Supplementary Material
OpenRooms: An Open Framework for Photorealistic Indoor Scene Datasets

Zhengqin Li[1] Ting-Wei Yu[1] Shen Sang[1] Sarah Wang[1] Meng Song[1] Yuhan Liu[1] Yu-Ying Yeh[1]
Rui Zhu[1] Nitesh Gundavarapu[1] Jia Shi[1] Sai Bi[1] Hong-Xing Yu[1] Zexiang Xu[2]
Kalyan Sunkavalli[2] Miloš Hašan[2] Ravi Ramamoorthi[1] Manmohan Chandraker[1]

[1]UC San Diego [2]Adobe Research

Our supplementary material includes an accompanying video, further details of the dataset and tools, as well as further results and comparisons on several tasks and applications. This supplementary document includes the following:

- Explanation of the accompanying video (Sec. S1)
- Demonstrations of editing applications (Sec. S2)
- Further results on inverse rendering (Sec. S3)
- Ground truth for friction coefficients (Sec. S4)
- Demonstrations to motivate robotic tasks (Sec. S5)
- Results on semantic and instance segmentation (Sec. S6)
- Multi-task estimation and domain adaptation (Sec. S6)
- Photorealistic ground truth for SUN-RGBD (Sec. S7)
- Details of the microfacet BRDF model (Sec. S8)
- Details of the lighting ground truth (Sec. S9)
- Details of physically-based GPU renderer (Sec. S10).

S1. Video

The accompanying video is included at the following link. It illustrates the following capabilities enabled by the proposed OpenRooms framework:

- Creating photorealistic synthetic versions of real acquired scans, with side-by-side comparisons
- Beyond shape, extensive ground truth for high-quality spatially-varying material and spatially-varying lighting with various elements of complex light transport
- Ground truth for semantic and instance segmentation
- Ground truth for friction coefficients
- Inverse rendering and scene understanding applications
- Image editing applications for augmented reality
- Motivation for robotics applications such as navigation, pushing and sim-to-real transfer studies, where variations in material and lighting may be important.

The video also illustrates various steps of the proposed dataset creation framework, where besides the images and ground truth, the involved tools are being publicly released as part of our open framework.

Barron13 [2]	Gardner17 [5]	Garon19 [6]	Li20 [8]	Ours
11.5%	28.07%	29.15%	34.84%	38.89%

Table S1: User study on object insertion by comparing to the ground-truth. Here we compare the lighting prediction results of different methods against ground truth lighting and report the % of times that users picked a particular method as being more realistic than ground truth; ideal performance is 50%. Our result is marked in (blue). Similar to the results in the main paper, our trained network outperforms previous state-of-the-art ones.

	[2]	[5]	[6]	[8]	Ours
Barron13 [2]	-	23.37%	13.25%	13.60%	11.81%
Gardner17 [5]	76.63%	-	36.25%	39.54%	33.84%
Garon19 [6]	86.75%	63.75%	-	42.28%	43.47%
Li20 [8]	86.40%	60.46%	57.72%	-	45.23%
Ours	88.19%	66.16%	56.53%	54.77%	-

Table S2: User study on object insertion with pairwise comparisons. $X\%$ in row I column J means that in $X\%$ of total cases, human annotators think method I outperforms method J. Comparisons with our method are labeled in (blue). We observe improvements over all prior methods.

S2. Applications: Photorealistic Image Editing

User study: Object insertion A user study was conducted to quantitatively evaluate object insertion performance using the inverse rendering network of the main paper. The network is trained on the proposed dataset and evaluated on the real dataset of [6], which provides 20 images with measured ground truth for spatially-varying lighting. Some qualitative and quantitative results have been included in Table 6 and Figure 13 of the main paper. We now provide more comparisons in Table S1, Table S2 and Figure S1.

In Table S1, we summarize comparisons for different methods against ground-truth lighting. Ideal performance for this task is 50%, which indicates that the predicted lighting and the ground-truth lighting are indistinguishable. The best two previous methods of [8] (34.84%) and [6] (29.15%) are trained on SUNCG-related datasets, while our method (38.89%) outperforms both of them. In Table S2, we show complete pairwise comparisons for object insertion among recent state-of-the-art lighting prediction methods. This is

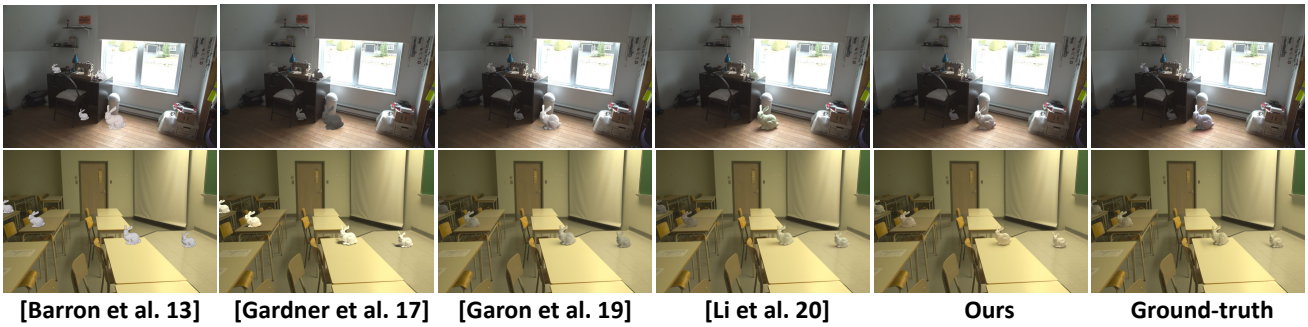

Figure S1: Qualitative comparisons of object insertion on real images from the dataset of [6].

Figure S2: Material replacement on a real image using the inverse rendering predictions from a network trained on our dataset.

a more detailed version of Table 6 in the main paper and reaffirms that a network trained on the proposed dataset achieves the best performances. In Figure S1, we show more qualitative comparisons. The network trained on our dataset achieves realistic high frequency shading and consistent lighting color.

In conclusion, the dataset created by our framework enables high-quality object insertion with performance better than methods built on previous datasets.

Qualitative results: material editing We show material replacement examples on real images in Figure 14 in the main paper and Figure S2 here. Since we use a per-pixel environment map to represent spatially-varying lighting, we can recover complex spatially-varying highlights when we replace the original material with another glossy material.

S3. Inverse Rendering Trained on OpenRooms

This section includes: **(a)** further results on light source detection, **(b)** quantitative results on per-pixel lighting estimation, **(c)** comparisons of normal estimation with prior works on real datasets, **(d)** comparisons for layout estimation, **(e)** ablation study for the network on our proposed dataset, **(f)** qualitative visualization of inverse rendering network outputs on synthetic and real data, when trained on a synthetic dataset created from ScanNet using the proposed dataset creation method.

	$A(10^{-3})$	$N(10^{-2})$	$D(10^{-2})$	$R(10^{-2})$	L
Cascade0	9.99	4.51	5.18	6.59	0.150
Cascade1	9.43	4.42	4.89	6.64	**0.146**
Bilateral solver	**9.29**	-	**4.86**	**6.57**	-

Table S3: Ablation study for the network architecture on our proposed dataset. We report the scale invariant L2 loss for albedo (A), L2 loss for normal (N), scale invariant log L2 loss for depth (D), L2 loss for roughness (R) and scale invariant $\log(x+1)$ L2 loss for per-pixel lighting (L). We observe both cascade structure and bilateral solver can improve the prediction accuracy.

Figure S3: Further results of light source detection on NYUv2 test images. The top row is with pre-training on OpenRooms and the bottom row without the pre-training.

Inverse rendering on test set of proposed dataset Table S3 quantitatively evaluates the performance of the network trained and then tested on the proposed synthetic dataset created from ScanNet. We observe that both the cascade structure and bilateral solver can improve the accuracy of prediction of most intrinsic components. Figure S4 shows a few inverse rendering results on our synthetic testing set. From the figure, we observe that through iterative refinement, the cascade structure can effectively remove noise and recover high-frequency signals, especially for lighting and normal prediction. The bilateral solver also helps remove noise by enhancing the smoothness prior.

