Data-Driven image-based material estimation network

Liutong Jiang, CSSE, SZU

Abstract

The influence of lighting on texture transfer and material prediction has always been a trouble problem. Most results of texture transfer would have some abnormal noise. The results of material estimation become more and more important, if we could predict the material of objects, the results in scene rendering are become more reliable. In order to achieve this goal, I try to estimate the material in images. So that we could get the base color, roughness, metallic and normal of original image. I believe that we could complete texture transfer or material prediction task through these four attributes. In this work, I complete a material estimation network. This network benefited from the large amount of data.

Keywords： Material Estimation, Texture Transfer, Material Transfer, Image Processing.

1 Introduction

In many existing texture estimation or material estimation works, most of the textures or materials are estimated from real world photos. However, when textures or materials estimated from photos, it is inevitably affected by illumination and the reflectance of the objects. The attributes of objects has a great influence on estimation of object textures and materials. In Photo-to-Shape[1], the authors mentioned in his failure case that his estimation of materials will be affected by some illumination reflectivity information: ”This incorrect material prediction is caused by different reflectance effects on different parts”. Actually, estimate the material is the work of inverse rendering. According to Indoor Scence Inverse Rendering[2],the research of inverse rendering is mainly focused on the decomposition of single object or single scene attribute.

In previous work, most of works are limited by the scarcity of data sets, they often use self augmentation to enhance the amounts of their data. Recently, ABO datasets[3] has been published, I notice that this dataset largely solves the problem of lacking enough real data to estimate material. In their work, they proposed a U-Net network with ResNet34 as the backbone to estimate the base attributes from a complex, real world single view on the basis of its dataset.

2 Related works

2.1 Scene attributes estimation

Most material estimation tasks focus on scene.According to Sengupta et al.[2], they first try to estimate physical attributes of a scene, they jointly estimates albedo, normals, and lighting of an indoor scene from a single image. They predict the lighting through the sum of direct renderer and predict supplemental lighting.

In 2021, Wang et al.[4] focus on decompose the indoor scene into albedo, normals, depth and lighting. This work model the scene illumination through an Volumetric Spherical Gaussian representation, which parameterizes the exitant radiance of the 3D scene surfaces on a voxel grid. Each grid use the HDR lighting inferred by their model to insert highly specular objects and produce realistic cast shadows and high-frequency details.

Also the work of Li et al.[5] realized material estimation by inverse rendering the scene. In this work, they could obtain a complete scene reconstruction, estimating shape, spatially-varing lighting and spatially-varing, non-Lambertian surface reflectance.In other words, this work predicts the SVBRDF attribute of the scene.

2.2 Object material estimation

There are some works focus on the object material estimtion. In Zhang et al.[6], try to predict the BRDF of objects. But in this work, we should first train the object on the network so that we could get the result of the object on other perspective.Also some works follow the Nerf, Boss et al.[7] try to explicityly optimize the material corresponding to each image for varing illuminations or globally for static illumination. Zhang et al.[8] predict lighting, material, geometry and surface normals from posed multi-view images of specular objects, and predict the other perspective with illumination as results. But these works are flawed, because they can estimate the material of the same category objects at most, they cannot realize the light decomposition across object categories. Object in real scenes often come from multiple categories.

Different from the previous methods. the model trained in this paper can be light decomposed in different categories, so it is of great significance in material estimation of scene objects.

2.3 Texture transfer and material prediction

In the work of Wang et al.[9], they try to avoiding the impact of lighting on texture transfer through select the most possible shade image of different light direction, and compute the sum of them to simulate the light. Huang et al.[10], they improve the previous methods to estimate the material before perform texture transfer, from this we can see how the material affects the texture and material prediction.

In the work of material transfer, the work of Park et al.[11] and the work of Hu et al.[1] also notice the influence of material to their work.The incorrect material could lead the bad results.

3 Method

3.1 Overview

The single view material estimation network is called SV-Net here. The structure of this network is shown in Figure 1.

It should be noted here that, as shown in Figure 1, a skip connection should be made between the Encoder and the Decoder, and the information obtained from the Encoder compression should be spliced with the Decoder before deconvolution.

