and the environment maps appear to correspond far more closely to ground truth than the results obtained after full convergence.

More analysis is necessary to determine when our alternating update scheme is successful, but this result highlights some of the inherent difficulty of this setting. In particular, it appears as though an excess of energy was placed in the top center of the fully converged environment map estimates, which corresponds to the retro-reflective light direction. This led to a reduction in residual error but did not improve results.

3. Robustness to Sensor Noise

Estimates with Simulated Noise Regardless of the accuracy of a given ray transport matrix used for reconstruction, any real image will have some amount of noise from the sensor. This includes shot noise caused by the random arrival time of photons, which is a Poisson process, and noise from the analog electronics in the sensor from thermal effects or dark current, which can be approximated by a Gaussian process. As such, we start with a "No Noise" image obtained from a path tracer with sufficiently high samples per pixel. We then add Poisson noise based on the pixel intensities of the "No Noise" image and add Gaussian noise with some standard deviation to the shot noise image.

We show results in Figure 7, where we plot the structural similarity index measure (SSIM) of the ground truth environment map compared to the estimated environment map given a noisey image. We see that reconstructions only begin to significantly degrade once sufficient Gaussian noise is added to the observed image. This provides some confidence that our method should work under realistic noise environments.

Variance from Path Tracer in Synthetic Results For all of our synthetic results, we render observed images using a path tracer. This is to ensure we are actually solving the inverse problem, as opposed to generating synthetic observed images using the ray transport matrix itself. If the observed images are rendered without enough samples per pixel, estimated environment maps are poor due to the variance of the path tracer's monte carlo estimate of the true image. This variance does not have an obvious physical analog, so we prefer to use sufficient samples to avoid problems. Figure 8 shows a range of values for the samples per pixel and the corresponding reconstruction results. We used these reconstruction results to determine how many samples per pixel to use in our synthetic experiments.

Using a Higher Resolution Ray Transport Matrix In the main text we used environment maps with a resolution of 32×64, which provided a decent trade-off between computation time and resolution. However, there is no reason that our method could not be used with higher resolutions. Unfortunately, the use of higher resolutions comes at a significant computational cost, as doubling the resolution increases the ray transport matrix memory requirements and flops for a forward calculation by $4\times$. Inverting $A^\top A$ has a higher computational cost as the ray transport matrix increases in size. While out of the scope of the paper, the uncertainty maps in Figure 7 of the main text may provide a useful cue for irregular-sampling of the ray transport matrix