Further examples of inverse rendering on real images Figure S5 shows inverse rendering results on several real

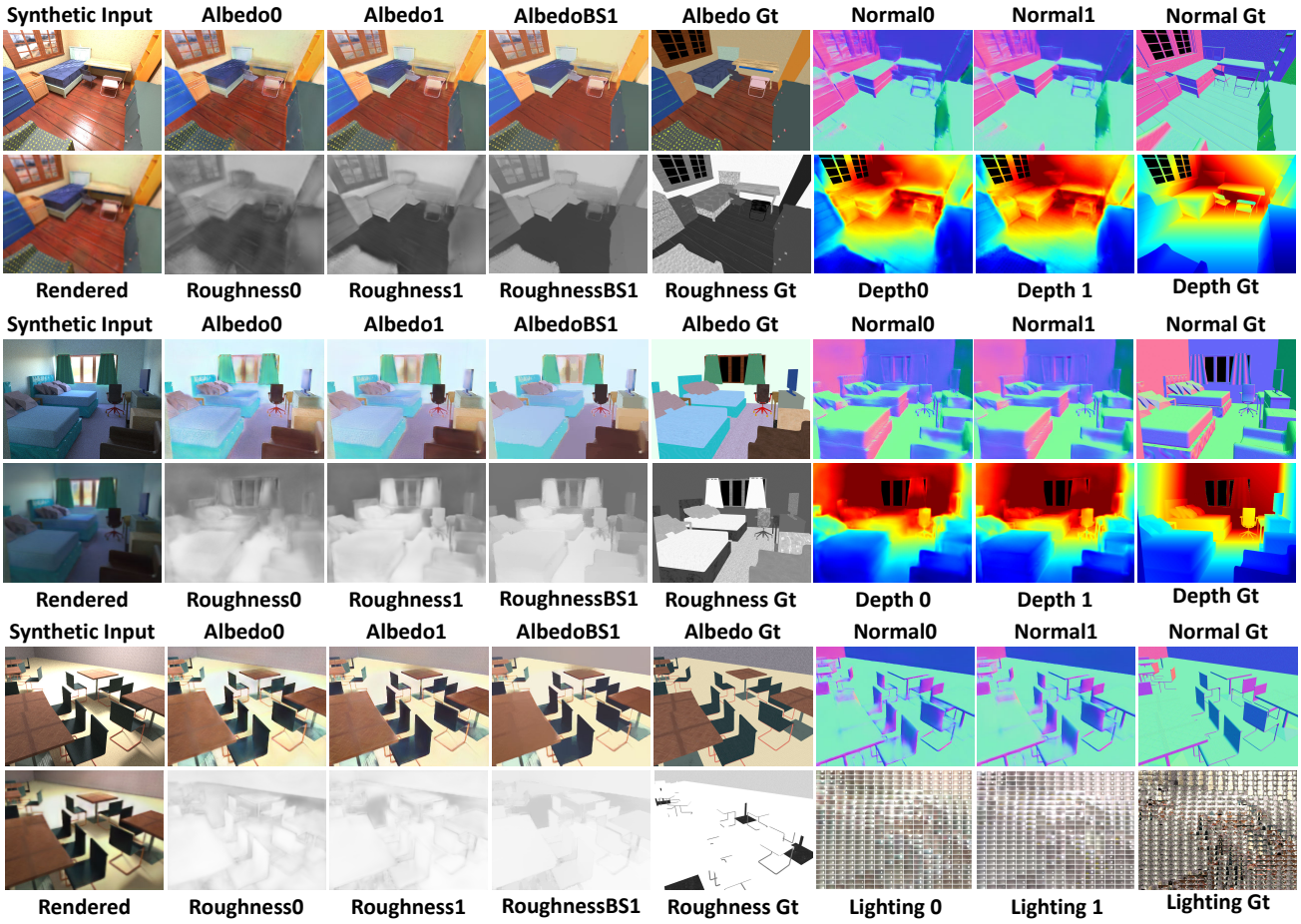

Figure S4: Qualitative visualization of inverse rendering results on synthetic images from the test set of the proposed dataset.

images. We observe that even though the network is trained on a synthetic dataset, it can generalize well to real data. For real data, the effectiveness of the cascade structure and bilateral solver is more apparent, probably due to noisier initial predictions on real data.

Light source detection We include further examples for light source detection in Figure S3, besides the quantitative numbers in Table 4 and visualizations in Figure 10 of the main paper. We again observe that pre-training on OpenRooms is beneficial for detecting both wondows and lamps.

Per-pixel lighting estimation We have shown per-pixel lighting prediction results on both real and synthetic examples in Figure 9 of the main paper. We now provide further qualitative results on real and synthetic data in Figure S5 and Figure S4, respectively. Table S3 shows quantitative numbers on our OpenRooms validation set. We observe that our per-pixel lighting prediction is consistent with spatially-varying intensity and the ground-truth light source position. Both quantitative and qualitative comparisons show that cascade structure can improve the per-pixel lighting prediction by making the prediction sharper and less noisy.

Normal prediction Figure S6 shows qualitative comparisons with [9] and [12] on three real examples from [9]. We observe that even though [12] achieves the best accuracy on NYU dataset, it might overfit to that specific dataset and might not generalize well to images from other sources. On the contrary, both [9] and our network achieve less noisy normal predictions. Our network may sometimes over-smooth the normal, probably since our scenes are built from Scan2CAD annotations that usually contain only a small number of large items of furniture in each room. Therefore, there may be less geometric detail in our synthetic dataset. This can probably be solved in the future by procedurally adding small objects to the rooms to increase the complexity of the dataset.

Layout prediction For future ease of annotation, we add an automatic layout predictor using Floor-SP [3] to the OpenRooms tools. It accepts a 2D top-down projection of the point cloud and its mean surface normal as inputs. In the subsequent steps, the room segmentation is predicted and

