1 Supporting information: Instructions for replicating analyses

Here I give detailed instructions on how to carry out the tests described in Section 6, Tests of the HDRP model.

1.1 Estimating the scale constant c

The project render_random, located in the 'unity projects' folder of the Supporting Information, generates the data for this test. 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 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 and OutdoorsScene is the active scene, select the 'Main Camera' object in the Hierarchy view, and in the Inspector view there will be GUI elements you can use 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. For this test, check the box for Lambertian material, uncheck the box for tonemapping, and request 1000 samples. 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 named data_L1_T0.txt, which will appear in the 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 test.

The analysis for this test is implemented in the Python script model_test_1.py. Place the data file data_L1_T0.txt and the module hdrp.py in the same directory as the script, and run the script. The script uses equations (2) and (4) in the main text, without the scale constant c, to predict the unprocessed values u_k from the random lighting and material parameters. The script uses linear regression to estimate the scale constant c, which it prints to the console. It also generates Figure 1 in the main text.

1.2 Testing model predictions for Lambertian material without tonemapping

This test is implemented in the script model_test_2.py, and uses the same data file data_L1_T0.txt as in the previous section. Place the data file and hdrp.py in the same directory as the script. The script has a boolean variable testLambertian that indicates whether the data to be tested comes from a run of render_random with a Lambertian material, and another boolean variable testTonemapping that indicates whether the run had tonemapping enabled. Set testLambertian to True, and testTonemapping to False. Run the script. The script plots post-processed values v_k predicted by equations (2) and (4) in the main text against the actual post-processed values, and also plots the prediction error as a function of the post-processed values v_k, the material color m_k, and the directional light color d_k. This plot is Figure 2 in the main text.

In order to make the rendering conditions similar to those in an experiment, the script does not assign a new, random exposure setting to the scene on each sample. However, the exposure can be adjusted by choosing the Global Volume object in

the Hierarchy view, and in the Inspector view entering a value for 'Fixed Exposure' in the Exposure panel. The exposure value is saved to the data file, and the analysis scripts take it into account when predicting rendered values. The results with different exposure values are similar to those with the default value of zero, which was used for the figures in the paper.

1.3 Testing model predictions for unlit material without tonemapping

Run the render_random project again, as in Subsection 1.1, but this time with the box for Lambertian material unchecked. This produces a data file data_L0_T0.txt, which contains results for an unlit material. Run the same script model_test_2.py as in Subsection 1.2, this time with the variable testLambertian set to False. The script plots post-processed values v_k predicted by equations (3) and (4) in the main text against the actual post-processed values, and also plots the prediction error as a function of the post-processed values v_k. This is Figure 3 in the main text.

1.4 Estimating knot point coordinates u_i^* using delta functions

The Unity project render_delta uses a set of 32 cube files, numbered delta_01.cube to delta_32.cube. As explained in detail in the main text, each file delta_m.cube maps the m^{th} knot point in each color channel to one, and all other knot points to zero. The scene in this project contains an achromatic Lambertian test patch under an achromatic directional light source. The project script loads the cube files one at a time, sweeps the light intensity through a wide range of values, and records the resulting post-processed values v_k at each intensity in a text file. A MATLAB script (find_knots.m) loads this data, and for the subset of data from each cube file, finds the rendered value u_k where the post-processed value v_k is at a maximum. For each tonemapping file delta_m.cube, this maximizing value of u_k is taken as an estimate of u_m^*. These estimates are given in the Supporting Information.

[revise; this kind of explanation should go in the main text; here just include instructions on how to run the test]

1.5 Estimating knot point coordinates u_i^* by optimizing model predictions

1.6 Testing model predictions for Lambertian material with tonemapping

Run the render_random project, as in Subsection 1.1, but with the box for Lambertian material checked and the box for tonemapping checked. This produces a data file data_L1_T1.txt. Run the same script model_test_2.py as in Subsection 1.2, with the variables testLambertian and testTonemapping set to True. The script plots post-processed values v_k predicted by equations (2) and (4) in the main text against the actual post-processed values, and also plots the prediction error as a function of the post-processed values v_k. This is Figure X in the main text.

1.7 Testing model predictions for unlit material with tonemapping

Run the render_random project, as in Subsection 1.1, but with the box for Lambertian material unchecked and the box for tonemapping checked. This produces a data file data_L0_T1.txt. Run the same script model_test_2.py as in Subsection 1.2, with

testLambertian set to False and testTonemapping set to True. The script plots post-processed values v_k predicted by equations (3) and (4) in the main text against the actual post-processed values, and also plots the prediction error as a function of the post-processed values v_k. This is Figure X in the main text.

1.8 Testing achromatic gamma correction

1.9 Testing chromatic gamma correction