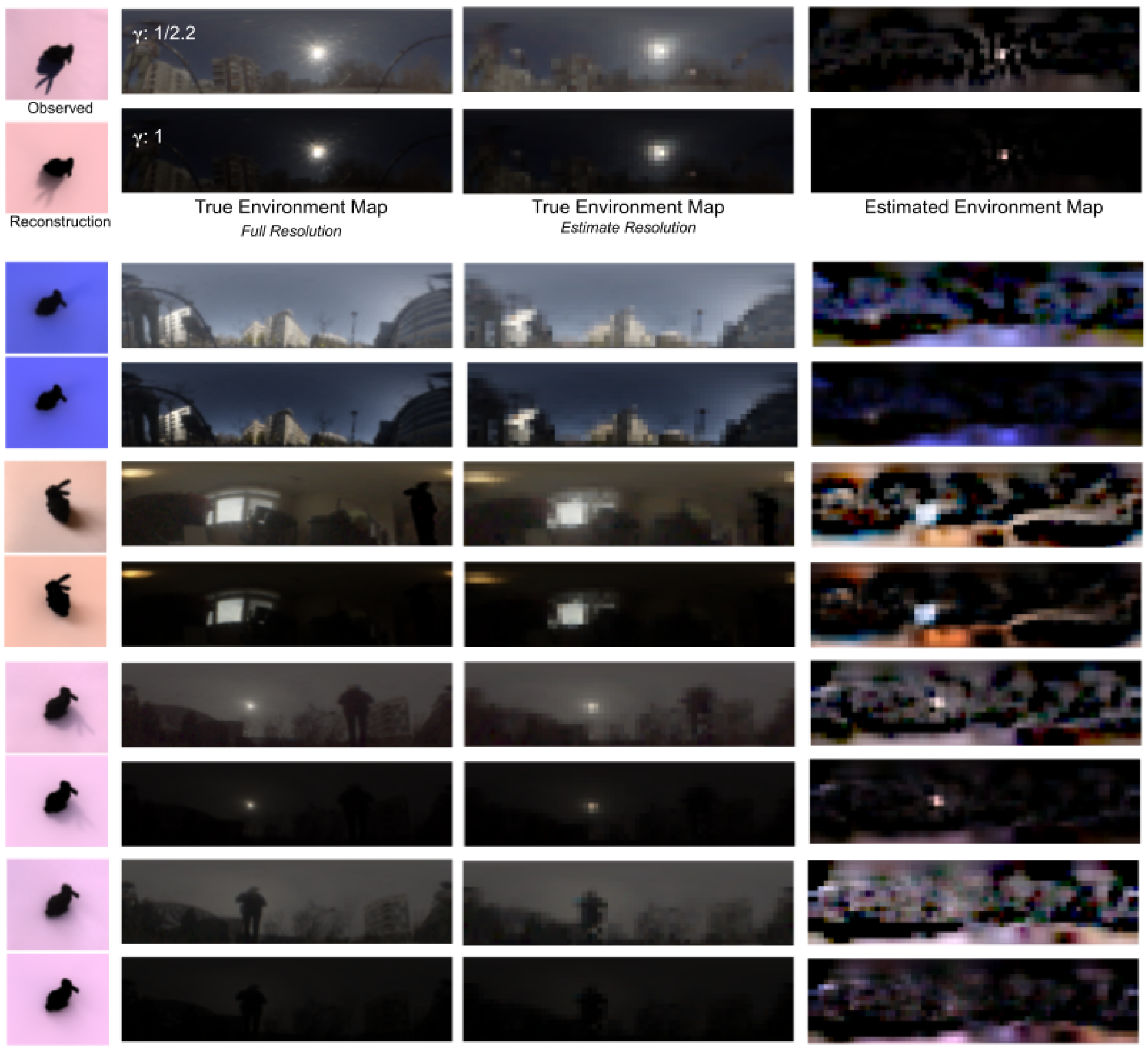

Figure 2. Environment map estimates using real data captured using the 3D printed bunny placed on a white sheet of paper and placed in various real environments. The left column shows the observed image captured by the camera and the corresponding reconstruction from the estimated environment map. The middle two columns shows a full-resolution ground truth environment map displayed with two gamma correction factors and the same environment map resized to the dimensions of the estimated environment map. The right column shows estimated environment maps. All environment maps are shown with two gamma correction factors to highlight the brightest sources in the scene.

such that higher resolutions can be preserved while reducing the total size of the matrix. Below we discuss further some of these issues and potential future directions when scaling this method.

4. Additional Sources of Model Mismatch

The paper assumes much of the object and surrounding surface is known, such as the surface geometry and material properties. While we show some ability to handle unknown diffuse albedo, there are other sources of model mismatch that lead to failed illumination estimates. To provide such a case study, we attempted to use photogrammetry to create the surface mesh and approximate material properties from a partial set of views. In this way, the complete model would not be needed *a priori* and could be obtained using standard computer vision tools. Results can be seen in Figure 10. We believe these results highlight additional sources of model mismatch that can be addressed in future work. We also propose a simple extension to the paper that may handle

