Supplementary Material: Sensor-realistic Synthetic Data Engine for Multi-frame High Dynamic Range Photography

1. Synthetic Dataset Examples

As introduced in the Dataset section, we used different synthetic datasets to fine-tune network-1 and network-2. For both networks, even though tested on static scenes, we generated corresponding dynamic scenes for exploring potential opportunities such as deghosting. The pure synthetic dataset is publicly available at https://github.com/nadir-zeeshan/sensor-realistic-synthetic-data. The amount of synthetic data will be expanded in future works to cover more types of interesting learning-based mobile imaging applications.

Figure 1(a) shows three examples of the synthetic dataset used to fine-tune network-1. This synthetic dataset is captured at two exposure levels statically and dynamically (i.e., static and dynamic at {EV-3 (low) and EV0 (high)}). Figure 1(b) shows two examples of the synthetic dataset used to fine-tune network-2. This synthetic dataset is captured at three exposure levels statically and dynamically (i.e., static and dynamic at {EV-2 (low), EV0 (mid), and EV2 (high)}).

The human subject in these examples can perform various types of motions, such as nod, walk, run, and waving hands, etc. The static sets are captured first with all motions stopped. Then, all motions are resumed to capture the dynamic sets. Specifically for network-2, we use the static sets to generate the HDR groud-truth, replace the dynamic_mid by the static_mid, and then feed the modified dynamic sets (dynamic_low, static_mid, and dynamic_high) together with the HDR groud-truth to fine-tune the network.

2. Generate Sensor-realistic Synthetic Data

The pure synthetic dataset we provided is a starting point for readers to generate the sensor-realistic synthetic dataset targeted on their mobile devices. The readers are welcome to contact us should they need any clarification. If you would like to control the source, such as the synthetic scenes, the human models and their motions, or the capture loop, please refer to the Unreal Engine documentation for more details. The sensor-realistic synthetic dataset modeled for your targeted device can be generated in the following steps.

1. Model the color space of your targeted camera sensor. After you perform the steps described in the Color Calibration section, the color transition matrix can be calculated by sampling the color differences between the two color checker images (one real and one synthetic).

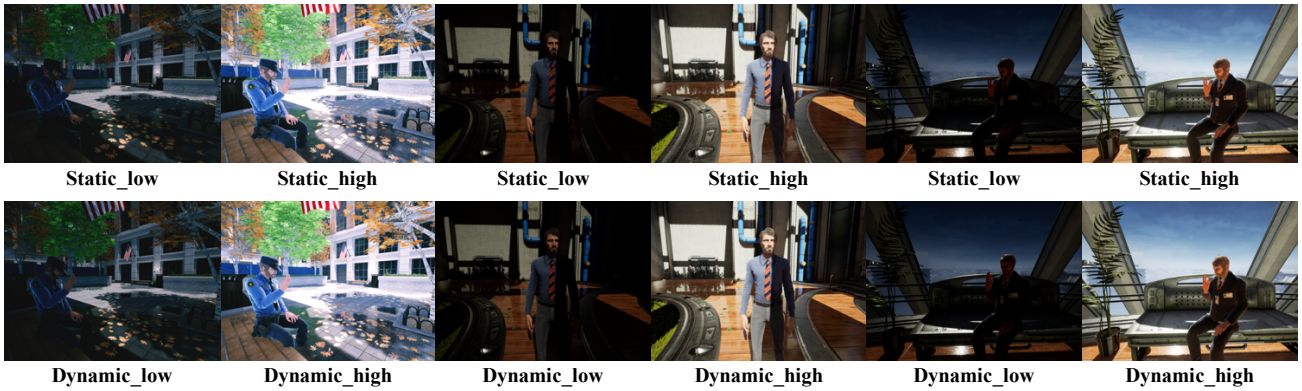
2. Model the noise characteristics of your targeted camera sensor. After you perform the steps described in the Noise Modeling section, the noise distribution parameters for each ISO level can be acquired individually.

3. Apply the modeled sensor realism (color space and noise characteristics). After you performed the previous two steps, you can hard-code the color transition matrix and noise distribution modeled for your targeted device to convert pure synthetic data into sensor-realistic synthetic data (can be easily implemented in Python/MATLAB). Figure 2 shows an example of the synthetic data before and after the sensor realism modeled for a Samsung Galaxy S10 Plus device is applied. We can observe more vivid (warmer) color style in the sensor-realistic synthetic dataset, similar to real captures from the modeled device.