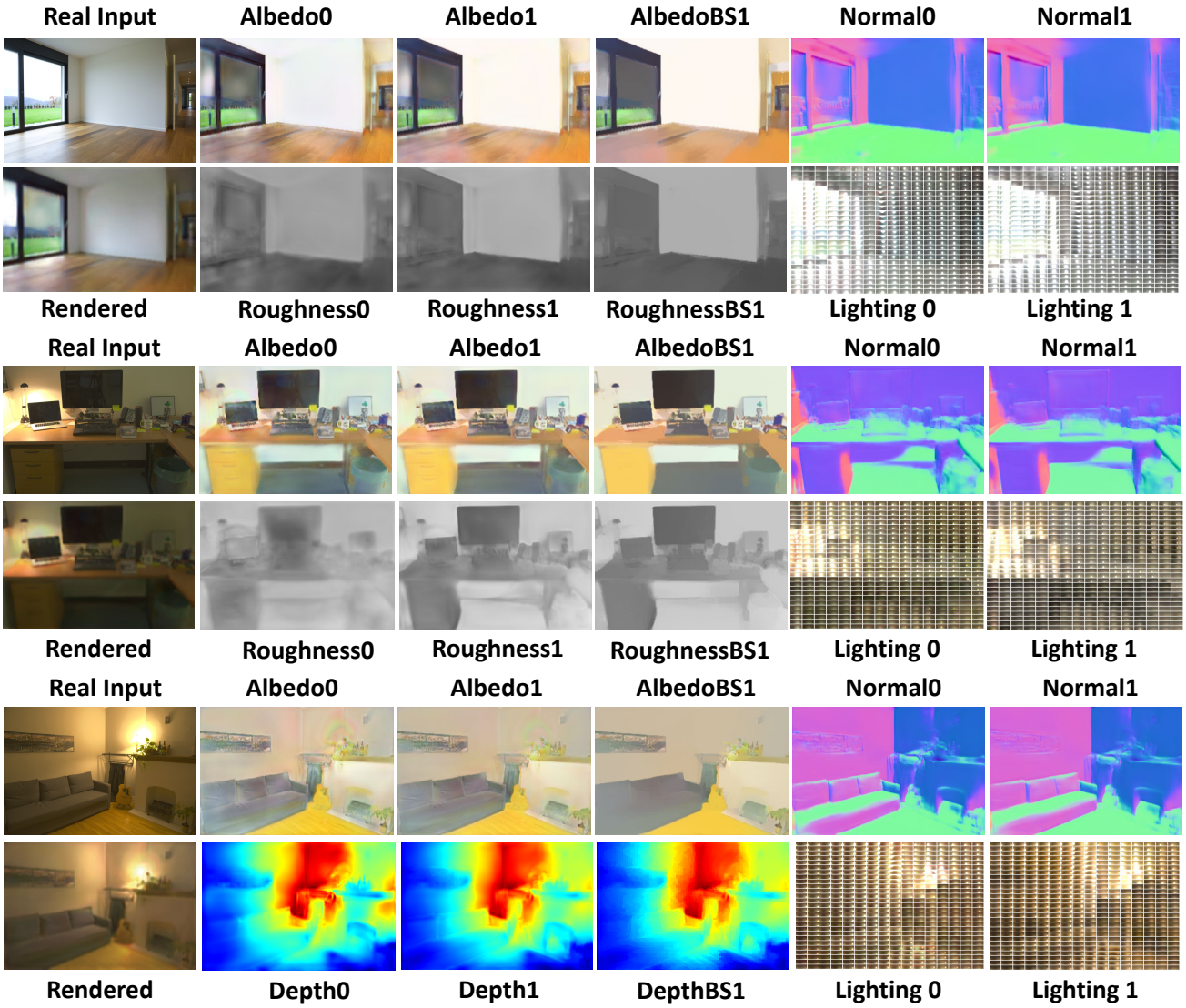

Figure S5: Qualitative visualization of inverse rendering results on real images, using a network trained on synthetic photorealistic images from the proposed OpenRooms dataset (created based on scans from ScanNet).

	Corner		Edge		Room
	Precision	Recall	Precision	Recall	IOU
Chen19 [3]	0.358	0.524	0.151	0.191	0.734
Trained on ScanNet	0.531	0.716	0.254	0.316	0.858

Table S4: Comparison of Floor-SP [3] models with pre-trained weights provided by [3] and weights trained on 1069 ScanNet scenes. The network trained on our newly labeled scenes perform significantly better on noisy scanned point cloud.

room loops are formed (we omit the loop merging step since ScanNet scans generally contain a single room). We refer the reader to [3] for more details. Since the point cloud generated by RGB-D scans contain higher levels of noise compared to the training data used by Floor-SP, we trained a randomly initialized model on a subset of ScanNet consisting of 1069 scenes with human-annotated layout as ground-truth.

The final layout is evaluated on 103 held-out scenes in terms of corner precision and recall, edge precision and recall, as well as intersection-over-union of the room segmentation. A corner prediction is deemed correct if its distance to the closest ground truth corner is within 10 pixels. An edge prediction is deemed valid if its two endpoints pass the criterion for corners and the edge belongs to the set of ground-truth edges.

Table S4 shows the comparison between the model trained on ScanNet and the pre-trained weights provided by the original implementation of Floor-SP. Figure S7 shows a qualitative comparison. Note that the room segmentation

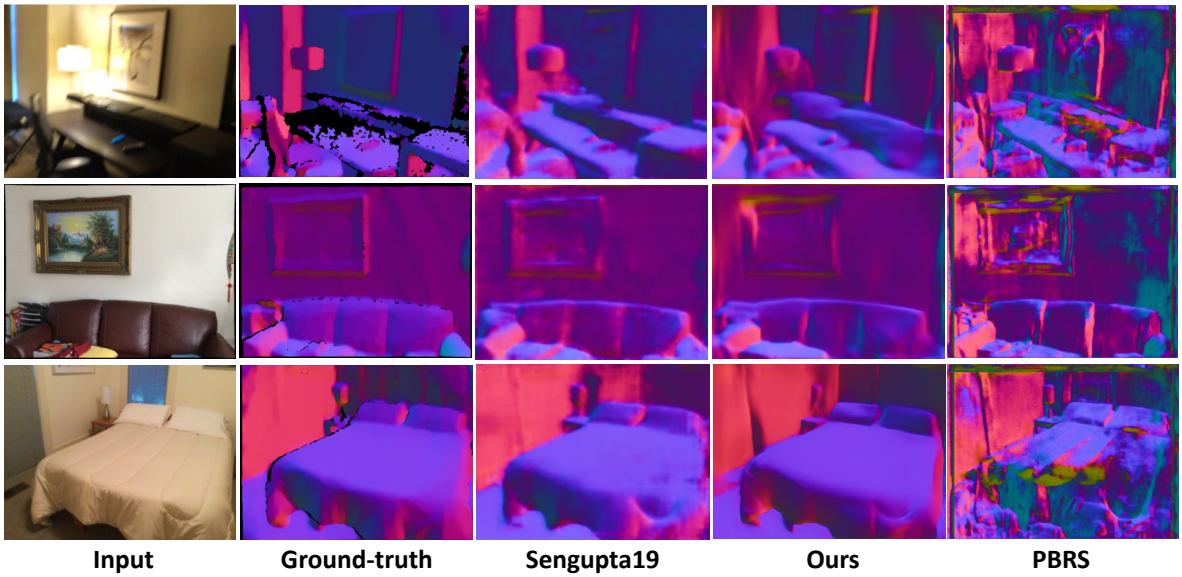

Figure S6: Qualitative comparisons of our normal estimation with Sengputa19 [9] and PBRS [12], on real images from [9].

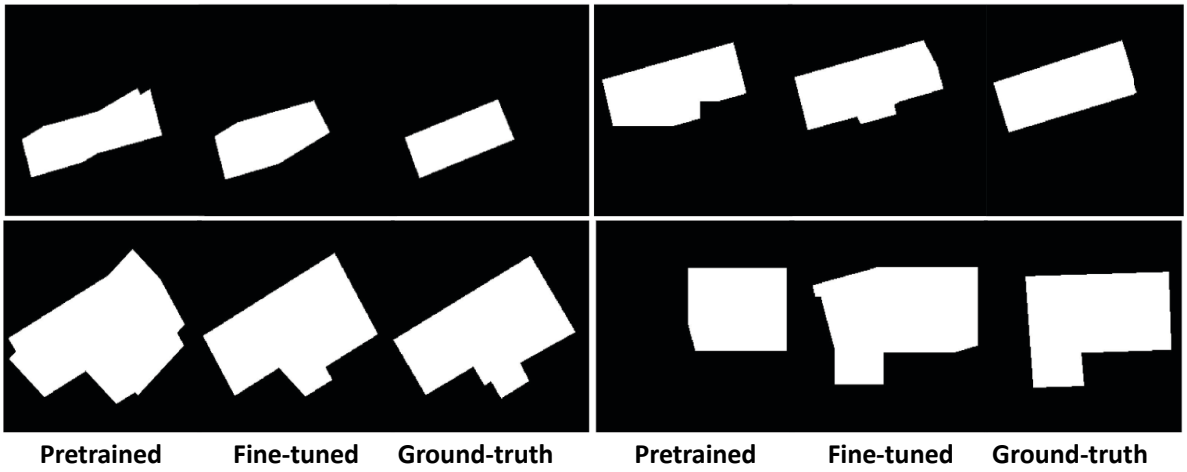

Figure S7: Comparison of layout reconstruction using the original network from [3] and the network trained on our ScanNet annotation.

performs moderately well despite the low precision and recall of the corners and edges. We believe that this is caused by the ambiguities during layout annotation. Since we require the walls to be arranged such that they form a closed loop, for the scans that do not cover the entire room, the human annotator would have to add false corners and edges that pass through open areas where the scan is incomplete, thereby affecting the evaluation of the corner and edge predictions. On the other hand, false corners and edges do not affect IoU since it measures the area covered by the room, rather than the occurrence of predictions.

S4. Ground Truth for Friction Coefficients

In this section, we describe in detail the process for assigning per-pixel ground truth for friction coefficients in OpenRooms scenes mentioned in Sec 4.3 of the main paper. We also show several qualitative examples for assigned friction coefficients. Since OpenRooms provides complete control over mapping arbitrary semantically-meaningful materials to indoor surfaces, such ground truth may enable future studies in inferring physical properties from images, or learning robotics tasks conditioned on material properties.

