

Homework Assignment 7

15-463/663/862, Computational Photography, Fall 2020
Carnegie Mellon University

Due Friday, Dec. 18, at 11:59pm

The purpose of this assignment is to familiarize yourselves with the use of physically-accurate rendering as a tool for evaluating computational photography algorithms. Physically-accurate renderers make it possible to simulate the measurements imaging systems would capture of different scenes; these simulated measurements can then be used as input to subsequent computational algorithms. Using simulations makes it easy to perform repeatable experiments for different conditions (e.g., a variety of scenes), quickly evaluate different imaging designs and computational algorithms without the need for lab prototyping, as well as make comparisons with idealized imaging systems that cannot be realized in practice.

To this end, you will use *Mitsuba CLT*, a modified version of the well-known Mitsuba renderer [2], to simulate a few of the imaging systems and 3D reconstruction algorithms we covered towards the end of the class. Compared to the vanilla version of Mitsuba, Mitsuba CLT offers a few capabilities that are particularly useful for computational imaging simulations, including the ability to simulate programmable projectors and light transport probing.

To get full credit on this assignment, you will need to implement the preparatory Part 1, and any *two* of Parts 2-4 you want. Parts 2-4 are independent of each other, and it is entirely up to you to decide which one to implement—pick whichever of the corresponding papers appeals to you the most. If you implement all of Parts 2-4, then the additional part will count as extra homework credit. Additionally, the assignment is intentionally light on instructions, to motivate you to explore the Mitsuba renderer and the extensions available in Mitsuba CLT, as well as the associated papers. For Mitsuba, we strongly encourage you to browse the [Mitsuba documentation](#) (you should keep a copy of this PDF handy while working on this assignment), and the documentation for Mitsuba CLT available in its [GitHub repository](#) and the [author's website](#).

1 Installing and testing Mitsuba CLT (20 points)

For the first part of this assignment, you will need to set up Mitsuba CLT and simulate a few simple scenes. You can download Mitsuba CLT from its public GitHub repository. This is a complex software system, written mostly in C++, which you will need to build from source. To build Mitsuba CLT, you should follow the instructions provided in the Mitsuba documentation.

Alternatively, the Mitsuba CLT repository provides an Amazon Web Services (AWS) AMI that you can use with AWS EC2 instances. This saves yourself the trouble of installing Mitsuba CLT yourself, but it requires spending some time setting up AWS cloud computing. If you opt for this route, we encourage you to use instances of the `c5` type. *Please note that using AWS will cost you money, and the course will not reimburse you for these costs.* If you would rather avoid using AWS, you can run Mitsuba CLT on any laptop with reasonably modern specs (most machines bought in the last 3-4 years should be fine).

The Mitsuba CLT repository includes some sample scenes under the folder `examples/cornellBox`. These scene files make use of most of the plugins you will be using in this assignment, including the additional plugins provided by Mitsuba CLT compared to the original Mitsuba. Use your installation to render example images for each of the sample scenes. Show the output results in your report.

2 Simulating structured-light 3D scanning (40 points)

In this part of the assignment, you will use Mitsuba CLT to implement structured-light 3D scanning. In particular, you should create a scene where a perspective projector and a perspective camera form a rectified pair. The sample scenes that come with Mitsuba CLT show how to use both the projector and camera. As we discussed in class, in a rectified pair, epipolar lines correspond to rows in the projector and camera, and depth can be reconstructed easily by first computing disparity, and then inverting it.

You will implement three different types of structured-light patterns. The first is simple column scanning: for this you will need to render a sequence of images where, for each image, only one column of the projector is illuminated. The second is Gray codes and the third XOR codes; we discussed these briefly in class, and you can read more details about them in the paper by Gupta et al. [1]. Note that the [project website](#) for this paper provides images corresponding to the projector patterns you will need to use, and you are welcome to use them directly.

