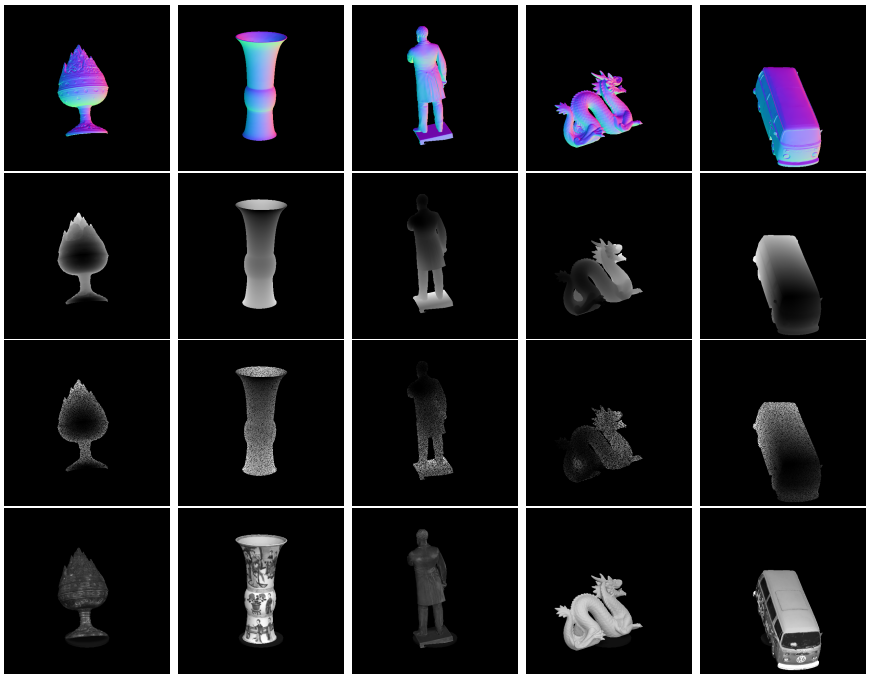
1 Training Details

The approaches are trained on dataset "synthetic-50-5" as mentioned in Chapter **??** with 3000 scenes. Each scene has a depth map with dimension $128 \times 128 \times$ in height, width and channel, an image with dimension $128 \times 128 \times 1$. The depth map is converted to 3D vertex map as introduced in Chapter **??**. The light map is calculated based on vertex map and the known light position. We create a tensor in PyTorch that includes vertex map, image and the light direction for each scene and considered it as one training case. Thus 3000 scenes has corresponding 3000 training cases. Each scene has a corresponding ground-truth normal map for loss calculation and the evaluation.

The training processes are evaluated in every 100 epochs with 29 evaluation scenes that model never seen before, which contains the 5 different objects in the "synthetic-50-5" test set. Figure 1 shows some of the test scenes during the training work.



(a) Baoshanlu (b) Vase (c) Garfield (d) Dragon (e) Bus

Figure 1: Some of the test scenes during the training. From top to bottom, normal map, depth map, noised depth map, grayscale image.

The training pipeline use batch size 8, Adam optimizer (**adam**), learning rate start from 1×10^{-3}, learning schedule [8,1000], learning decay factor 0.5. The model is trained with PyTorch 1.10.0a0, CUDA 11.4.1, GPU with single NVIDIA GEFORCE RTX 3090. It takes 14 hours to train GCNN and 35 hours

to train the Trip-Net. We terminate the training when the evaluation on the test dataset converged.

2 GCNN model based on Geometry Information

The GCNN model is the base model of the whole thesis. The architecture is described in **??**. We use a single GCNN to estimate the surface normal based on geometry information. It uses vertex map as input to estimate the corresponding tangent surface normal map.