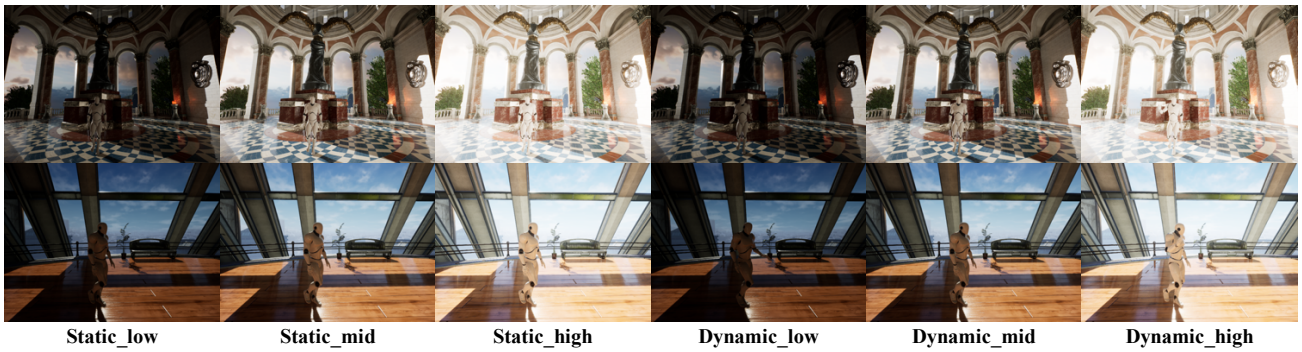
3. Experimental Results

Figure 3 demonstrates more examples of the HDR improvement after network-1 is fine-tuned with our sensor-realistic synthetic dataset. Because Network-1 is already pre-trained with real data captured by the modeled S10 Plus device (the same sensor realism model applied to synthetic data), the HDR improvement coming from the sensor-realistic synthetic data is not very drastic in this network model. However, in these examples, we can still observe better details on the ground between legs, folds on the clothes, and patterns on the transparent window accordingly in each sensor-realistic version, closest to the ground-truth. This means the HDR performance of a network pre-trained on real data can be further improved by the sensor-realistic synthetic data under the premises of correct sensor characteristics modeled and applied.

Figure 4 demonstrates another example of the HDR improvement after network-2 is fine-tuned with our sensor-realistic synthetic dataset. Network-2 is pre-trained on the Kalantari dataset which is captured by a Canon EOS-5D Mark III camera. Thus, the pre-trained network model does not perform well on the real testset captured by the S10 Plus device. However, after fine tuning with the sensor-realistic synthetic data modeled for the S10 Plus device, the HDR output from network-2 has been drastically improved. Particularly in this example, more room details can be observed clearly even in the extremely dark environment in the sensor-realistic version, matching closely with the ground-truth. This means the network correctly learned sensor characteristics specific to the modeled device from the synthetic data.



(a) Three examples in the synthetic dataset used for fine-tuning network-1. The dataset is captured at two exposure levels (EV-3 (low) and EV0 (high)). Motions are small in this dataset to mimic small motions that could happen in between consecutive captures.



(b) Two examples in the synthetic dataset used for fine-tuning network-2. The dataset is captured at three exposure levels (EV-2 (low), EV0 (mid), and EV-2 (high)).

Figure 1: Pure synthetic dataset examples deirectly generated from the Unreal Engine. Synthetic data with various motions and at different exposure levels can be easily generated in Unreal Engine.

(a) Pure synthetic data generated from the Unreal Engine lacks the color characteristics of the modeled S10 plus device.

(b) Sensor-realistic synthetic data after application of sensor realism model learned for S10 plus device.

Figure 2: Examples of pure synthetic data and sensor-realistic synthetic data. After sensor realism applied to pure synthetic data, the color becomes more vivid (warmer), similar to a real photo captured by the modeled S10 Plus device.

Figure 3: Network-1 fine-tuned with our sensor-realistic synthetic dataset (sensor-realistic) outputs more details (closest to the ground-truth) for the ground between legs, folds on clothes, and patterns on transparent windows than the outputs from the network fine-tuned with pure synthetic data (pure-synthetic) and the network pre-trained with real data only (pre-trained). Note that the improvement coming from the sensor-realistic synthetic data is not very drastic in Network-1 because this network is already pre-trained with real data captured by the targeted S10 Plus device (the same sensor realism model applied to synthetic data).

Figure 4: Network-2 fine-tuned with our sensor-realistic synthetic dataset (sensor-realistic) outputs more details even in the extremely dark environment, while the pre-trained network (pre-trained) fails. Images are equally enhanced by Photomatix for visualization.