Jingyuan Sha

June 3, 2022

Abstract

This is the abstract.

Chapter 1

Introduction

Surface normal inference from a single view deals with the task of object surface orientation estimation. It has many practical applications, such as augmented reality and robotics.

Standard methods compute normals from point cloud using neighboring information in image space or use Shape from Shading. However, the point clouds data provided by Kinect or similar RGB-D, LiDAR sensors are only semi-dense. As shown in Figure 1.1. If map these normals to polygon mesh, that is not good at all.

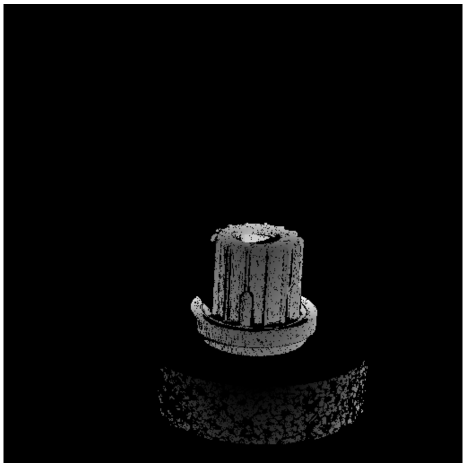


Figure 1.1: A depth map captured via infrared sensors. The missing pixels distributed almost all of the image.

Standard methods derives normal from each point and its neighborhoods' positions. It dependents on chosen neighbor size. There are two main drawbacks. First, if neighbor size is too small, it has larger noise sensitive, if it is too large, the output will too smooth and not crispy. Second, it is not robust with noise. Errors may occur in regions with inter-reflections, where we have errors in the 3D measurement. On the other side, it is also weak with missing pixel handling, which is a common issue in depth maps.

Although the depth image is incomplete, the depth sensor usually able to capture grayscale texture image, which are typically fully dense due to their passive nature. Furthermore, if the texture image is already illuminated by strong directional light of a video projector, whose position is known, then there should exist theoretical relations between light direction, normal direction, and grayscale image . Thus the normal can be inferenced better using the given image information and depth map.

Both depth and grayscale image are initially relevant to the machine learning algorithms. Based grayscale image and corresponding semi-dense depth map, a CNN model can be designed to inference the normal map, which gives more density and robust comparing to standard algorithm. However, the missing pixels

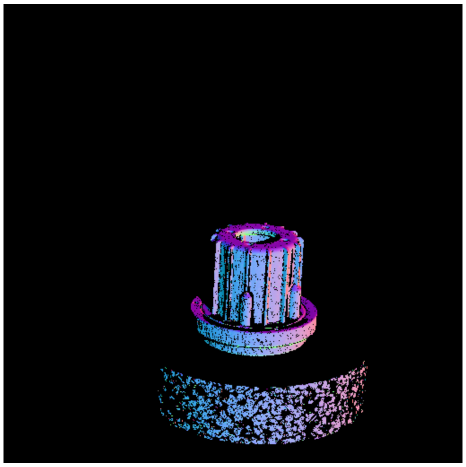


Figure 1.2: Semi-dense Normal Map calculated from depth map using standard method

in depth map can be distributed around the whole image, some of the image regions leave complete empty holes. This situation imposes further processing for the missing regions and some other challenges on the machine learning methods. In this thesis, we found a solution for the problems mentioned above.

The deep convolutional neural network is typically used for image classification, which achieved great success in last several years. [16], [18]. These kinds of network architecture takes a single image as input which usually employed for classification problems. The image is usually convoluted with convolutional layer and downsampling with pooling layers. The outputs of the network consists of a single value to represent the ID of corresponding class [18], or with set of values to represent the position of bounding boxes.[16].

However, in many other vision tasks, the output is demanded as an image, instead of predicting one or several classes of the input, but the classes of each pixel are predicted, like image segmentation[17]. In this case, the traditional network architecture is not suitable anymore.

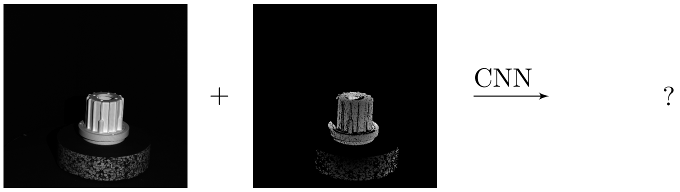


Figure 1.3: Left: Grayscale Image, Middle: Depth Map

In this thesis, we proposed a novel deep learning architecture for surface normal inference.

By applying our approach to the task of normal inference, a custom dataset is created from Unity using 3D models. The trained Normal models achieve a remarkably better prediction accuracy at a low computational cost comparing to the standard approaches for semi-dense point clouds.

Chapter 2

Related Work

2.1 Sparse Input processing

The depth map is supposed to be dense. Therefore, how to accept sparse depth map as input in CNNs is one of the most important problem. Some trivial solutions like median filters are good enough, if the missing pixels are sparse enough, however, for the case of huge missing holes in the depth map, it produces just a paltry result. Thus a reasonable guess is required for missing areas. Generally, it can be solved as image inpainting problems,[19],[14].