Use the three types of patterns to virtually 3D-scan some scene of your choice. The scene should include a variety of light transport effects that make structured light be unsuccessful (e.g., specular BRDFs, strong interreflections, subsurface scattering). The paper by Gupta et al. [1] shows many examples that you can use for inspiration. For your selected scene, use Mitsuba CLT's **disparity** and **depth** cameras to produce groundtruth disparity and depth images, and compare them with the disparity and depth images produced from your structured light simulations. In your report, discuss the relative performance of the three structured-light methods, and how they are affected by different light transport effects.

In your submission, make sure to include all generated depth and disparity images. Additionally, for each structured light pattern, include a `.gif` file showing an animation of the scene images for different illumination patterns.

3 Simulating direct-global separation (40 points)

In this part of the assignment, you will use Mitsuba CLT to implement different direct-global separation techniques. To keep things simple, you should again create a scene where a perspective projector and a perspective camera form a rectified pair.

You will now implement three different direct-global separation techniques. The first is the technique introduced by Nayar et al. [3], and in particular, the variant that uses a high-frequency checkerboard. The second is the version of epipolar probing using primal-dual coding introduced by O'Toole et al. [6]. The third is the version of epipolar probing using homogeneous codes introduced by O'Toole et al. [4]. The first technique can be implemented using simply a perspective projector and a perspective camera. The second technique requires using a perspective projector and a *coded* perspective camera. All of these projector and camera types are available as plugins in Mitsuba CLT. The third technique is already implemented by Mitsuba CLT as *row probing*. In implementing these direct-global separation techniques, you should remember that, for a rectified pair, epipolar lines correspond to rows in the projector and camera.

Use the three techniques for direct-global separation to create direct-only and global-only images of some scene of your choice. The scene should include a variety of interesting global illumination effects (e.g., diffuse and specular interreflections, subsurface scattering, volumetric scattering). The paper by Nayar et al. [3] shows many examples that you can use for inspiration. For your selected scene, use Mitsuba CLT's **maxDepth** parameter to produce groundtruth direct-only and global-only images, and compare them with the direct-only and global-only images produced from your simulations of the three techniques. In your report, discuss the relative performance of the three direct-global separation techniques, and how they are affected by different light transport effects.

In your submission, make sure to include all generated direct-only and global-only images. Additionally, for the first two direct-global separation techniques, include a `.gif` file showing an animation of the scene images for different illumination patterns.

4 Simulating optical computing and dual photography (40 points)

In this part of the assignment, you will use Mitsuba CLT to implement two techniques related to light transport matrices, namely, using optical singular value decomposition (SVD) to approximate the light transport matrix of a scene, and then using the approximate light transport matrix to implement dual photography.

Unlike the other two parts, for this one you will likely produce more compelling results if you set up your scene so that they use a perspective projector and perspective camera that do *not* form a rectified pair. You should set up a scene that shows interesting light transport effects (e.g., diffuse and specular interreflections,

subsurface scattering, geometric occlusions). The paper by O’Toole et al. [5] shows many examples that you can use for inspiration.

For your selected scene, implement the optical Arnoldi algorithm described by O’Toole et al. [5] to approximate the light transport matrix of the scene. Note that implementing this algorithm will require exchanging the locations of the camera and projector, to implement both the left-illumination and right-illumination procedures described in the paper. Note that the [project website](#) for this paper provides Matlab code for the Arnoldi iteration; you are welcome to adapt it to Python. Then, use the low-rank light transport matrix approximation you produce to approximate images of the scene under different projector patterns. Compare the approximate relightings with exact ones rendered using Mitsuba CLT. In your report, discuss how successful the approximate relightings are, and what kind of light transport effects they fail to reproduce.

Additionally, for your selected scene, use the low-rank light transport matrix approximation you produce to simulate images of the scene from the viewpoint of the projector, as described in the dual photography paper by Sen et al. [7]. Compare with groundtruth images rendered by exchanging the locations of the camera and projector.