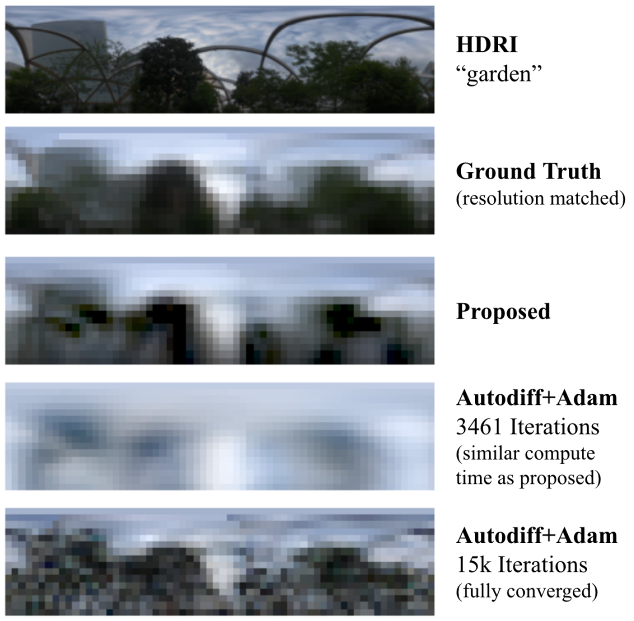

Figure 3. Visualization of main text Table 1 results. Predicted environment maps compared to the gradient descent baseline for two stopping criteria. Within a similar computation time as the proposed method gradient descent produced lower quality estimates, requiring a greater number of iterations to converge. While convergence may be accelerated by using a larger learning rate, this led to poorly converged results. Table 1 in the paper contains the RMSE/SSIM values for these images.

some of these sources of error.

Below, we describe our photogrammetry experiments. We make use of an existing photogrammetry pipeline Meshroom [5, 2, 1] to recover camera poses, scene geometry, and diffuse albedo.

Using Meshes captured from Photogrammetry In practical settings, known 3D shape and camera orientation are rarely available. We demonstrate our approach when the visible surface is entirely estimated by photogrammetry. We capture 10 images along a line about 2 meters away from the object, sampling a limited set of viewpoints within about a 20 degree angular fan originating from the object. The recovered surface mesh from photogrammetry does not model the back surfaces of the objects (see Figure 10a).

In order to obtain better results of texture estimates of our surface mesh, we found it useful to capture two images at each camera pose. The first image is taken with a flash, and the second with no flash. We subtract out the non-flash image from the flash image to remove all non-flash illumination from the image. The recovered flash-only images led to much better results in the full surface estimation pipeline.

Masking out Object Surfaces For all of our experiments, we create a segmentation mask to remove the pixels of the object in the scene. We found that the reconstructions still succeed in these cases, but the region of the environment map pointed back towards the camera (i.e. the retro-reflective direction) sometimes contains bright phantom sources. We hypothesize that this is due to the lack of shadow information for these illumination orientations (the object blocks the image of the shadow from the camera perspective). For this reason, object surfaces can help reduce this ambiguity. Adding object surfaces back into the reconstructions reduce these artifacts, but do not seem to significantly improve reconstruction in other incident illumination directions.

While the reconstructions in Figure 10 show the two brightest reconstructions (the blue on the left and yellow on the right), the estimated environment map contains other artifacts. To this end, we hypothesize the following sources of error are highly influential, and could be addressed in future work. Given a ray transport matrix without these sources of model mismatch, the main text demonstrates that high frequency environment maps can be estimated.

Surface Mesh Error In Figure 10a, the carpet surface seems well captured. However, the resulting geometric texture produces a complex appearance for grazing illumination angles near the horizon. If the recovered surface mesh is incorrect, a small error in this mesh could cause it to cast a shadow over many observed pixels. Simplifying the resulting mesh and baking in this fine surface detail into normal maps may reduce the effect of false self-shadowing while still maintaining realistic appearance for less extreme incident illuimation angles.

Albedo Estimation Error We use flash-only images (using the processing described above) for photogrammetry and albedo estimation. While the resulting textures appear correct, small stitching artifacts and other artifacts from non-uniform illumination from the flash are apparent upon close inspection. While our method can be somewhat robust to diffuse albedo under certain conditions, reducing these artifacts may be beneficial.