In order to verify the applicability of UNet architecture and Gated Convolution layer for the normal inference task, two similar models are created. We replace all of the gated layer to standard convolution layers in the network but keeps all of the other settings same in model "CNN". It is used to verify the performance the gated convolution layers. As mentioned in chapter **??**, the gated layer is designed to deal with noised input. Since all of the vertex map in the dataset has been added noise, the GCNN is supposed to over-perform "CNN". Another model called "NOC" is designed to verify the skip connection in the UNet, which simply removes the skip connections in the network but keeps other settings same. Is is designed to show the validation of skip connections. Figure 2 shows the training history on BerHu Loss. The GCNN approach achieves a lower loss from start to the end of the training.

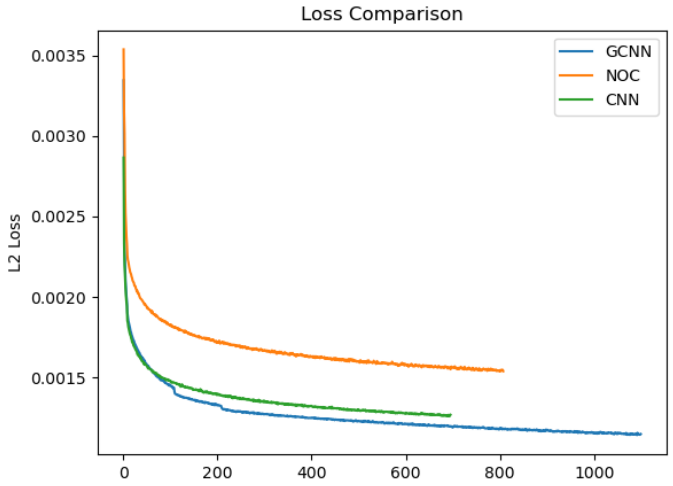


Figure 2: The training history of GCNN model. The line chart records the training BerHu loss of the model GCNN, NOC and CNN.

3 Trip-Net model based on Calibrated Illuminated RGB-D Image

The Trip-Net model uses three times GCNN architecture with 4 times fusions, which is more difficult to train. It takes the calibrated illuminated RGB-D images as input to estimate the surface normal map. For the sake of comparison, we take the GCNN model as a baseline, to observe the beneficial of Trip-Net architecture. Since the Trip-Net is more complicate than GCNN, we also explored the optimum fusion times of the Trip-Net to see any possibility for the model simplification. A set of similar models have been trained with same settings but different fusion times, denotes by Trip-Net-Fx, where x denotes the fusion times. We evaluate the fusion times from 1 to 4. Figure 3 shows the training history of these models on BerHu Loss.

As shown in the loss figure, ... (discussion about the figure)

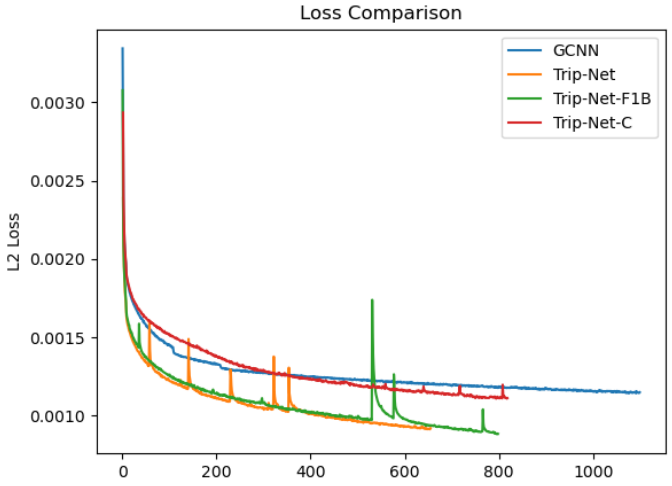
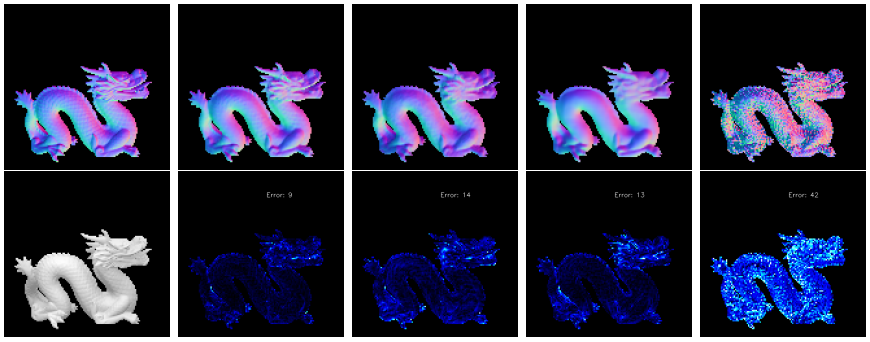