Reflectance disks We follow the concept of reflectance disks from Zhang et al. [11] for predicting friction coefficients for various materials. The acquisition setup of

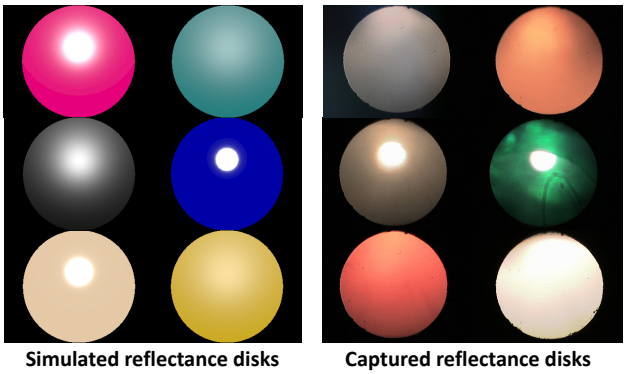

Figure S8: Comparisons of randomly sampled reflectance disks captured by the system of Zhang et al. [11] (left) and rendered by our virtual environment (right). We observe the distributions of highlights and spatially-varying intensities to be similar.

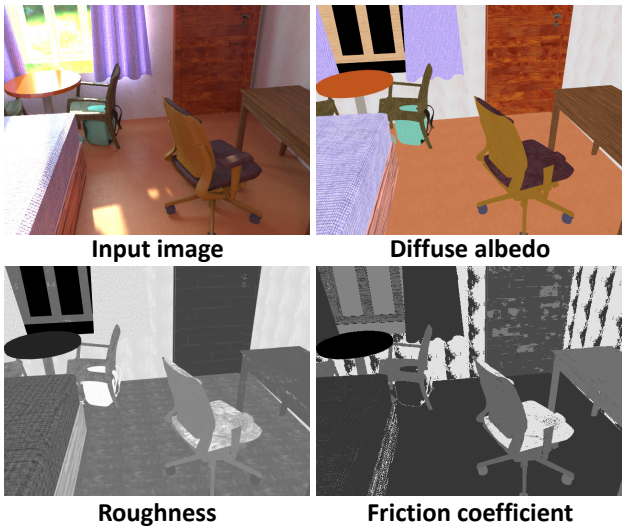

Figure S9: Visualization of friction coefficient in OpenRooms dataset. We map diffuse albedo and roughness parameters to friction coefficient based on nearest neighbor search. We observe that specular materials usually have smaller friction coefficients.

[11] includes a beam splitter, an orthographic camera and a parabolic mirror, to capture material appearances by densely sampling from a large range of view directions and a small range of lighting directions (please see Figure 3 of [11]). We mimic this capture system to render the reflectance disk using our physically-based renderer. We uniformly sample the parameter space of our microfacet BRDF model and render a reflectance disk for each sampled point. Figure S8 compares the reflectance disks rendered under our virtual environment and captured by the system. We observe that the distribution of specular highlights and intensities of the two sets of reflectance disks can match well.

Deep reflectance codes After obtaining the reflectance disk, Zhang et al. [11] use a pretrained deep network to map the reflectance disk to a low dimensional latent space, which is termed a deep reflectance code. Due to the dense down-sampling operations, the deep reflectance code is robust to translation and rotation, which makes it a suitable representation for modeling intrinsic properties of materials, including the friction properties. Thereafter, they use K-nearest neighbor method to map deep reflectance code to friction coefficients. Following their implementation, we also map our reflectance disks to a deep reflectance code, to the friction coefficients using nearest neighbor search for each of our sampled microfacet BRDF parameters. This gives us a table that allows us to map our microfacet BRDF parameters to friction coefficients through bilinear interpolation or nearest neighbor search. Figure 16 in the main paper and Figure S9 in the supplementary show some examples of our friction coefficient predictions. We observe that specular materials are more likely to have small coefficients of friction, which is consistent with physical intuition.

S5. Applications: Robotics, Embodied Vision

In this section, we provide details and examples for the following: **(a)** integration of OpenRooms scenes with PyBullet for physical simulation, **(b)** qualitative results of OpenRooms scenes and capabilities enabled by such integration, **(c)** demonstration of navigation in OpenRooms scenes, **(d)** demonstration of pushing tasks with different coefficients of friction.

Integrating OpenRooms with PyBullet To transform a static OpenRooms scene to an interactive environment, we treat each object in the scene as a single link robot and equip it with a URDF to describe its physical and visual properties. In our dataset, the object's 3D mesh and the associated MTL file are recorded in an OBJ file. Given this OBJ file, we generate another OBJ file by convex decomposition. The URDF links these two OBJ files, using the first one for rendering and the other for collision detection. From the albedo and roughness images provided in the MTL file, we can estimate the object's friction coefficient. Other physical properties, such as the mass, center of mass and inertial matrix can also be provided in the URDF or set in the physics engine later. Having URDFs for each object in the scene, we then load them along with the robot's URDF into the physics engine (for example, PyBullet) to allow full interactive physics simulations.

Qualitative examples All kinds of OpenRooms scenarios can be integrated with the physics engine (Pybullet in our case) to create interactive environments where a robot can act (for example, navigate or push objects). The objects can be rearranged, the light sources replaced and the materials changed, boosting the variety of the scenes and motivating

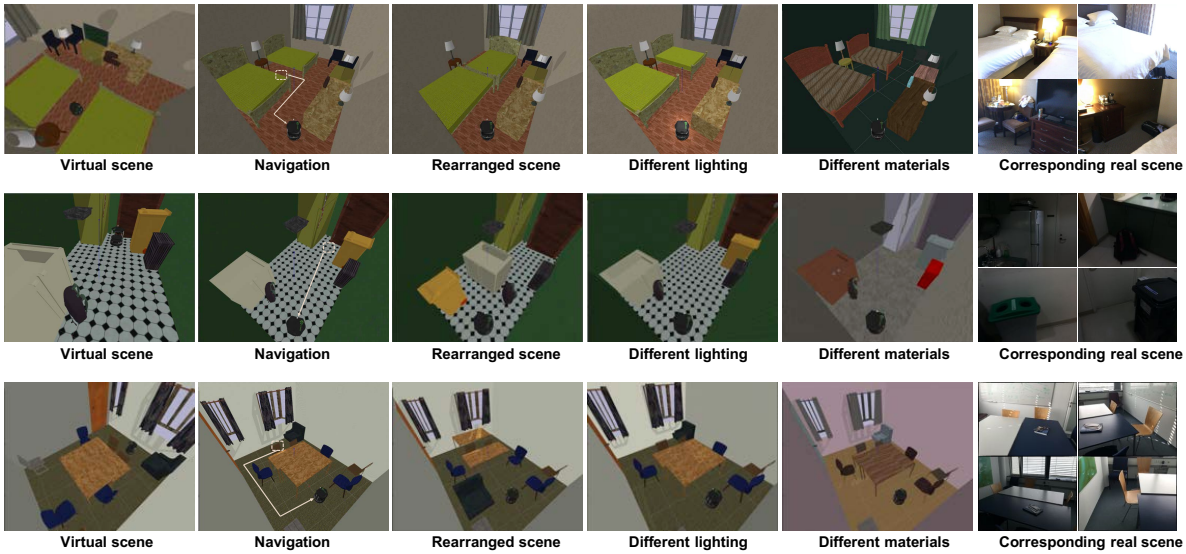

Figure S10: More examples (bedroom, kitchen, and conference room) of OpenRooms scenarios integrated with a physics engine under different settings, as well as the images from the corresponding real scenes.

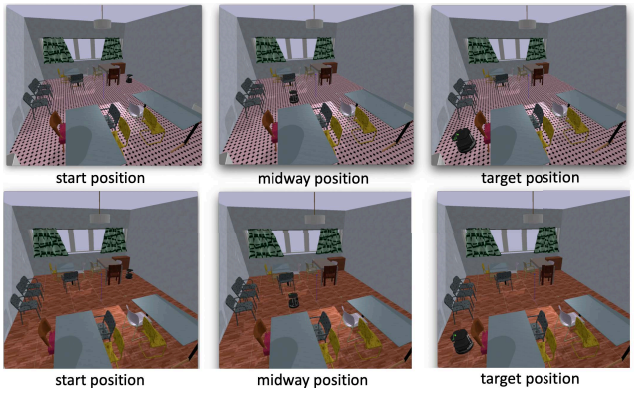

Figure S11: A Turtlebot navigating in a classroom from the brown cabinet to the pink chair, on floors with different materials.