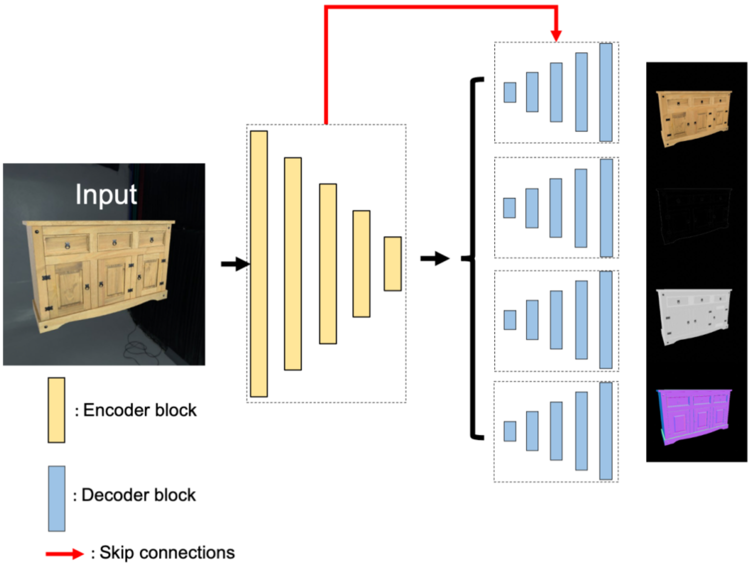
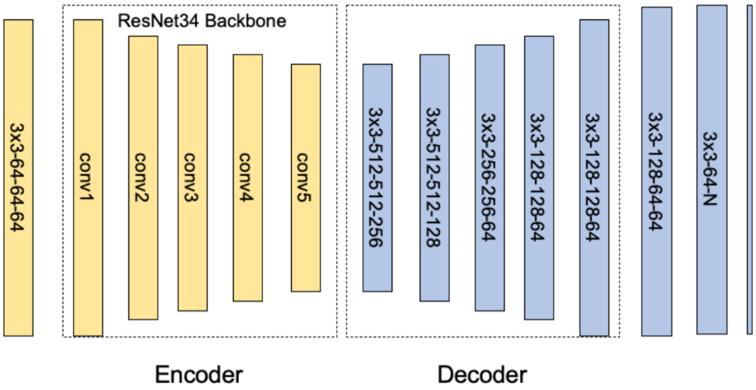


Figure 1: Overview of the method

3.2 The structure of Encoder and Decoder

In SV-Net, the structure of Encoder and Decoder is shown in Figure 2.



The detail structure of Encoder and Decoder 2: Interface

In Figure 2, K×K-N-M-X means the kernel size is K×K. At the same time, there are N input channels, M medium channels and X output channels. The network in Decoder means double convolutional block. But the last layer only has one convolutional block, it means 64 input channels and N output channels.

This network mainly improves the medical image segmentation network U-Net. The whole network follow the structure of Encoder-Decoder. The Encoder uses the Backbone of ResNet34, and Decoder decode the image features through deconvolution methods. Four decoders are constructed to obtain the base color, roughness, metallic, and normal of the input image.

3.3 Loss

The loss in this method is MSE loss, we input the image and mask into network, and the output of network is predict base color, roughness, roughness, normal. At the same time, we have ground truth base color, roughness, metallic and normal. In order to minimize the image distance between predict result and ground truth, We just need to compute MSE loss between them.

4 Implementation details

4.1 The process of Datasets

In order to input the data set into the network for training, our goal is to build a class extend from torch.utils.data.Dataset. Before building a data sets, we still need to do some processing on the original files so that we could complete the building of the data set class.

・Divide the train and test data

After download the data set of material estimation, we actually need to split the training data and testing data according to the given csv file. There are some information about the purpose(train or test) of data in csv file. In this step, we only need to read the csv file, determine whether the corresponding cell is ’Train’ or ’Test’, and then use the ’os’ toolkit to put it into different folders.

・Get the path of data

The goal of us is to build the class extend from torch.Dataset, so we need to use the process the ’Train’ data according to the convenient format. According to the arrangement path of each rendering, we could find that each folder always contains the following folders:

1. depth: stores the depth information of Ground Truth, which can be used to compute loss with network output.

2. metallic_roughness: stores the ground truth of metallic and roughness. It should be noted that the metallic and roughness information are concat to the same images. It’s means the metallic and roughness only occupy two of three channels.It could be used to compute loss.

3. normal: stores the ground truth of normal, which can be used to compute loss with network output.

4. render: stores rendered images of the same object in different scenes. There are 3 sub folders under this folder. The folders are different scenes, and each scene folder contains 91 images from the same perspective, which are used as network input.