Figure 3: The training history of Trip-Net model. The line chart records the training BerHu loss of the model Trip-Net, Trip-Net-F1, Trip-Net-F2, Trip-Net-F3, GCNN.

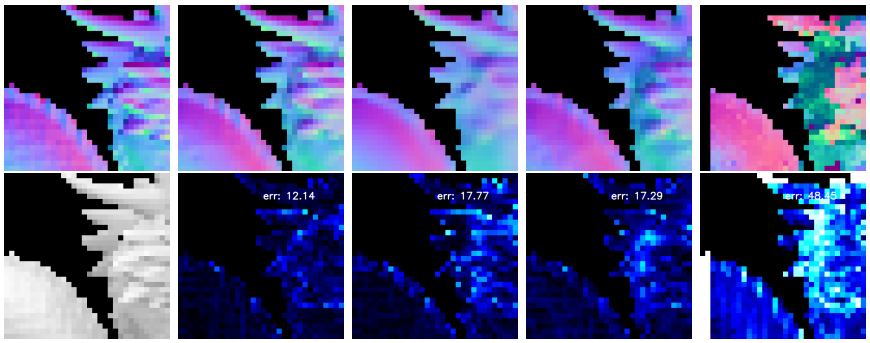
4 Evaluation



(a) GT (b) GCNN (c) CNN (d) NOC (e) SVD

Figure 4: Surface Normal Inference based GCNN model on "Dragon´´ object. The first row shows the estimated surface normal. The second row is the angle error map.

A qualitative evaluation on object "dragon" is shown in Figure 4. SVD approach is considered as the baseline shown in last column. As shown in the figure, learning based methods performs better than SVD in terms of angle error. The SVD approach is failed to deal with semi-dense input since there exists many points that missing neighbors. The GCNN model is especially good at noise input due to the gated convolution layer design. As a further detailed comparison, 5 gives a closer visualization on the same object.



(a) GT (b) GCNN (c) NOC (d) CNN (e) SVD

Figure 5: Zoom in of the center region of Dragon object. The first row is surface normal, the second row is the corresponding errors. NOC model has no skip connnection, CNN model replace gated convolution layer to standard convolution layer.

As shown in figure 5, the GCNN method gives a sharper edge prediction on the horn area of the dragon object, as well as the scales, whereas the no skip version (NOC) is blurry in the same area.

The CNN version has the skip connection thus gives a better detail than NOC model. However, if we compare the error map of GCNN and CNN in figure 4, the CNN has less accurate in the smooth area than GCNN model. Like the dragon body, CNN model has a overall higher error than GCNN. It is because the noise of the input still disturb the CNN model and it takes the input noise into account for normal estimation which deviate to the correct surface normal. When we look back to the GCNN based method, we can found that the surface normal has better performance in the smooth area compare to the CNN approach and a sharp detail compare to the no skip connection version.

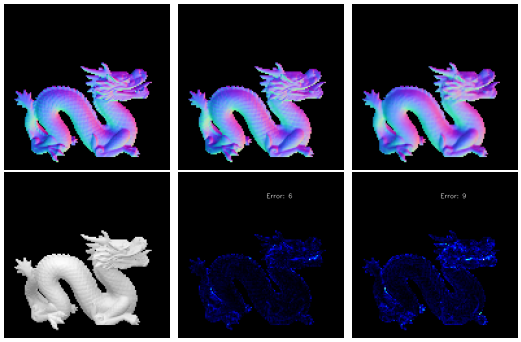
Table 1 gives a quantitative evaluation for GCNN model. It bases on 100 different test scenes in the "synthetic-50-5´´ dataset with angle metrics for evaluation.

