Learning-based Spotlight Position Optimization for Non-Line-of-Sight Human Localization and Posture Classification — Supplemental Material —

Sreenithy Chandran
Arizona State University
schand56@asu.edu

Tatsuya Yatagawa
Hitotsubashi University
tatsuya.yatagawa@r.hit-u.ac.jp

Hiroyuki Kubo
Chiba University
hkubo@chiba-u.jp

Suren Jayasuriya
Arizona State University
sjayasur@asu.edu

1. Synthetic Dataset Augmentation

The generation of a large amount of scene data is a time-consuming process. Consequently, we perform data augmentation to enhance the number of synthetic scenes available in this study during training. To perform data augmentation, we have developed a plugin for Blender, an open source software package for creating digital content. The plugin, coded in Python, automates the randomization of both component postures and material parameters, which include, but are not limited to, diffuse and specular albedo, as well as surface roughness. Afterward, the plugin computes rendered images efficiently using GPU-accelerated path tracing provided by the Blender Cycles rendering engine. In our pursuit of enhancing the realism of the rendered images, we have also added real noise to the resulting images, thereby mimicking the noise inherently present in real-world photography.

When using Blender, for the spotlight source, we use 250 W–350 W power lamps to match closely with the real illumination source. We employ the spotlight's blend feature to create a radial decrease in intensity, similar to what is seen in real life, where light spreads to adjacent surfaces. We add one or two light sources to the scene, and each light source is a point, spot, or area light. We position the light sources and make them not too bright or can cast shadows from moving objects. The camera is also moved randomly at varying distances from the wall, between 1ft - 2ft, the height of the camera was also changed. We do not, however, move the camera in-between a NLOS trajectory capture. When exporting the scenes from Blender to obj format for the differentiable rendering, we triangulate the faces.

In Fig. 4, we show examples of both synthetic and real scenes used for training and testing. These scenes have a variety of textures and obstacles. Furthermore, we included objects in the line of sight to broaden the dataset. We also take advantage of the data augmentation module discussed earlier to enhance the diversity of synthetic scenes. We use publicly available models of indoor spaces as well as models of some scenes created with Solidworks.

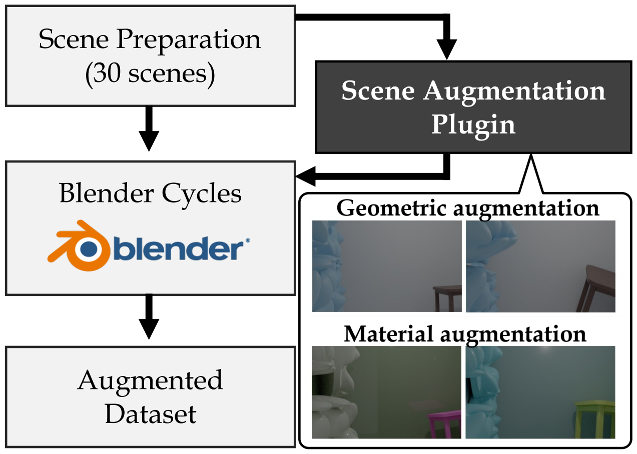


Figure 1. Simulated dataset generation: We design the scenes using SolidWorks and use a data augmentation module that works with Blender Cycles to randomize the postures and materials of components in each scene.

Visit our project page for more details. We have shared the real dataset which contains highly accurate ground truth localization values with a precision of 0.50 mm. This could be of great benefit for future NLOS research.

2. Pipeline Details

The pipeline is written in PyTorch, and a forward pass of the differentiable rendering takes approximately 0.05 seconds for synthetic data. The differentiable renderer, which transforms the mesh into a rendered image, loads the obj file