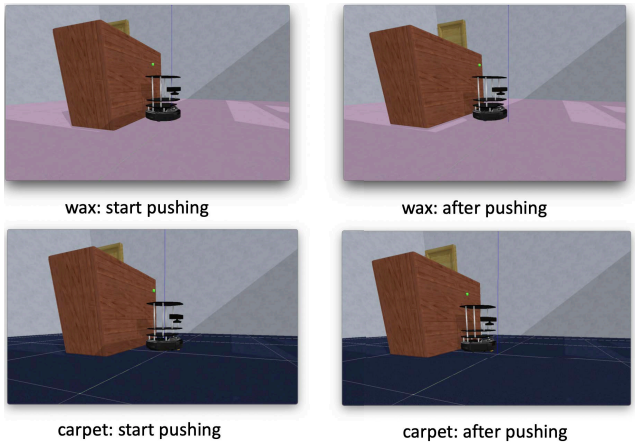

Figure S12: A Turtlebot pushing a wooden cabinet on floors with materials of different friction coefficients: wax and carpet, leading to differing physical outcomes (also see the accompanying video).

studies on effect on robotic tasks when such scene properties are varied. Further, our dataset and tools allow a correspondence between the real scenes used for creating the dataset and the rendered synthetic scenes, which motivates their use to create testbeds for studies in sim-to-real transfer. Similar to Figure 15 of the main paper, we show several examples of such capabilities enabled by OpenRooms in Figure S10.

Navigation We provide a simple example to show the support for navigation tasks. In this example, a two-wheeled Turtlebot is asked to navigate in an indoor room, from a starting location to a target location. The agent has a three-dimensional state space S and a four-dimensional continuous action space A. The state $s \in S$ is the agent's 3D position. The first two dimensions of the action space A correspond to moving forward or backward for an non-negative distance d. The other two dimensions represent turning left or right for an angle within $[-\pi, \pi]$. The robot is given a sequence of actions $\{a_1, a_2, \ldots, a_T\}$ to accomplish the navigation task. Figure S11 shows a few frames from the resulting video. Besides directly working on the robot's 3D positions, we also provide a variety of observation modes, RGB images, surface normals, depth images, semantic segmentations and joint level state space for studying the navigation problems from different perspectives. In particular, we note that OpenRooms may allow navigation studies under different material properties and lighting conditions.

Pushing with different frictions We conduct pushing experiments to show how the friction coefficients associated with different materials impact the object's behavior given the same pushing force applied by a robot. With the experiment setup (Table S5), a cabinet is placed at position P_0^C and

Experimental setup	
Gravity (m/s^2)	(0,0,-9.8)
Robot mass (kg)	0.27
Cabinet mass (kg)	2.00
Robot initial position P_0^R (m)	(0.5,0,0)
Cabinet initial position P_0^C (m)	(0,0,0)
Force exerting time t^F (s)	0.08
Observing time t^O (s)	1.67
Robot moving speed v^R (m/s)	(2.4,0,0)
Robot moving distance (m)	0.20

Table S5: Experimental setup for the pushing tasks.

object	material	friction coefficient
cabinet	wood	0.76
floor 1	carpet	0.76
floor 2	wax	0.31

Table S6: Physical properties of the objects involved in the pushing tasks. The friction coefficients are in the range [0,1].

scenario	object position offset (meter)
Pushing on floor 1	0.12
Pushing on floor 2	0.23

Table S7: Comparison of results on the pushing tasks with different friction coefficients for the floor.

a Turtlebot is initialized at position P_0^R in the world frame. To generate a horizontal pushing force to the cabinet, the robot moves towards it at a constant speed v^R for time t^F. Then the robot keeps still for time t^O and during this period of time, the cabinet will eventually stop due to the friction between the cabinet and the floor.

We perform this pushing task with two different floor materials while keeping the other conditions the same. Table S6 summarizes the physical properties of the objects appearing in the two scenarios. Table S7 compares the cabinet's position offset with respect to its starting position after being pushed by the robot on the carpet and on the wax floor, respectively. Figure S12 shows initial and final snapshots from the simulation videos of the pushing tasks.

Given the floor's material information, we compute the per-pixel friction coefficient map from the albedo and the roughness images, according to the method in Section 4.3 of the main paper and Section S4 of the supplementary. The average of the resulting map is then fed into the physics engine as the floor's friction coefficient. Note that although not yet supported by popular physics simulators such as PyBullet, our dataset provides ground truth for spatially-varying friction coefficients, which can be incorporated into higher quality simulators in the future.

	bbox	seg
AP(0.5:0.95)	39.1	48.464
AP-cabinet	34.00	52.97
AP-bed	56.63	66.33
AP-chair	43.89	48.10
AP-sofa	51.19	61.29
AP-table	43.86	53.27
AP-door	63.52	75.74
AP-window	42.81	65.53
AP-bookshelf	46.07	53.24
AP-counter	6.92	7.50
AP-desk	13.39	23.39
AP-curtain	41.27	35.05
AP-bathtub	58.55	62.98
AP-bag	16.52	52.02
AP-otherstructure	1.01	1.19
AP-otherfurniture	60.52	67.78
AP-otherprop	51.81	59.83

Table S8: Instance segmentation results on the OpenRooms dataset using the label space of NYU.

S6. Segmentation, Multi-Tasking, Adaptation

In this section, we provide: **(a)** further qualitative results for semantic segmentation, **(b)** results for instance segmentation, **(c)** quantitative and qualitative results for multi-task shape, material and semantics estimation, **(d)** domain adaptation for depth estimation.

Semantic segmentation The main paper provides quantitative numbers and some visualizations for semantic segmentation networks trained on OpenRooms data. More qualitative results for semantic segmentation using PSPNet(50) and DeepLabV3 evaluated on OpenRooms test set and the NYUv2 test set are in Figures S13 and S14, respectively.

Instance segmentation The results of instance segmentation on OpenRooms, using the same network architecture as light source detection, are shown in Table S8 for the categories that overlap with the NYU label space. A few qualitative examples are shown in Figure S15, indicating that the proposed dataset may also be useful for studies in instance segmentation.

Multi-task learning We quantitatively evaluate our multi-task model (Cascade0) with estimation of albedo, normal, depth, roughness as well as semantic segmentation on the test set of OpenRooms, reporting the results in Table S9. As compared to the original model which does not include a segmentation branch, our multi-task model provide competitive results to albedo, normal, depth and roughness estimation while enable additional semantics estimation with reasonable performance. Besides the qualitative results in Figure 12 of the main paper, we provide further qualitative results of the multi-task model in Figure S16 for OpenRooms test images and Figure S17 for real images.

Domain Adaptation for depth estimation The availability of large-scale synthetic data with labels also allow studies where labels from OpenRooms may be used for domain adaptation to real scenes where labels might not be available. In this section, we show an example for depth estimation, where