Model	**Angle**	**Time /ms**	**bz**	**lr-schedule**	**lr-df**
SVD(baseline)	41.14	320.40	8	8,1000	0.5
GCNN	10.64	10.44	8	8,1000	0.5
GCNN-NOC	13.61	5.38	8	8,1000	0.5
CNN	15.35	4.15	8	8,1000	0.5

Table 1: The performance of the GCNN model for geometry information based normal inference. The angle error is the average angle error of all valid pixels in the test case. bz stands for batch size, lr-schedule stands for learning rate schedule, lr-df stands for learning rate decay factor.

5 Surface Normal Inference based on Calibrated Illuminated RGBD images

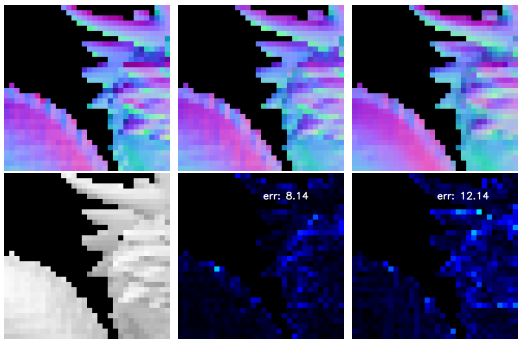
For the approach using illuminated calibrated RGBD image, the task is undertaken by Trip-Net introduced in **??**. The qualitative evaluation is shown in figure 6. As a comparison, we placed GCNN result in the last column. The training settings for all the models are exact the same to ensure fairness. As shown in the figure, TripNet uses illuminated calibrated RGBD image has a better performance than GCNN model. The dragon scales are sharper in TripNet result. Figure 9 gives a closer visualization.



(a) GT (b) TripNet (c) GCNN

Error: 0 90

Figure 6: TripNet qualitative evaluation. Surface Normal Inference on "Dragon" object. The first row shows the estimated surface normal. The second row is the angle error map.