5. segmentation: stores the mask information of the object, which is used as the input of the network.

6. metadata.json: stores information about the camera pose.

For each large folder, we need to read all file paths under the render folder into the same list, we should randomly select 40 random numbers, and select 40 images paths from this list for training. At the same time, we can obtain the mask path and ground truth path by computing the number of render sub folders * 91 based on this random number.

Through the above operations, we can obtain the mask path correspond to the render image, and all labels path correspond to the image.We just need to input these image into the data set class, and use the PIL toolkit to open the image to get the RGB matrix. It’s should be noted that roughness and metallic should be processed because they only occupy the channel of the image.

4.2 The design of the network

For the input structure as shown in Figure 2, the leftmost layer is a 3×3-64-64-64. Actually, the channel we input should be 4 channels, 3 channels stores the RGB information, and 1 channel stores the mask information. I think the author may have made an error when drawing the network structure diagram, so in the initial preprocessing layer, we need to adjust it to 3×3-4-64-64. At the same time, it should be noted that the author has processed that the activation function used in its network structure is LeakyReLU. There is no difference among the network structure of other Encoder layers, we just need to follow the the structure of ResNet34.

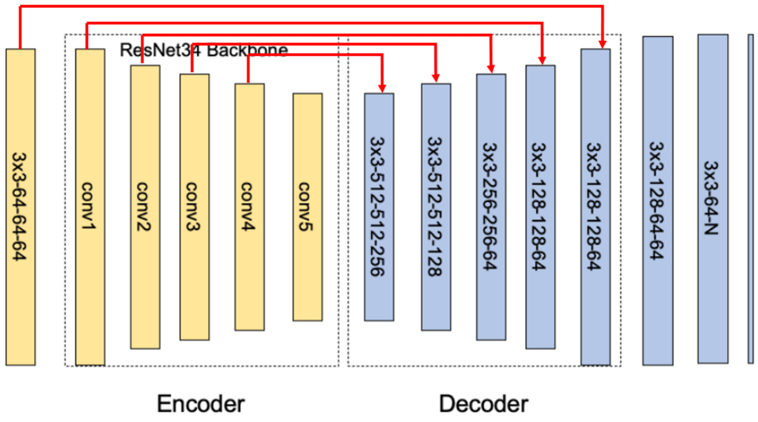


Figure 3: The skip connection

For the output structure, we just need to design the network structure as shown in Figure 2. For the structure of one encoder and four decoders, we only need to set four identical decoders, the parameters do not share among these four decoders. For the case that the output channel of one layer in decoder is not equal to the input channel of the next layer, the reason is that the U-Net skip connection is referenced here. The low dimensional feature map need to be concated with the result of the deconvolution. The key here is that we need to consider the concat details. Here I have supplemented the skip connection to the network structure originally given by the author. As shown in Figure 3, the red represents the concat between Encoder and Decoder layers. For example, the results of the X we input should be concat to the last layer of Decoder after being processed by preprocessing layer. The results of the conv1 need to be concat to the penultimate layer of the Decoder.

4.3 Details

When processing data set, after we get path from os.listdir(), we need to sort all the sub folders and sub files. When I wrote the code in Windows, even if I do not use list.sort, I can also get the list order by the number of the picture name. But after two complete trainings in Linux, I found that I still could not get good results, and the results of any training would be particularly vague. At this time, I found that only use os.listdir() is not enough in Linux, we need to use the list.sort to specify the numeric keyword of the file name for sorting after obtaining all files.

When doing the skip connection concat, we need to do the central crop for the part obtained by original Encoder. Through the central crop operation, we can get the feature map with the same size as the Decoder part.The Decoder will concat the feature map of low and high dimensions and input it into the next double convolution block.