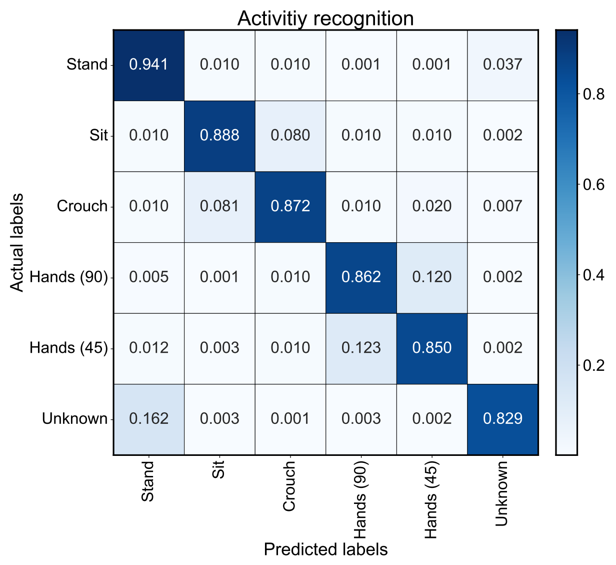


Figure 2. Confusion matrix for posture classification for real data

as a tensor. Hence, we store the mesh files in tensor format to expedite data loading.

For NLOS Networks, the last layer of ResNet-18 is removed and the output from it is subsequently input to an MLP comprised of 256 neurons. The fully connected (fc) layers in the MLP are each followed by a ReLU. The input to the first fc layer is 256 and is reduced to an output dimension of 128. The second fc layer takes this 128 size input and outputs 2 values corresponding to the position (x,y). For posture classification, the last linear layer has an output of 6, and this is followed by a softmax activation.

3. Additional Analyses

3.1. Posture Classification – Real Data

We plot the confusion matrix of the posture classification results for real data in Fig. 2. Our network performs excellently in distinguishing the various postures, as demonstrated by the confusion matrix. It did have a slight difficulty in distinguishing between the “Sit” and “Crouch” labels, as well as the “Hand 90°” and “Hand 45°” labels. This could be due to the network’s difficulty in accurately recognizing the temporal signatures of hand movements. Nonetheless, the difficulty of distinguishing similar actions is a common problem in the field and is not exclusive to our approach.

3.2. Dynamic Illumination Prediction

Our proposed approach estimates which part of an LOS surface to shine a light on, resulting in an improvement in the metric scores across the NLOS volume. We further investigated whether performance could be further improved if a rough estimate of the NLOS position is available to adjust the system. See Fig. 3 for the network modification to our proposed pipeline. We use the same IEN architecture and NLOS networks. To obtain a coarse localization input of the NLOS object, we used a ResNet+MLP similar to the NLOS Network. After removing the final fc layer of the ResNet, we apply Global Average Pooling(GAP) to the output feature maps. Then the MLP on top of the GAP is used to perform the localization. This study was carried out using only synthetic data.

Table 1. Comparison of the performance of dynamic illumination update with our proposed method

Task	Ours	Dynamic Illumination
Localization (↓) (Average Error [cm])	8.10	7.42
Posture Classification (↑) (Accuracy [%])	96.13	96.46

We noticed that the illumination face predicted by the IEN did not vary much when the NLOS estimated position was available as an additional prior. The most significant change in prediction was observed when the NLOS object was moved far away from the wall. We can consider various reasons for this observation, such as the LOS mesh being a rough approximation of the scene geometry or the illumination spotlight spreading across adjacent faces. The improvement in performance was so small that we did not feel the need to incorporate this dynamic illumination into the real acquisition process.

In Fig. 5, we present further real-world results of our method’s illumination prediction compared to [1]. The first column displays lidar scans of LOS walls, the second column illustrates pictures of the scene illuminated by our method, and the third column shows the illumination determined by [1] using an optimization-based approach. The last row of the figure shows that our method selects the shiny cupboard as the illuminated patch, since it predicts that more light will be sent into the NLOS scene, compared to [1], which chooses the region closest to the camera based on its mathematical modeling approach.

References

[1] Sreenithy Chandran and Suren Jayasuriya. Adaptive lighting for data-driven non-line-of-sight 3d localization and object identification. *British Machine Vision Conference (BMVC)*, 2019.