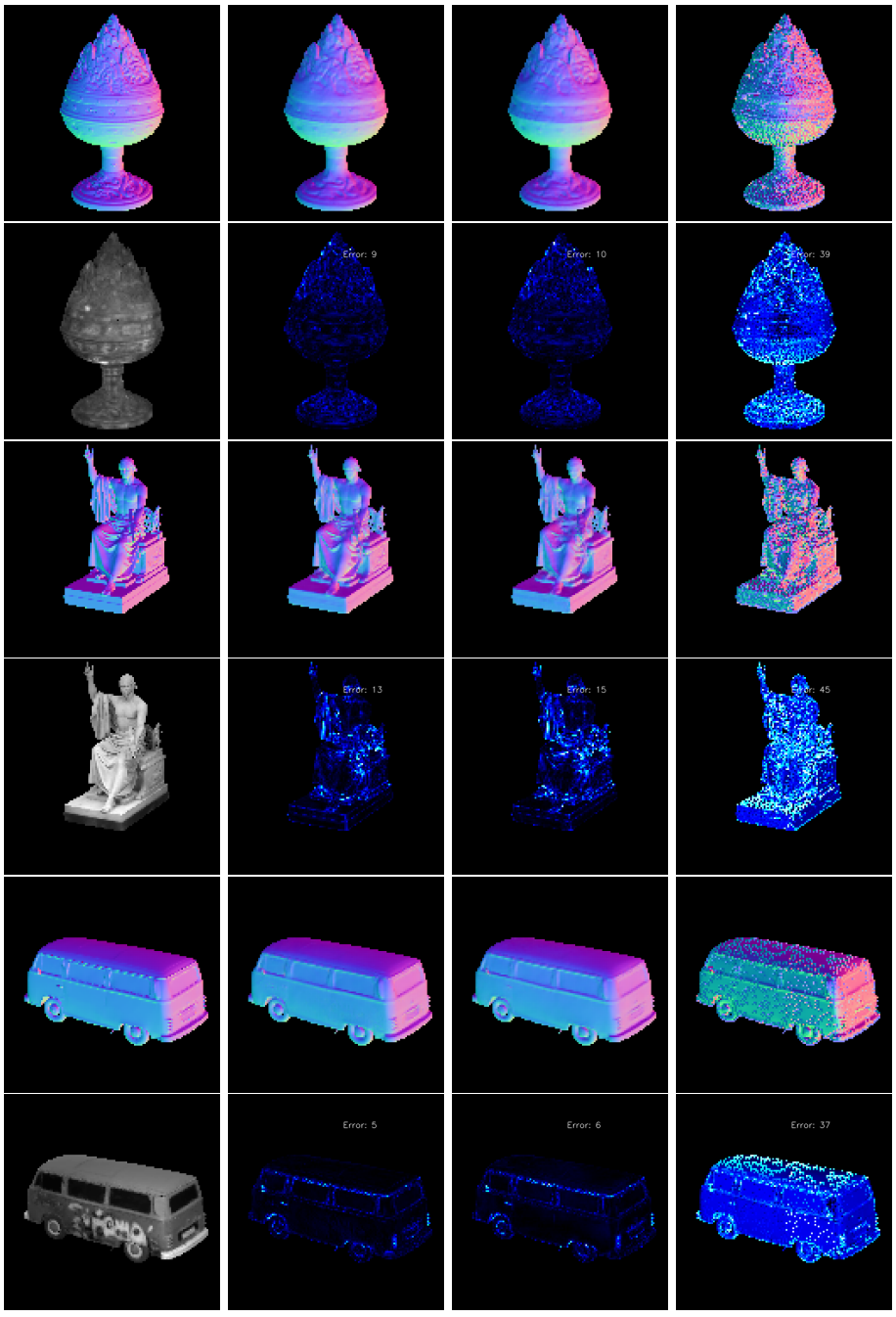
(a) GT (b) Trip-Net (c) GCNN

Figure 7: Zoom in of the center region of Dragon object. The first row is surface normal, the second row is the corresponding errors.

Model	**Angle**	**Time /ms**	**bz**	**lr-schedule**	**lr-df**	**l/i. Nr.**
SVD	41.14	320.40	-	-	-	0
GCNN	10.64	10.44	8	8,1000	0.5	0
TripNet-CNN	10.46	28.74	8	8,1000	0.5	1
TripNet-F1B	9.22	44.59	8	8,1000	0.5	1
TripNet	**9.17**	43.79	8	8,1000	0.5	1

Table 2: A quantitative evaluation on proposed approaches. The angle error is the average angle error of all valid pixels in the test case. The time unit is in millisecond. bz is the batch size, lr-schedule is learning rate schedule. lr-df is learning rate decay factor, l/i. Nr is the number of light-image maps used for each scene