5 Results and analysis

5.1 Qualitative experiment

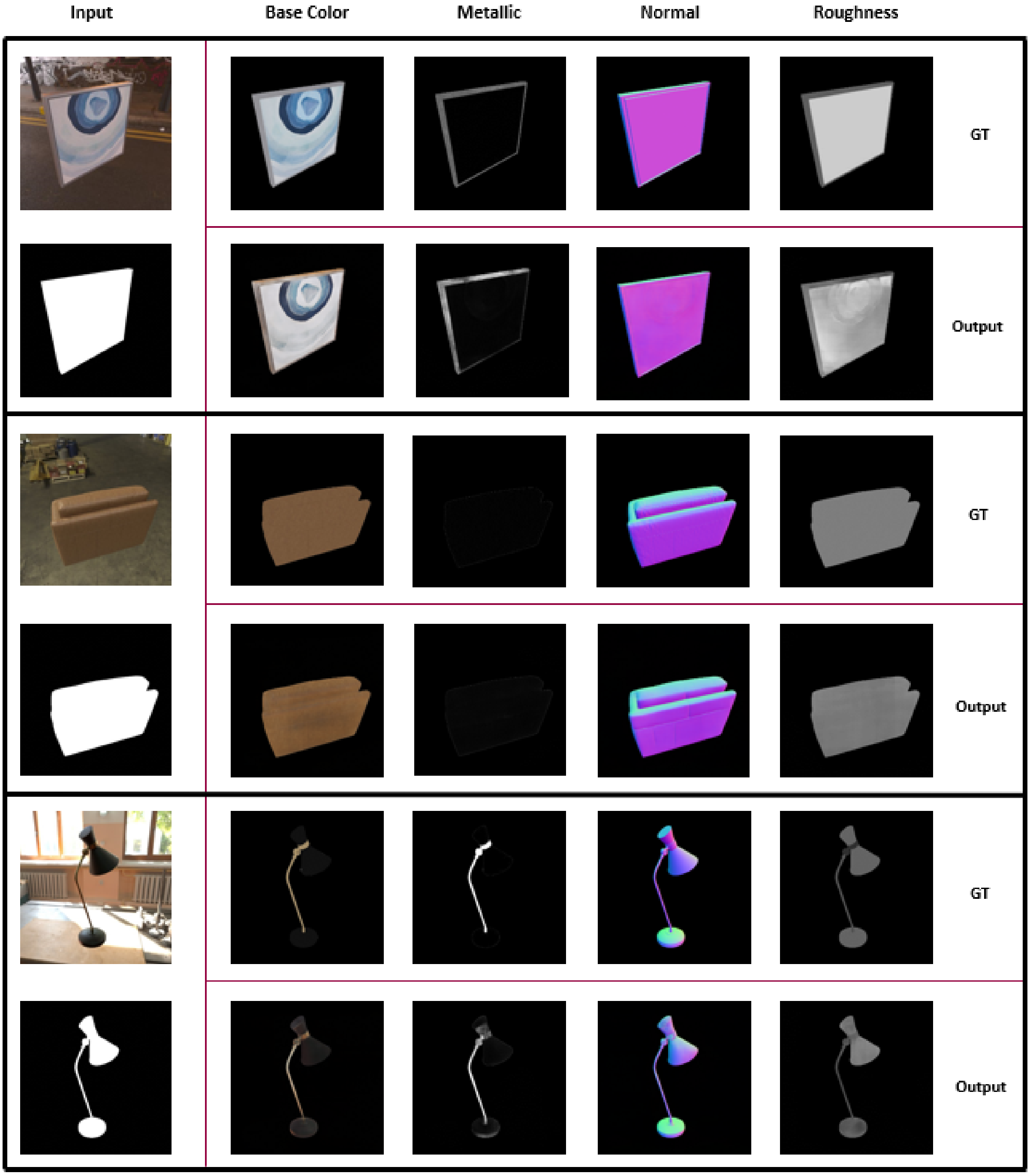


Figure 4: Experimental results on the U-Net with ResNet34 backbone. The leftmost is input which consist of source image and correspond mask. The right images are outputs and corresponding ground truth. There are four groups in this image.

As shown in Figure 4, we could compare the network results and ground truth. We could find that the results are generally similar to ground truth. In Figure 5, we show the result on the U-Net, it is obvious that the results of U-Net are worse than the U-Net with ResNet34 backbone.