Object Large relative to illumination distance Interestingly, the estimated environment map in Figure 10 corresponding the "Dragon" appears compressed relative to ground truth. The two brightest sources appear to be shifted slightly to the left and are closer together than expected. It's important to note that the dragon is actually fairly large, about 30cm tall, while the nearest illumination source (the blue-tinted source) is only about 0.5 meters from the dragon. Furthermore, the other bright source, the yellow-tinted luminaire is only about 2 meters from the dragon center. In this situation, the dragon's size may introduce a warped environment map estimate due to parallax. The use of environment maps may be inappropriate for nearby light sources, and instead a more general 4D light field may be necessary to account for such effects. Regardless,

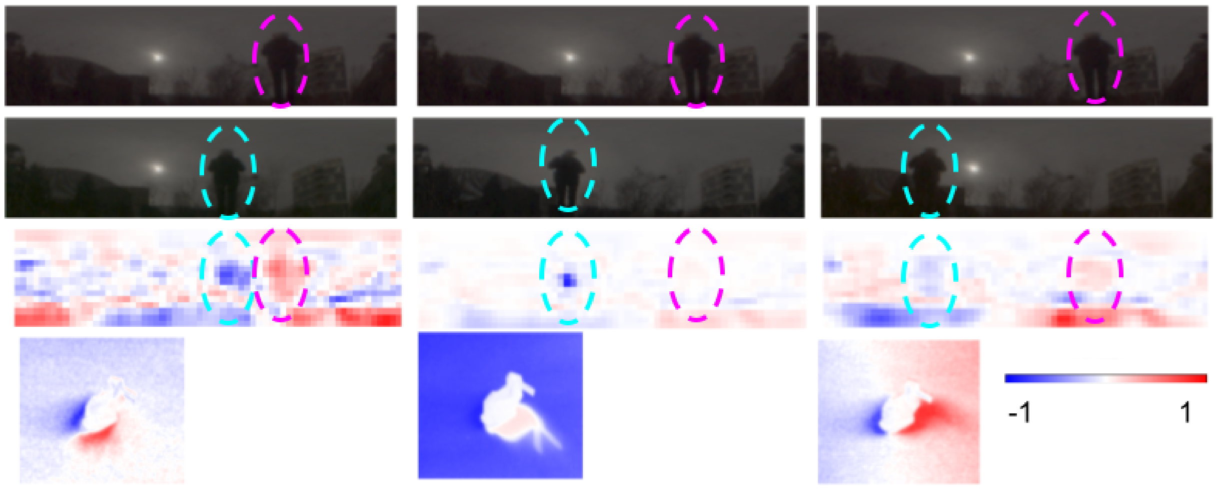

Figure 4. Estimated "differential environment maps" given a temporally subtracted image. The top row shows the starting frame from a sequence of 4 images with a human figure (magenta). The second row shows the subsequent 3 images in the sequence as the human figure (cyan) moves from right to left in the frame. The third row shows the estimated environment maps without a non-negativity constraint applied, showing how adding a figure subtracts light (cyan), and removing a figure adds light (magenta) to the scene from particular directions. The bottom row shows the observed differential images taken by subtracting the observed image corresponding to the top row from the observed image corresponding to the second row. All images have been normalized with a maximum value of either [-1,1] for display purposes.