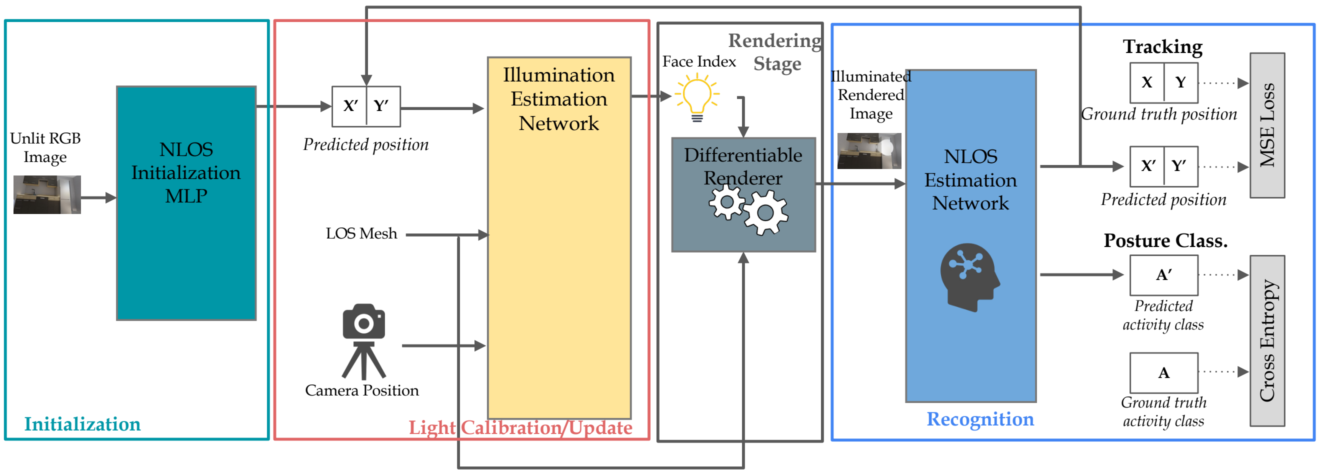


Figure 3. Pipeline for dynamic update of illumination. A simple coarse localization network is added at the beginning of our pipeline. This takes an rgb capture of the LOS surface and predicts a localization value that is fed to our existing pipeline. There is also a feedback of the final NLOS prediction back into the IEN.