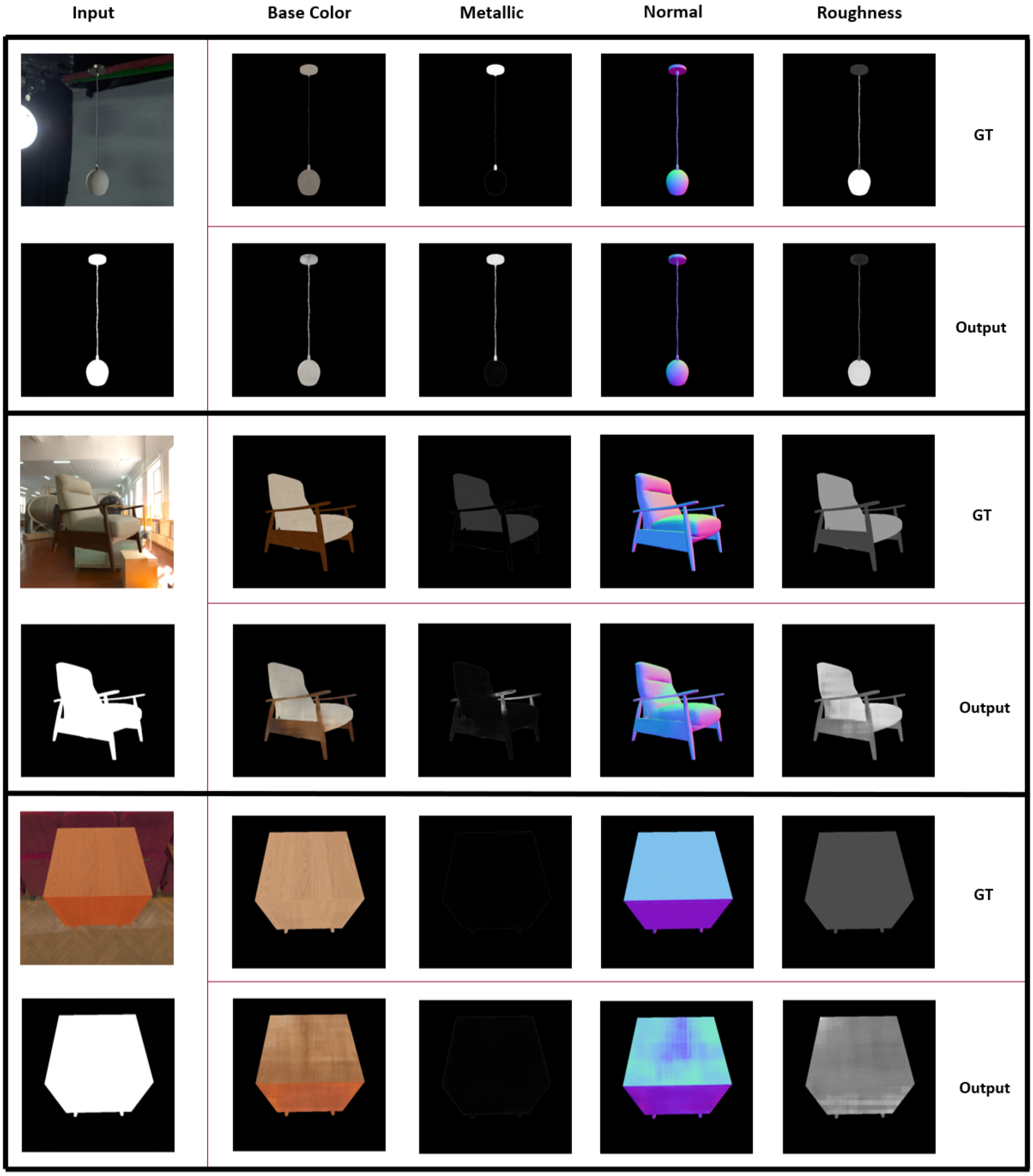


Figure 5: Experimental results on the U-Net. The structure in this table is same as Figure 4

5.2 Quantitative experiment

Methods	base_color	metallic	normal	roughness
U-Net	0.00322	0.00630	0.00246	**0.02237**
Resnet34&U-Net	**0.00280**	**0.00472**	**0.00128**	0.02334

Figure 6: Quantitative results.

Here, we propose two networks. We evaluate the results through compute the MSE loss between the predict results and the ground truth. As shown in Figure 6, we could notice that the U-Net with ResNet34 backbone are better than the U-Net.

5.3 Failure case

The results in this work also have some failure cases. As shown in Figure 7. In this case ,we could find that the source image is affected by a strong light. Even the human eye can hardly distinguish the specific texture information. In this case, it is very unrealistic to complete the light decomposition.