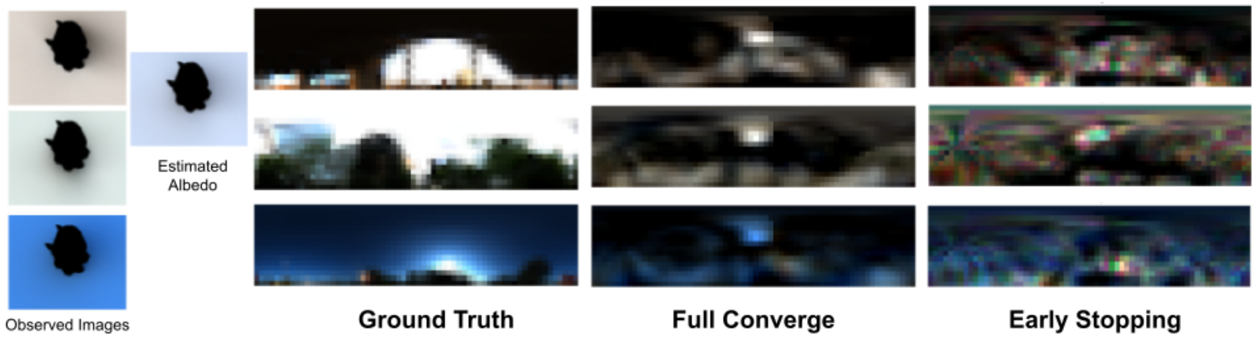

Figure 5. Simultaneous estimation of illumination and albedo for realistic outdoor environment maps. Solving the same problem shown in Figure 3 of the main text, but with realistic extended sources. On the left column we see three observed images. Given known geometry but unknown albedo, a ray transport matrix was generated using a white albedo for all surfaces. The second column shows the per-pixel estimated albedo after convergence. The environment maps associated with each observed image are shown in the next 3 columns: (third column) The ground truth environment map for 3 realistic scenes, (fourth column) estimated environment maps after running all iterations, (fifth column) estimated environment maps when running for 3 iterations where more structure is preserved.

warped estimated environment maps may still be useful for re-lighting or applications where accurate angular recovery is not needed.

Updating the Ray Transport Matrix We might also attempt to reduce model-mismatch by optimizing the ray-transport matrix using a differentiable renderer. While we did not make use of the feature in the experiments, Mitsuba2 [6] can calculate gradients of internal scene parameters with respect to rendered pixels. We sketch out a simple way to incorporate our lighting estimates to iteratively update the ray transport matrix. This would be very useful, as effects such as global illumination are not modeled by our linear diffuse albedo model, but could be "baked-in" to the ray transport matrix. In this way, we could alternately solve for illumination and rendering parameters to account for non-linear interactions of parameters such as color bleeding.

Given an initial guess of the scene parameters, θ, and an illumination estimate, $\tilde{\mathbf{x}}$, we can minimize the following objective.

$$\begin{aligned} \arg\min_{\theta} \quad & ||f_\theta(\tilde{\mathbf{x}}) - \mathbf{y}||_2^2 + \lambda_i R_i(\theta) \\ \text{s.t.} \quad & C(\theta) > 0 \end{aligned} \tag{1}$$

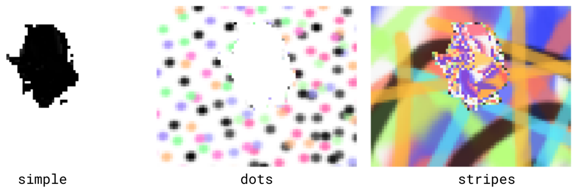

Figure 6. Ground truth albedo maps used to compute the RMSE/SSIM metrics in Fig 3 and Table 1 in the paper. Mitsuba2 'aov' integrator was used to output the per-pixel uv coordinates, which were then read from the texture. This also enabled measurements of the RMSE/SSIM of the recovered albedo texture obtained using Mitsuba2 autodifferentiation and gradient descent in image space for the gradient descent section of Table 1. Note that some artifacts from the Mitsuba2 'aov' integrator are apparent near the feet and ears of the bunny, potentially artificially raising the RMSE of the estimated albedos.

With regularization terms $R_i(\theta)$ and constraints, $C(\theta)$, which indicates physically valid parameter values, and $\mathbf{y}$, the pixel data for the captured image. Equation 1 can be solved using stochastic gradient descent. The resulting scene can be used to render a new transport matrix for environment map estimation using the method described in the paper, alternating with the scheme described above until convergence.

5. Alleviating Storage Requirements for Large Linear Models

One potential drawback of the linear systems approach to high frequency illumination estimation is that the size of the model matrix $\mathbf{A}$ can become quite large for modestly sized problems, and this can lead to memory issues. In the main text we largely avoided this issue by limiting ourselves to low resolution measurement images and environment map estimates. However there are more sophisticated strategies that exploit the peculiar nature of the illumination estimation problem to reduce memory requirements.