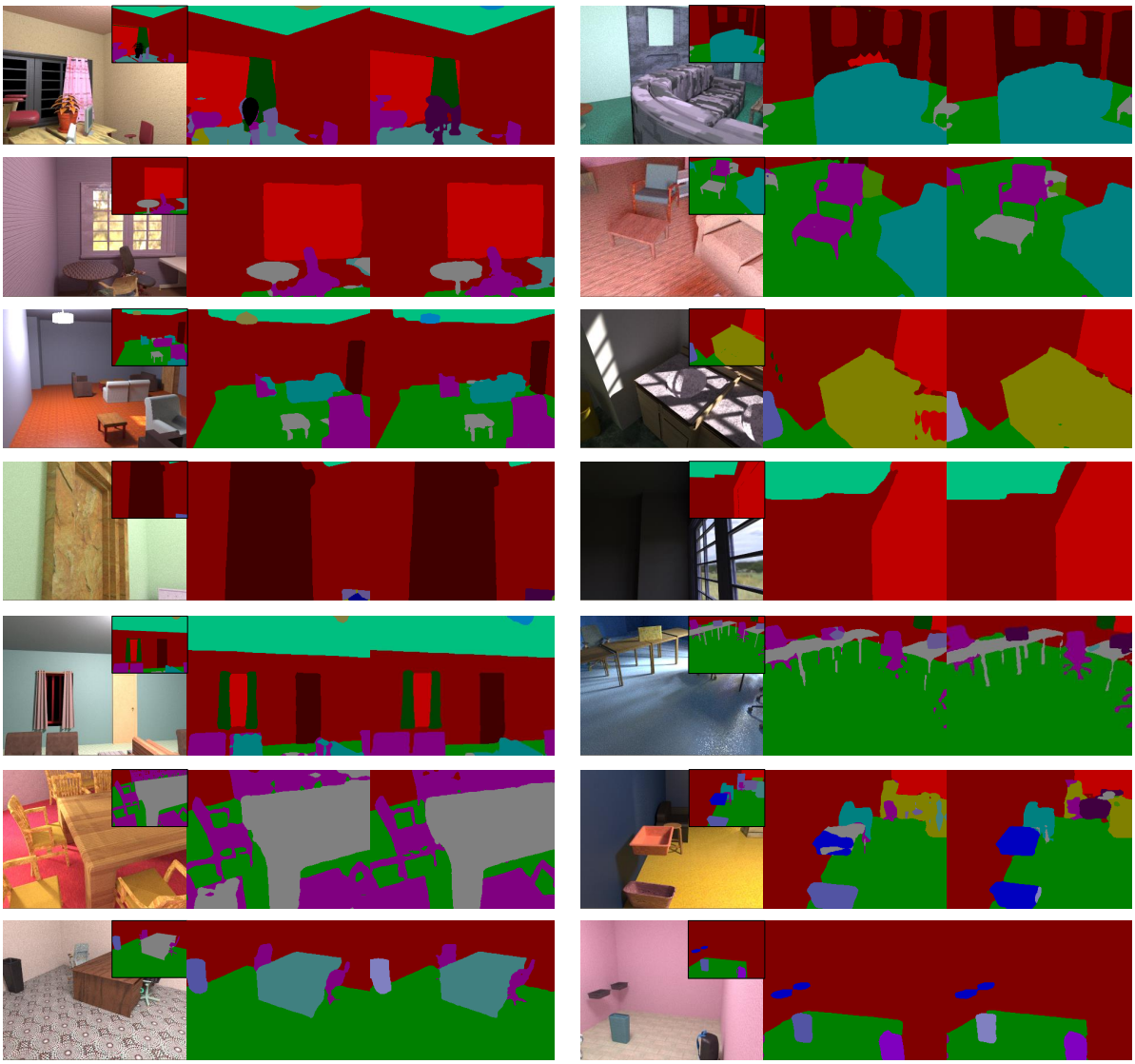

Figure S13: Further qualitative examples of semantic segmentation on OpenRooms.

	Albedo	Normal			Depth			Roughness	Semantics	
	loss	loss	mean(°)	med.(°)	loss	Abs Rel	RMSE	loss	mIoU	mAcc
Multi-task model	9.47	4.08	14.17	4.90	2.98	0.1066	0.2647	6.69	23.1	28.8
Segmentation-only	-	-	-	-	-	-	-	-	23.4	28.9
W/o segmentation	8.66	4.12	14.32	5.19	3.15	0.1070	0.2573	6.33	-	-

Table S9: Ablation study for models including the multi-task model, the segmentation-only model where the albedo, normal, depth and roughness heads are removed, as well as the model without the segmentation head while keeping all other four heads, evaluated on the OpenRooms dataset. We report the scale invariant L2 loss (10^{-3}) for Albedo, L2 loss (10^{-2}) for normal, scale invariant log L2 loss (10^{-2}) for depth, L2 loss (10^{-2}) for roughness and scale invariant $\log(x+1)$ L2 loss for per-pixel lighting. Angular errors for normal estimation, Abs Rel and RMSE errors for depth estimation (valid depth range of 0.1m – 8m), as well as mean IoU (mIoU) and mean accuracy (mAcc) for semantic segmentation are also included.

OpenRooms is the source domain with 100k images, while the target domain is unlabeled NYU with 15k frames from its raw dataset. We train an unsupervised domain adaptation model for depth estimation using T2Net [13]. We clip the depth values within the range of $[0, 10]$ meters and evaluate the predictions within the range of $[1, 8]$ meters, following

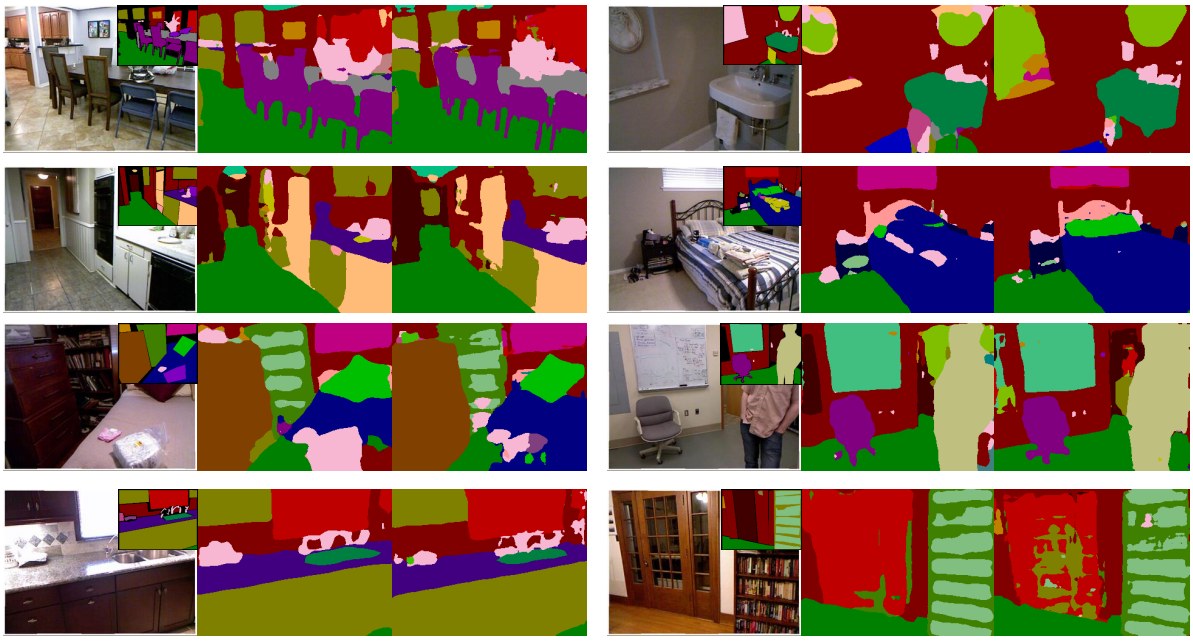

Figure S14: Further qualitative examples of semantic segmentation on NYUv2.

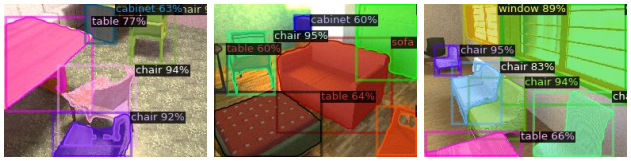

Figure S15: Qualitative examples of instances segmentation results on Openrooms.

the settings of [13]. We observe that the domain gap can be largely addressed with unsupervised domain adaptation in Table S10 and Figure S18. The target supervised numbers are trained on the NYUv2 training set which is composed of 1440 frames. Further, pre-training on the large-scale OpenRooms synthetic data before fine-tuning on the smaller scale NYUv2 training data leads to improved performance.

S7. Dataset Creation using SUNRGBD Data

To demonstrate that our framework can generalize to other datasets, we present our scene reconstruction results based on scanned indoor scenes from the SUNRGBD dataset. Unlike ScanNet [4], SUNRGBD only contains partial scans of the rooms with extremely incomplete and sparse point clouds. Moreover, unlike Scan2CAD [1], SUNRGBD only has 3D bounding box annotations for furniture locations and lacks full poses. Using this as initialization, we adjust the pose of the CAD models by simply using grid search to minimize the Chamfer distance between the CAD model and the point cloud in the bounding box. Then we assign appropriate materials and lighting to the CAD models, as described in the main paper. In our experience, differing qualities of scans need to be addressed for geometry creation in different datasets, but our material and lighting mapping transfer across datasets with minimal effort. In Figure S19, we visualize the reconstruction results for SUNRGBD by rendering the created scenes from different viewpoints, with different material assignments. The rendered images present diverse appearances with plausible material and lighting assignments, with complex visual effects such as soft shadows and specularity being correctly handled.