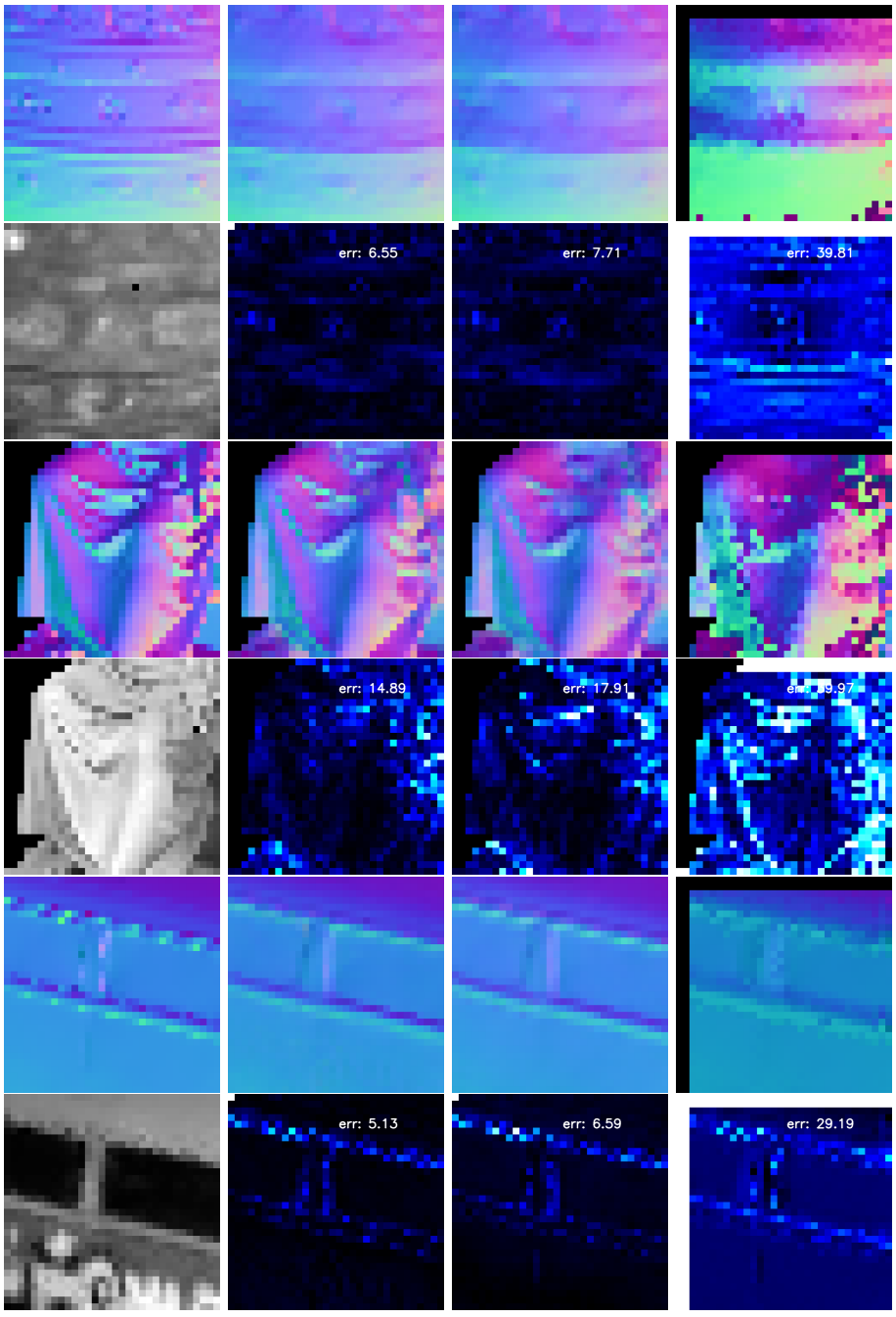
5.1 Comparison



(a) GT (b) TripNet (c) GCNN (d) SVD

Error: 0 90

Figure 8: Evaluation on objects Baoshanlu, Washington statue, Bus(from top to bottom).



(a) GT (b) Trip-Net (c) GCNN (d) SVD

Figure 9: Zoom in of the center region of the objects in Figure 8

From the Figure **??** we can observe the normal difference between ground-truth and GCNN predicted normals in another dimension. It separates the interval $[-1, 1]$, which is exactly the range of normal vector, to 256 sections. Then it counts the number of points locates in each section for 3 axes. The 3 axes are fitted quite well in most of interval but other than $[-0.25, 0.25]$ for x and y axes and interval close to -1 for z axis. Therefore a further constraint can be considered to the loss function related to the normal difference shown in this figure.

It is faulty that almost no normal has -1 z-component in GCNN predicted normal map. The reason?