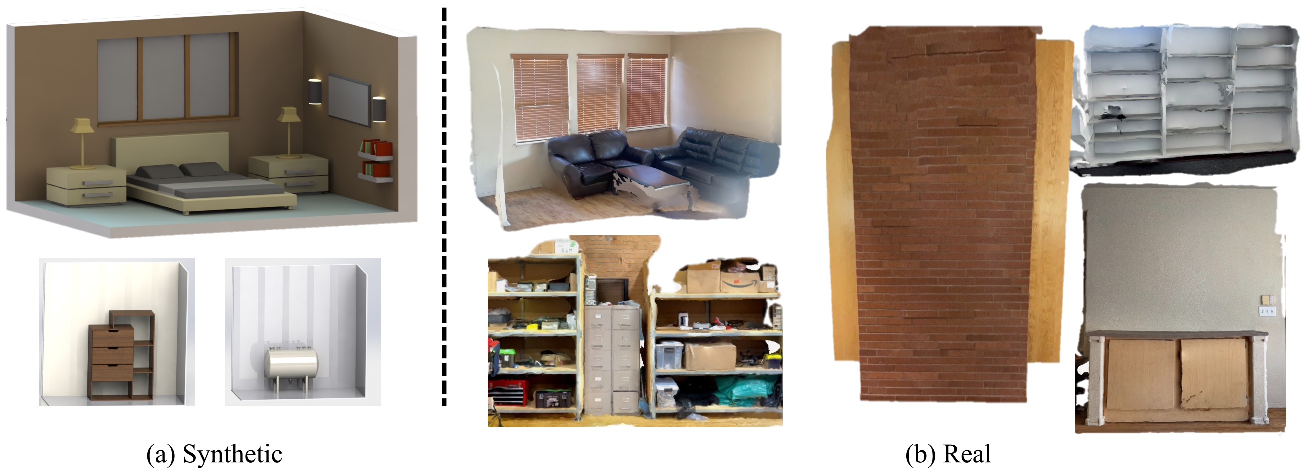
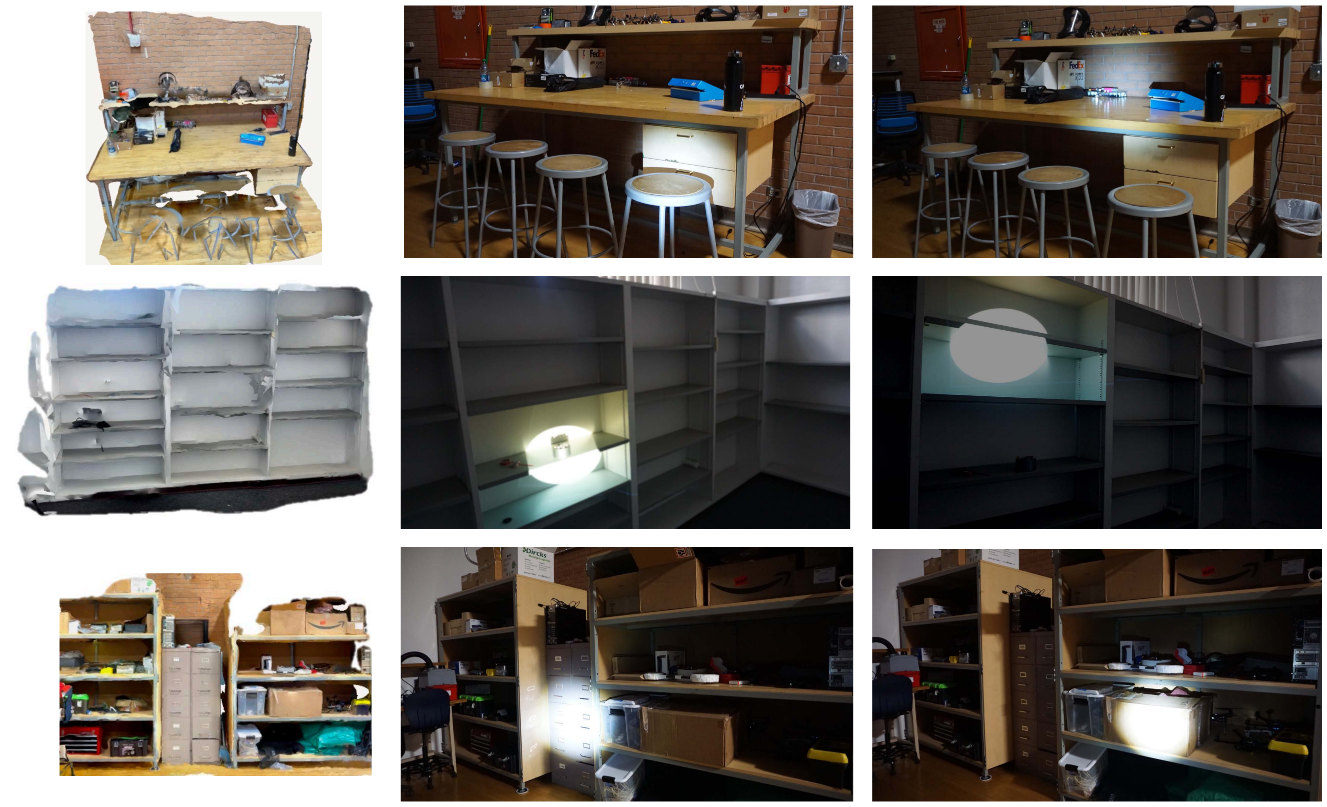


Figure 4. Sample scenes used for training and testing, note the variation in terms of textures, occlusions, etc. We consider walls with varying levels of objects/clutter present in the LOS. We also consider planar walls with different textures.



(a) LOS Mesh (b) Our Method (c) Chandran et. al.

Figure 5. Results of illumination estimation network, where (a) the first column shows polygonal meshes of LOS walls, (b) the second column shows pictures of the scene spotlighted based on the suggestion from our learning-based technique, (c) the third column shows the illumination determined by [1]