S8. Microfacet BRDF Model

We use the simplified microfacet BRDF model of [7]. Let A, N, R be the diffuse albedo, normal and roughness. Our BRDF model $\mathbf{f}(A, N, R)$ is defined as

$$\mathbf{f}(A, N, R, l, v) = \frac{A}{\pi} + \frac{\mathbf{D}(h, R)\mathbf{F}(v, h)\mathbf{G}(l, v, N, R)}{4(N \cdot l)(N \cdot v)}$$

$$\mathbf{D}(h, R) = \frac{R^4}{\pi((N \cdot h)^2(R^4 - 1) + 1)^2}$$

$$\mathbf{F}(v, h) = (1 - F_0)2^{(-5.55473(v \cdot h) - 6.98316)v \cdot h} + F_0$$

$$\mathbf{G}(l, v, N, R) = \mathbf{G}_1(v, N)\mathbf{G}_1(l, N)$$

$$\mathbf{G}_1(v, N) = \frac{N \cdot v}{(N \cdot v)(1 - k) + k}, \quad k = \frac{(R + 1)^2}{8}$$

where v and l are the view and light directions, while h is the half angle vector. Further, $\mathbf{D}(h, R)$, $\mathbf{F}(v, h)$ and $\mathbf{G}(l, v, N, R)$ are the distribution, Fresnel and geometric terms, respectively. We set $F_0 = 0.05$, following [7].

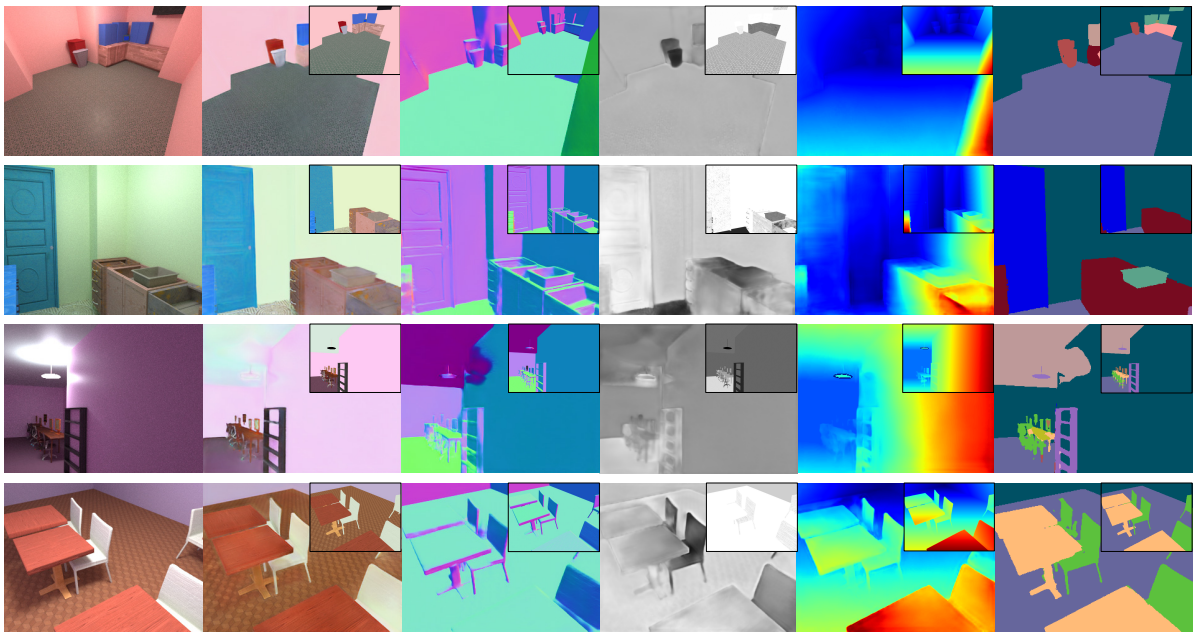

Figure S16: Further qualitative examples of multi-task estimation on OpenRooms.

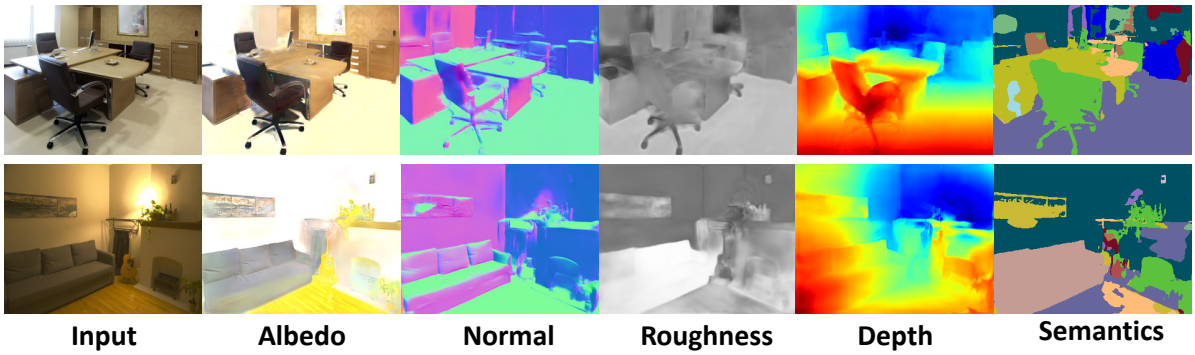

Figure S17: Further qualitative examples of multi-task estimation on real images.

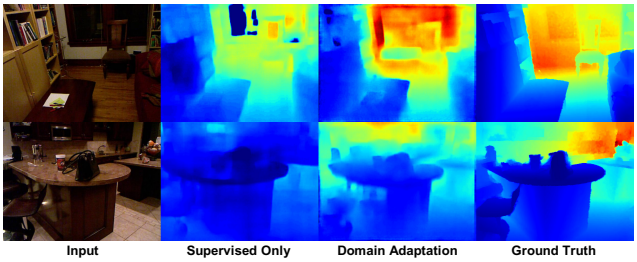

Figure S18: Qualitative results of depth estimation on Supervised only model and Domain Adaptation model.

S9. Ground Truth for Lighting

Ground truth for complex light transport effects is difficult to acquire in real scenes and not available in prior synthetic datasets. Compared to prior datasets for indoor lighting estimation such as [8], OpenRooms not only provides spatially-varying per-pixel environment maps but also extensive ground truth for light source positions, colors and intensities. Moreover, it also provides ground truth for individual contributions of each light source to the pixel intensity, rendered with direct illumination and with direct and indirect illumination combined, with and without occlusion being considered. Such new supervisions allow us to model the scene appearance with light sources in the scene turned off or on, which may enable new challenging lighting editing applications in the future. Figure 5 in the main paper and Figure S20 in the supplementary show examples of our light source supervision. We now provide further implementation details.

Per-light shading To render the individual contribution of each light source in the scene, we need to turn all other light sources off and keep only one light source on. This is straightforward for lamps, but not for windows, especially if

	lower is better				higher is better		
	Abs Rel	Sq Rel	RMSE	RMSE log	δ ¡ 1.25	δ ¡ 1.25^2	δ ¡ 1.25^3
Source Only	0.4723	1.0160	1.8520	0.7765	0.2049	0.3806	0.5379
Unsupervised Domain Adaptation	0.2388	0.3060	0.9430	0.3280	0.5770	0.8266	0.9336
Target Only	0.2031	0.2139	0.7477	0.2603	0.6536	0.8968	0.9675
Source + Target Finetuned	0.1721	0.1865	0.6663	0.2206	0.7465	0.9261	0.9753

Table S10: Unsupervised domain adaptation results for depth estimation on NYUv2 labeled test set. *Source Only* model is trained on OpenRooms while *Unsupervised Domain Adaptation* model is trained on OpenRooms and adapted to NYU unlabeled data. *Target Only* model is trained on labeled NYUv2. *Source + Target Finetuned* model uses the Source Only model as pre-trained weight and finetuned on labeled NYUv2. Metrics are computed using the labeled data which does not appear during training.

Figure S19: Synthetic scene reconstruction results using scanned indoor scenes from SUNRGBD dataset. We visualize the reconstructed scenes rendered from different views with different material assignments.

there are multiple windows in the room, as shown in Figure S20. To achieve this, we provide the plane parameters of each window to the renderer. When sampling the environment map, we check whether the ray hitting the environment map passes through the plane approximation of the window geometry. We only consider the contributions of those rays that pass through the window.

Shading without occlusion We render all the ground-truth with our customized OptiX-based GPU renderer. OptiX handles visibility term by calling its `rtTrace` function to detect if a ray will be occluded. To render without the visibility term, we simply do not use `rtTrace` in the renderer.

