

## Supporting Information for Appendix B

Here I give instructions on how to use the code in the Supporting Information to carry out model tests described in Appendix B.

The project `render_random`, located in the ‘unity projects’ folder, generates the data for these tests. Add this project to the Unity Hub, and open the project. In the version of the project included in the Supporting Information, many files that Unity can reconstruct have been deleted in order to reduce the total size. One consequence is that opening the project for the first time will be slow, and after the project is open, you will have to double-click the `OutdoorsScene` object in the Assets panel in order to make it the active scene.

When the project is open, select the ‘Main Camera’ object in the Hierarchy view, and in the Inspector view there will be GUI elements you can set to configure the test. You can choose whether to test a Lambertian or Unlit material, with or without tonemapping, and you can enter the number of random samples to render and record. After making these settings, run the project, and in the Game view you will see a plane take many random orientations and colors. The random scene parameters and rendered values are recorded in a data file with a name like `data_L1_T0.txt` in the same directory where the `render_random` project is located. The digit after the letter L indicates whether the material is Lambertian (0 = Unlit, 1 = Lambertian), and the digit after T indicates whether the project was run with tonemapping (0 = no, 1 = yes). The data from the first run after opening the project often contains a few rendered values that are outliers, so I suggest making a first run and discarding the data, before generating the data you will use for the tests below.

The first test is implemented in the MATLAB script `model_test_1.m`. Place the data file `data_L1_T0.txt` and the directory ‘tools’ in the same directory as the script, and run the script. This test uses equations (2) and (4) in the main text, without the scale constant  $c$ , to predict the rendered values  $u_k$  from the random lighting and material parameters. Figure 1 shows typical results. The predictions are highly accurate, except that they are off by a scale factor. The script uses linear regression to estimate the scale factor, and reports it in the MATLAB command window.

The second test is implemented in the MATLAB script `model_test_2.m`. Again, place the data files `data_L1_T0.txt` and `data_L0_T0.txt`, and the directory ‘tools’, in the same directory as the script. The script has a boolean variable `testLambertian` that you can set to choose whether to test results from a run of `render_random` with a Lambertian or Unlit material, and a boolean variable `testTonemapping` that you can set to choose whether to test tonemapping. Set `testTonemapping` to False for now; we will test tonemapping in Appendix C. This script plots post-processed values  $v_k$  predicted by equations (2), (3), and (4) in the main text against the actual post-processed values, and also plots the prediction error against the post-processed values. Figure 1 in the main text shows results for a Lambertian material, and Figure 2 in this document shows results for an Unlit material. In both cases, the model’s predictions are highly accurate, and almost always within one or two multiples of  $1/255$  of the correct value.

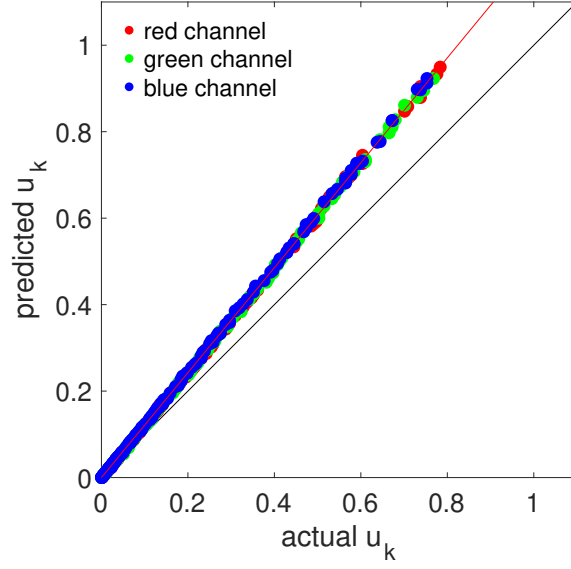


Figure 1: Model predictions for rendered values  $u_k$  from a Lambertian surface with no tonemapping, plotted against the actual rendered values. The solid black line shows  $y = x$ , and the solid red line is the least-squares regression line.

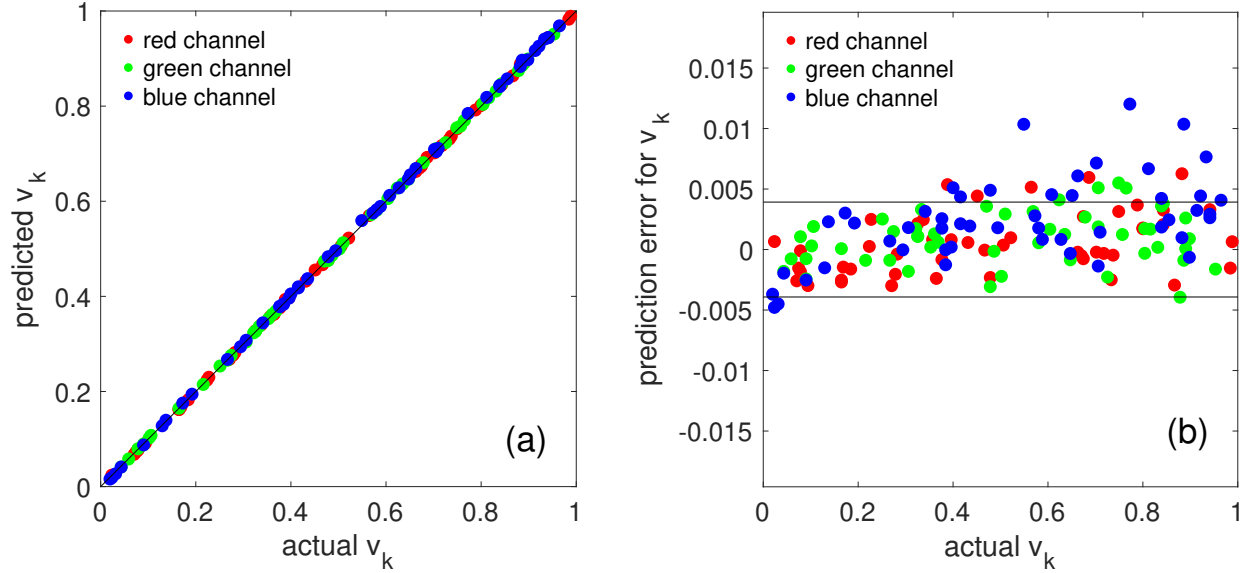


Figure 2: (a) Model predictions for post-processed values  $v_k$  in renderings of an Unlit surface with no tonemapping, plotted against the actual post-processed values. The solid line shows  $y = x$ . (b) Prediction error for post-processed values  $v_k$ , i.e., predicted minus actual value. The solid lines show  $y = \pm 1/255$ .