5.2 Light Inpainting

The light inpainting task in this section in order to evaluate the noise inpainting performance of GCNN model. It takes semi-dense light map as input to predict the fully dense light map, as shown in Figure 10. The network is trained on 3000 light maps in the "synthetic-50-5" dataset with initial learning rate 0.001, learning schedule 8,1000 with decay factor 0.5, the batch size is 8, the feature map channel is 128 in all of the gated convolution layers. All the models are tested on a laptop with a single NVIDIA GeForce 940MX, 2GB memory.

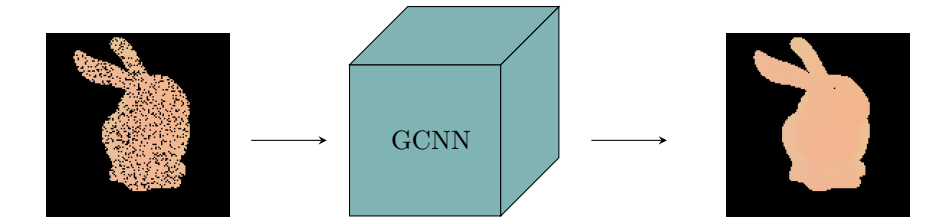
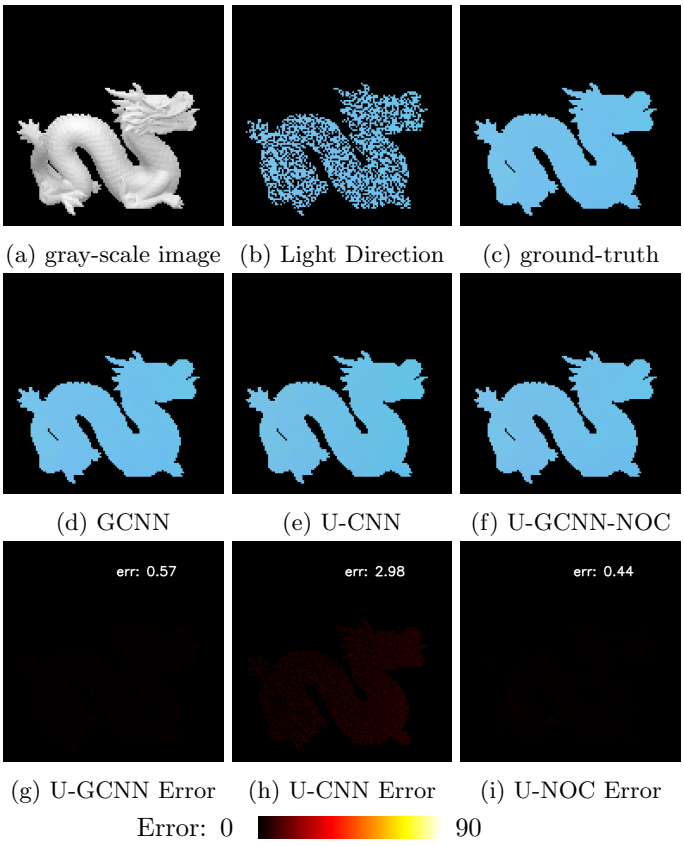


Figure 10: Light map inpainting based on GCNN architecture.

The result is shown in Table 3. Besides the GCNN model, we use two extra model as the comparisons. The first one is GCNN-NOC, where NOC means "no skip connection", the second one is CNN, which has the same architecture as the GCNN but only use standard convolution layers in the network. The GCNN architecture achieves the average angle error lower than 1 degree. As a comparison, U-CNN model has 4 degrees error, which shows the relative weakness of noise filter competence. Besides, the skip connection also booster the performance, since the no skip connection version still has error greater than 1 degree. The both two models for the inpainting task reveals the superiority of GCNN architecture. Figure 12 and 11 gives a quantitative and qualitative evaluation.

Model	Angle	Time /ms	bz	lr-schedule	lr-df
U-GCNN	0.59	14.45	8	8,1000	0.5
U-GCNN-NOC	1.31	16.14	8	8,1000	0.5
U-CNN	4.10	10.52	8	8,1000	0.5

Table 3: The performance of the GCNN model for light map inpainting. The angle error is the average angle error of all valid pixels in the test case. bz stands for batch size, lr-schedule stands for learning rate schedule, lr-df stands for learning rate decay factor.