S10. Physically-Based GPU Renderer

We render our dataset efficiently using a physically-based GPU renderer. We make one crucial design choice to improve the rendering speed while maintaining the rendering quality – when rendering the spatially-varying lighting, we not only uniformly sample the hemisphere, but also sample the light sources. The contributions of the two sampling methods can be combined together using the standard power rule in multiple importance sampling [10]. This allows us to capture the radiance from small light sources in the scene with far fewer samples. More formally, let η be the ray direction, $\mathbf{P}_L(\eta)$ be the probability of sampling η when sampling the light sources, $\mathbf{P}_U(\eta)$ be the probability of uniformly sampling the hemisphere and $\mathcal{I}$ be an indicator function that is equal to 1 when a light source is sampled and 0 otherwise. Further, let $\mathbf{L}(\eta)$ be the radiance. Then, the contribution of sampling η towards the corresponding pixel on the hemisphere can be written as:

$$I \cdot \frac{\mathbf{P}_L^2}{\mathbf{P}_L^2 + \mathbf{P}_U^2} \frac{\mathbf{L}}{\mathbf{P}_L} + (1 - I) \cdot \frac{\mathbf{P}_U^2}{\mathbf{P}_L^2 + \mathbf{P}_U^2} \frac{\mathbf{L}}{\mathbf{P}_U}, \qquad \text{(S1)}$$

where dependence of $\mathbf{L}$, $\mathbf{P}_L$, $\mathbf{P}_U$ on η is omitted for clarity.

References

[1] Armen Avetisyan, Manuel Dahnert, Angela Dai, Manolis Savva, Angel X Chang, and Matthias Nießner. Scan2CAD: Learning CAD model alignment in RGB-D scans. In *Proc. CVPR*, 2019. 10

[2] Jonathan T Barron and Jitendra Malik. Intrinsic scene properties from a single RGB-D image. In *Proc. CVPR*, 2013. 1

[3] Jiacheng Chen, Chen Liu, Jiaye Wu, and Yasutaka Furukawa. Floor-SP: Inverse CAD for floorplans by sequential room-wise shortest path. In *Proc. ICCV*, 2019. 3, 4, 5

[4] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-

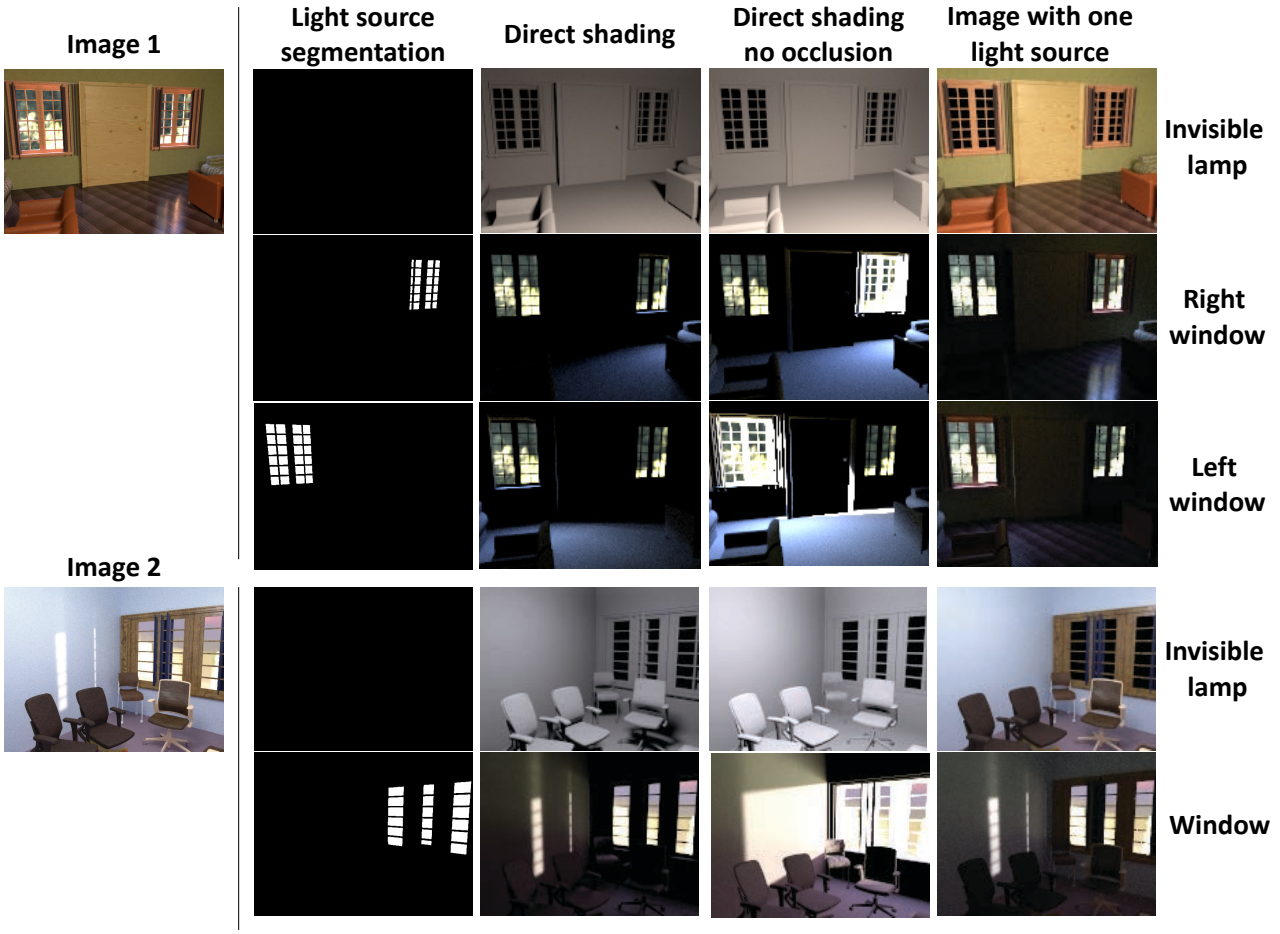

Figure S20: Light source ground-truth that can be provided by OpenRooms dataset. Unlike prior dataset [8], we provide the contribution of each individual light source, with the influences of direct/indirect illumination and visibility being separated.

annotated 3d reconstructions of indoor scenes. In *Proc. CVPR*, 2017. 10

[5] Marc-André Gardner, Kalyan Sunkavalli, Ersin Yumer, Xiaohui Shen, Emiliano Gambaretto, Christian Gagné, and Jean-François Lalonde. Learning to predict indoor illumination from a single image. *ACM Trans. Graphics*, 9(4), 2017. 1

[6] Mathieu Garon, Kalyan Sunkavalli, Sunil Hadap, Nathan Carr, and Jean-François Lalonde. Fast spatially-varying indoor lighting estimation. In *Proc. CVPR*, 2019. 1, 2

[7] Brian Karis and Epic Games. Real shading in unreal engine 4. *Proc. Physically Based Shading Theory Practice*, 2013. 10

[8] Zhengqin Li, Mohammad Shafiei, Ravi Ramamoorthi, Kalyan Sunkavalli, and Manmohan Chandraker. Inverse rendering for complex indoor scenes: Shape, spatially-varying lighting and SVBRDF from a single image. In *Proc. CVPR*, 2020. 1, 11, 13

[9] Soumyadip Sengupta, Jinwei Gu, Kihwan Kim, Guilin Liu, David W Jacobs, and Jan Kautz. Neural inverse rendering of an indoor scene from a single image. In *Proc. ICCV*, 2019. 3, 5

[10] Eric Veach. *Robust Monte Carlo methods for light transport simulation*, volume 1610. Stanford University PhD thesis, 1997. 12

[11] Hang Zhang, Kristin Dana, and Ko Nishino. Friction from reflectance: Deep reflectance codes for predicting physical surface properties from one-shot in-field reflectance. In *Proc. ECCV*, 2016. 5, 6

[12] Yinda Zhang, Shuran Song, Ersin Yumer, Manolis Savva, Joon-Young Lee, Hailin Jin, and Thomas Funkhouser. Physically-based rendering for indoor scene understanding using convolutional neural networks. In *Proc. CVPR*, 2017. 3, 5

[13] Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. T2net: Synthetic-to-realistic translation for solving single-image depth estimation tasks. In *Proc. ECCV*, September 2018. 9, 10