The unwieldly size of $\mathbf{A}$ is due to the fact that it is effectively a 4-dimensional structure that maps 2D environment maps to 2D measurement images. However, all of the information required to generate $\mathbf{A}$ can be stored in a lower dimensional surface mesh and texture map. This suggests the possibility of more memory-efficient inversion algorithms that could query a rendering engine to obtain necessary matrix elements rather than forming the entire matrix in memory.

As an example, consider the linear system posed in Eq. 6 of the main text. The system can be written more simply as

$$\mathbf{B}\mathbf{x} = (\mathbf{A}^\top \mathbf{A} + \mathbf{R})\mathbf{x} = \mathbf{A}^\top \mathbf{y} = \mathbf{z} \tag{2}$$

We note that $\mathbf{B}$ has dimensions $N \times N$ and $\mathbf{z}$ has dimension N, where N is the size of the environment map vector. These dimensions are independent of the number of pixels (M) in the measurement image. If we could form this sysem directly, without storing $\mathbf{A}$, this would make the use of high resolution measurement images more practical. The elements of $\mathbf{B}$ and $\mathbf{z}$ can be written as

$$B_{ij} = \mathbf{a_i}^\top \mathbf{a_j} + r_{ij} \quad , \quad z_i = \mathbf{a}_i^\top \mathbf{y}. \tag{3}$$

Here r_{ij} denotes the ij^{th} element of R, and $\mathbf{a}_i$ is the i^{th} column of $\mathbf{A}$, which can be generated using a rendering engine. We see that each element of $\mathbf{B}$ can be generated by rendering and taking the inner product of two frames. Because $\mathbf{B}$ is symmetric, N^2 total frames (N^2M total pixels) must be rendered to generate the entire matrix. These frames could be recycled to generate $\mathbf{z}$.

Although this method allows us to avoid storing $\mathbf{A}$ directly, we have traded speed for memory. Although $\mathbf{A}$ may be much larger than $\mathbf{A}^\top\mathbf{A}$ when the measurement images are large, $\mathbf{A}$ can be generated from just N rendered frames (NM pixels). As an intermediate option we might instead consider the case where we are allowed to store blocks of $\mathbf{A}$ with a maximum size of s^2 entries. In this case we can compute the product $\mathbf{A}^\top\mathbf{A}$ using block matrix multiplication. By re-using elements in the pre-rendered block, we can reduce the number of pixels that need to be rendered by approximately a factor of s, such that we only need to render $\frac{1}{2s}N(N-1)M$ pixels total.

References

[1] AliceVision. Meshroom: A 3D reconstruction software., 2018. 4

[2] Michal Jancosek and Tomas Pajdla. Multi-view reconstruction preserving weakly-supported surfaces. In *CVPR 2011*. IEEE, jun 2011. 4

[3] L. Kneip and P. Furgale. Opengv: A unified and generalized approach to real-time calibrated geometric vision. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8, 2014. 1

[4] Laurent Kneip, Hongdong Li, and Yongduek Seo. Upnp: An optimal o(n) solution to the absolute pose problem with universal applicability. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 127–142, Cham, 2014. Springer International Publishing. 1

[5] Pierre Moulon, Pascal Monasse, and Renaud Marlet. Adaptive structure from motion with a contrario model estimation. In *Proceedings of the Asian Computer Vision Conference (ACCV 2012)*, pages 257–270. Springer Berlin Heidelberg, 2012. 4