In your submission, make sure to include all generated relightings and dual photography images. Additionally, include a `.gif` file showing an animation of the projector illumination and camera singular-vector images produced during the optical Arnoldi iteration.

Deliverables

As described on the course website, solutions are submitted through Gradescope. We recommend you use one of the methods that use git to maintain your directory structure. Your submission should include the following:

- A PDF report explaining what you did for each problem, including the various visualizations requested throughout Parts 1-4, as well as answers to all questions asked throughout each problem. The report should include any figures and intermediate results that you think may help. Make sure to include explanations of any issues that you may have run into that prevented you from fully solving the assignment, as this will help us determine partial credit.
- All of the `.xml` scene files you used to generate results.

Hints and Information

- Mitsuba 2 is a recently released update of Mitsuba, and the Mitsuba website takes you to this new version by default. You should *not* use Mitsuba 2 for this assignment, as it is intended primarily intended for differentiable rendering, and does not include many of the additional plugins provided by Mitsuba CLT. Instead, you should only use materials (including documentation and example scenes) from Mitsuba 1.
- Most of the algorithms described in this assignment require using radiometrically linear images. Mitsuba can generate such images if you set it to produce output files in OpenEXR format. You can then process the `.exr` files produced as output in the same way you did in homework assignment 2. The sample scene files provided with Mitsuba CLT already output `.exr` results. Note additionally that for the optical computing part, you will need to use radiometrically linear images as input to the projector. You can again use `.exr` files for this.
- If your images appear completely black or very noisy, there are two main parts of a scene file you can debug. The first is the integrator you use. For most parts of this assignment, you can use the BDPT integrator provided by Mitsuba. Note that Mitsuba CLT has some specific requirements for BDPT when it is combined with epipolar probing, as described in the documentation—these are already implemented in the corresponding sample scene files.

The second part is the number of samples used for rendering, controlled by the parameter `sampleCount`. As you are simulating scenes with complex geometry and light transport effects, you will likely need to use a large number of samples (probably at least 1024) to get a result that does not have strong noise.

- For several of the algorithms you are asked to implement for this assignment, you will need to use Mitsuba CLT to render multiple (potentially hundreds) of images of the same scene, under different illuminations. This can quickly become very tedious. Fortunately, Mitsuba CLT makes it possible to make some of the elements used by a scene (e.g., the illumination pattern) parameters, that can be specified when calling Mitsuba from `bash`. This makes it easier to script the various experiments you are asked to simulate through simple `bash` scripts.

Credits

The Mitsuba CLT renderer used for this homework was developed by CMU and 15-463 alumna Jiatian Sun. It is based on the original Mitsuba renderer by Jakob [2].

References

- [1] M. Gupta, A. Agrawal, A. Veeraraghavan, and S. G. Narasimhan. A practical approach to 3d scanning in the presence of interreflections, subsurface scattering and defocus. *International journal of computer vision (IJCV)*, 2013.
- [2] W. Jakob. Mitsuba Renderer, 2010. <http://www.mitsuba-renderer.org>.
- [3] S. K. Nayar, G. Krishnan, M. D. Grossberg, and R. Raskar. Fast separation of direct and global components of a scene using high frequency illumination. *ACM Transactions on Graphics (TOG)*, 2006.
- [4] M. O’Toole, S. Achar, S. G. Narasimhan, and K. N. Kutulakos. Homogeneous codes for energy-efficient illumination and imaging. *ACM Transactions on Graphics (ToG)*, 2015.
- [5] M. O’Toole and K. N. Kutulakos. Optical computing for fast light transport analysis. *ACM Transactions on Graphics (TOG)*, 2010.
- [6] M. O’Toole, J. Mather, and K. N. Kutulakos. 3d shape and indirect appearance by structured light transport. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.
- [7] P. Sen, B. Chen, G. Garg, S. R. Marschner, M. Horowitz, M. Levoy, and H. P. Lensch. Dual photography. *ACM Transactions on Graphics (TOG)*, 2005.