(a) gray-scale image (b) Light Direction (c) ground-truth

(d) GCNN (e) U-CNN (f) U-GCNN-NOC

(g) U-GCNN Error (h) U-CNN Error (i) U-NOC Error

Figure 11: GCNN Normal Inference for light map inpainting based on dragon object. The degree error map is shown in last row. The average error is shown in the upper right corner in the error map.

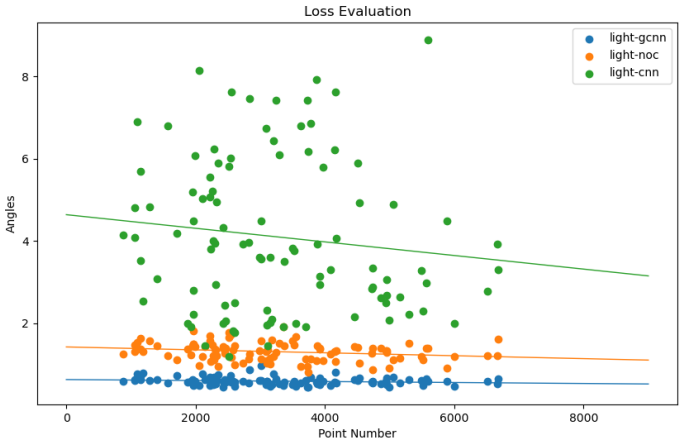
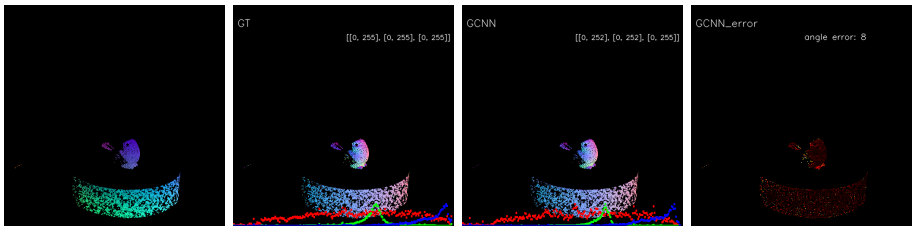


Figure 12: GCNN for light map inpainting on 100 scenes. The evaluated models are corresponding to the table 3, the line in the figure is the corresponding regression line of each model.

The evaluation visualization on real dataset is shown in Figure 13



(a) point cloud (b) ground-truth (c) GCNN (d) Angle Error

Figure 13: Evaluation on Real Dataset

We evaluate the U-GCNN model on 100 test images as a quantitative evaluation. The result is shown in Figure 14. The GCNN based method has average angle error around 10 degrees, whereas the no skip connection version has error above 12 degrees and the CNN version has error above 15 degrees.

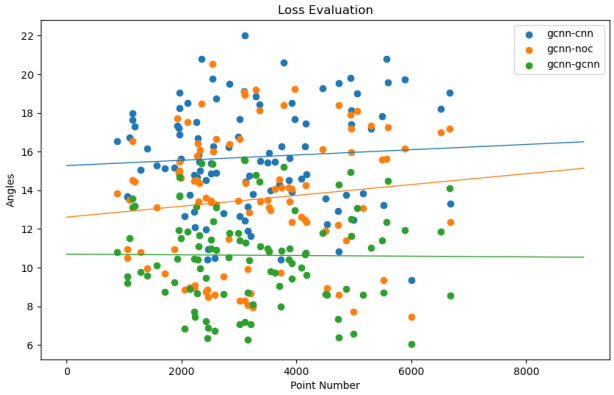


Figure 14: Evaluation of average angular loss on the whole test dataset with 100 scenes. The x-axis indicates the point number, the y-axis indicates the angles. The lines show in the picture are the regression line of each model.