[6] 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), Dec. 2019. 5

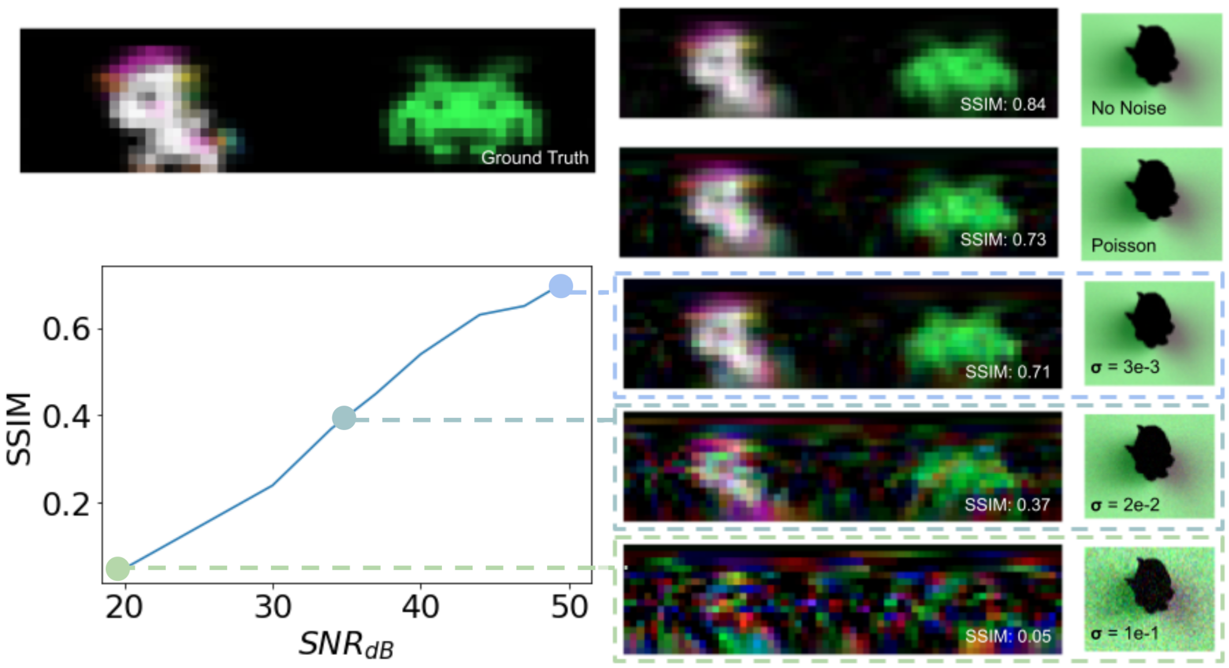

Figure 7. Estimated environment maps given noisy observations and their corresponding structure similarity index measure (SSIM). We synthetically added noise to a rendered image with pixel values between [0,1]. (Right column) Estimated environment maps with no noise, Poisson (i.e. shot noise) only, and Poisson + additive Gaussian noise with increasing standard deviation (σ) and zero mean. (Bottom left) The SSIM plotted against SNR ($20\log_{10}(\frac{1}{\sigma})$) in dB.

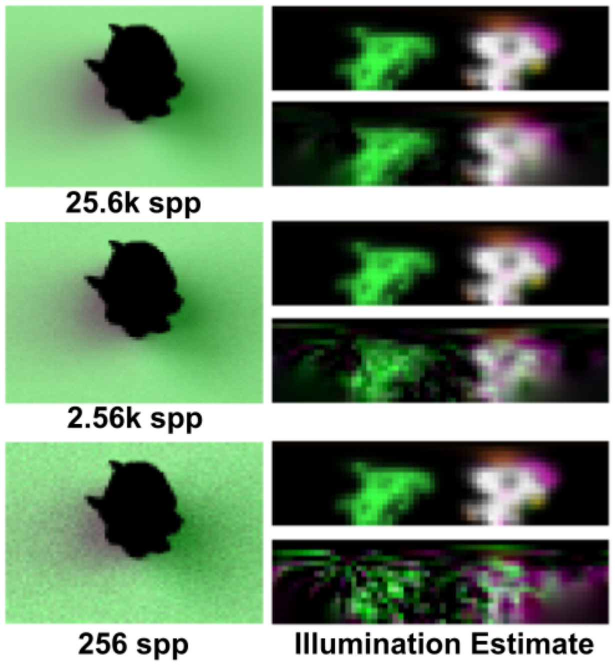

Figure 8. Estimated environment maps compared to ground truth given rendered observed images with decreasing number of samples per pixel (spp). The left column shows the rendered observed images with decreasing number of spp, and the right column shows ground truth and the estimated environment map given the corresponding observed image.