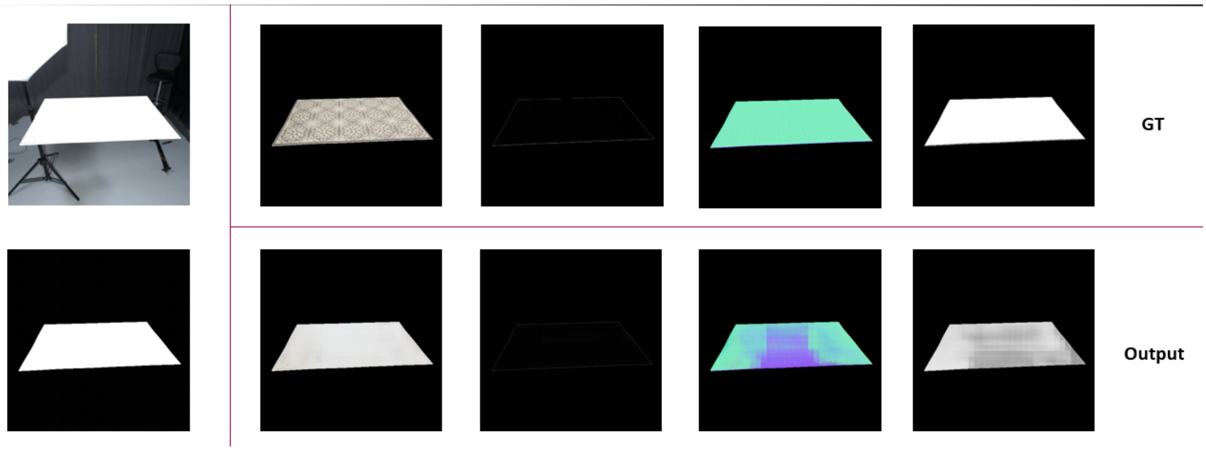


Figure 7: Failure case. The leftmost is input image and corresponding mask. In the right of image, there are ouputs and corresponding groundt truth.

6 Conclusion and future work

In the case of massive data drive, the difference between the current network output and the ground truth are not too big, most of them can be very close to ground truth. In my opinion, the method in this paper obtains good results on val sets.

In the future, we could improve the network structure. Because the network in this paper is a very simple network. If we try to conclude more information into network, it could be believe that our network would get the best effect. It is still worth improving is that we can try to add some realistic constraints. For example, the main body of a wooden cabinet should be made of wood, and the normals of the same face shoud be consistent without changing. After adding these material and geometric constraints, the results should be better.

The disadvantage is that I did not synthesize all the decomposed results back to the source image, because this requires an additional renderer to be trained for synthesis.

References

[1] HU R, SU X, CHEN X, et al. Photo-to-Shape Material Transfer for Diverse Structures[J]. ACM Trans. Graph., 2022, 41(4).

[2] SENGUPTA S, GU J, KIM K, et al. Neural inverse rendering of an indoor scene from a single image [C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019: 8598-8607.

[3] COLLINS J, GOEL S, DENG K, et al. Abo: Dataset and benchmarks for real-world 3d object understanding[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022: 21126-21136.

[4] WANG Z, PHILION J, FIDLER S, et al. Learning indoor inverse rendering with 3d spatially-varying lighting[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021: 12538-12547.

[5] LI Z, SHAFIEI M, RAMAMOORTHI R, et al. Inverse rendering for complex indoor scenes: Shape,

spatially-varying lighting and svbrdf from a single image[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 2475-2484.

[6] ZHANG X, SRINIVASAN P P, DENG B, et al. Nerfactor: Neural factorization of shape and reflectance under an unknown illumination[J]. ACM Transactions on Graphics (TOG), 2021, 40(6): 1-18.

[7] BOSS M, BRAUN R, JAMPANI V, et al. Nerd: Neural reflectance decomposition from image collections[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021: 12684-12694.

[8] ZHANG K, LUAN F, WANG Q, et al. Physg: Inverse rendering with spherical gaussians for physics-based material editing and relighting[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 5453-5462.

[9] WANG T Y, SU H, HUANG Q, et al. Unsupervised texture transfer from images to model collections. [J]. ACM Trans. Graph., 2016, 35(6): 177-1.

[10] HUANG H, XIE K, MA L, et al. Appearance modeling via proxy-to-image alignment[J]. ACM Transactions on Graphics (TOG), 2018, 37(1): 1-15.

[11] PARK K, REMATAS K, FARHADI A, et al. Photoshape: Photorealistic materials for large-scale shape collections[J]. arXiv preprint arXiv:1809.09761, 2018.