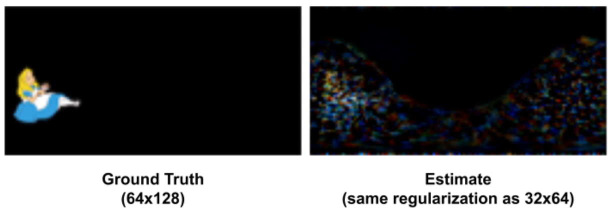

Figure 9. Higher resolution estimates. Illumination estimation using a higher resolution ray transport matrix (environment map 62×128), using the same regularization parameters used in the main text. Reconstructions are not significantly better than the lower resolution case and artifacts are apparent. This suggests that tuning regularization parameters could improve results for higher resolution reconstructions.

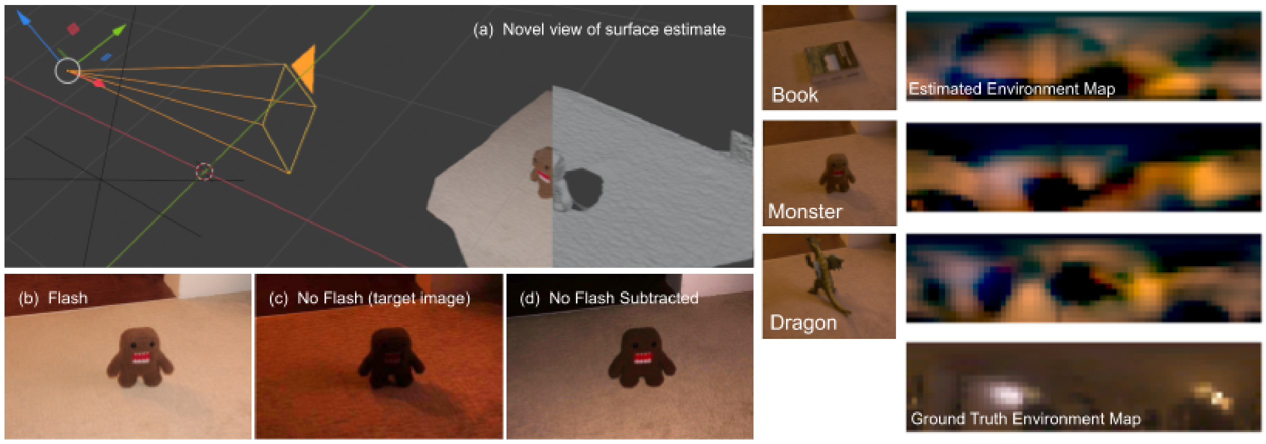
(a) Novel view of surface estimate
(b) Flash
(c) No Flash (target image)
(d) No Flash Subtracted
Book
Monster
Dragon
Estimated Environment Map
Ground Truth Environment Map

Figure 10. Estimation results from meshes obtained using photogrammetry. The left set of panels shows the capture and pre-processing approach: (a) Using photogrammetry, we obtain a camera pose, surface mesh, and diffuse texture. We capture two images (b) a flash image and (c) an image without a flash. (d) We use the flash-only image as input to the texture estimation using photogrammetry. On the right section, we use the approach on the left to recover the 3D geometry and diffuse texture of three objects: a book, monster, and dragon. For each of these objects we generate the corresponding ray transfer matrix and produce an estimated environment map. While the brightest sources can be identified in the estimated environment maps, the estimates appear to suffer from model mismatch, such as the warped recovery due to the larger dragon object and nearby light fixture. Addressing these failure modes is an area for future work.