Some deep learning based method for image inpainting also achieved quite good performance for the hole mending task. Notably, in 2016, Oord et al. [13] proposed a gated activation unit for a CNN model,

$$\mathbf{y} = \tanh(W_{k,f} * \mathbf{x}) \odot \sigma(W_{k,g} * \mathbf{x})$$

to substitute the standard activation layer, where σ is the sigmoid function, which constricts the output value of second part between $[0, 1]$. The function is inspired by Long Short-Term Memory (LSTM) [8] and Rated Recurrent Unit (GRU).[2] It is originally used for learning complex interactions as LSTM gates does. In 2018, Yu et al. [20] employed same function for free-form image inpainting, which can be used to learn mask automatically from image it self.

Different to aforementioned approaches, Knutsson et al. in 2005 introduced normalized convolution [11] dealing with missing sample case for convolution operation, which aims to reconstruct the missing pixels from the sparse output sensed by cameras, which particularly considered the confidence of each interpolated pixels, since it provides the trustworthiness of the predicted value. The higher the reliability of the value inference, the better the model shape reconstruction.

In 2018, Eldesokey et al. [5] applied normalized convolution in CNN as normalized convolution layer that takes both sparse depth map and a binary confidence map as input to perform scene depth completion. In 2020, Eldesokey et al. [4] focus on modeling the uncertainty of depth data instead of assuming binary input confidence.

Guided method [10] requires addition information like RGB image or the certainty map of depth map, and fuse them together to predict the dense depth map.

2.2 Normal Inference

Usually, we based on point cloud, depth map or RGB/Grayscale image of the objects or scenes to inference the normals.

Traditional methods evaluate normals based on neighbor information of point cloud or depth map. In 2012, Holzer et al. [9] proposed method to calculate normal from covariance matrices. This method use integral image as input, which is able to run algorithm in a high frame speed. They smooth the depth data in order to handle the noise of depth image. The drawbacks are, as mentioned in the paper, the normals error go up when point depths change severely. In 2013, Fouhey et al. [6] proposed a method constructing a over-determined function systems to predict normals and solving it by algebra methods. Similarly, this approach gives a quick but coarse normal inference.

Recently, CNN based methods improve the performance of image processing to a brand new stage. In 2014, Eigen et al.[3] proposed a method predicting depth map directly from RGB image using CNN. In this case, no depth map is required. In 2016, Laina et al. [12] proposed a deeper network based on ResNet [7] with a well designed upsamling part. In 2018, Qi et al. proposed GeoNet[15], it integrates both algebra method and also CNN method to inference depthmap based on [12] [6].

It is worth to noticed that, the output of normal inference CNN model is not one or severl labels but an entire image or normal map with same size. Recently, Ronneberger et al proposed an architecture called UNet [17] for biomedical image segmentations. The architecture is shown in Figure 2.1.The first half network is a usual classification convolutional network, the second half replace the pooling layers and traditional fc layers in the traditional CNNs to upsampling layers, thus in the end of the second half, the output is able to back to the input size. The proposed network can successfully assigned each pixel a class for segmentation tasks. Under this symmetric network, an input image is downsampled 3 times and upsampled 3 times. Output image has exactly the same size as input image. The downsampling and upsampling both have large number of feature channels, which guarantee the network propagates the information to higher resolution layers.

In some case, the input is unstructured point cloud which can not be fed into a CNN entirely. Thus, a challenge task connect to the deep learning is the input format. Since different point clouds have different sizes. In 2018, Ben-Shabat et al. [1] presented Nesti-Net. It predicts the normal point by point with the help of neighbor points. It fixed the distance of considering neighbors to provide an unified input for CNN. In 2021, Zhou et al. [21] presents a method considering overlapping of different patches (a group of neighboring points) as input to evaluate normals.

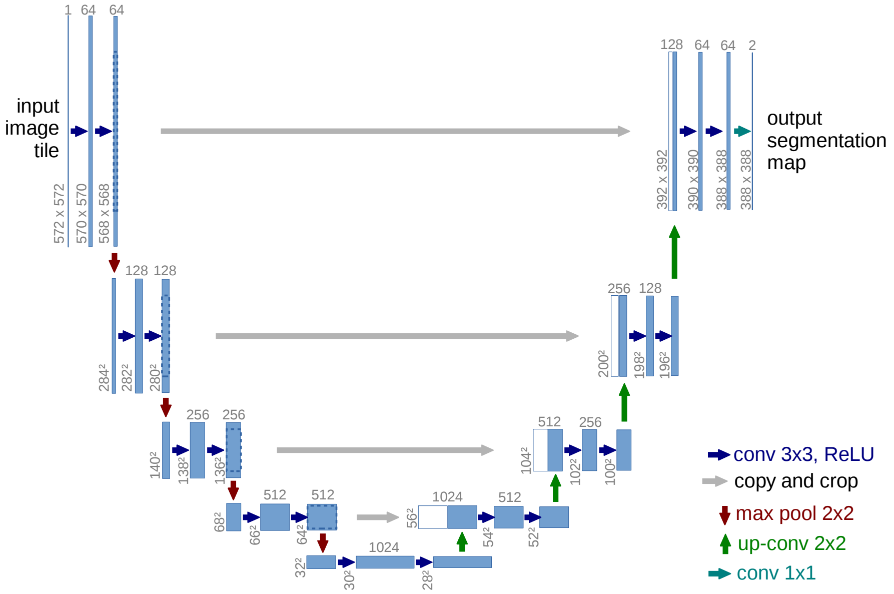


Figure 2.1: U-net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

Chapter 3

Approach

3.1 Dataset

The depth map captured by Kinect is usually semi-dense with a number of missing pixels, as shown in Figure 3.1. Consequently, the ground truth of inferenced normal map is incomplete, either.

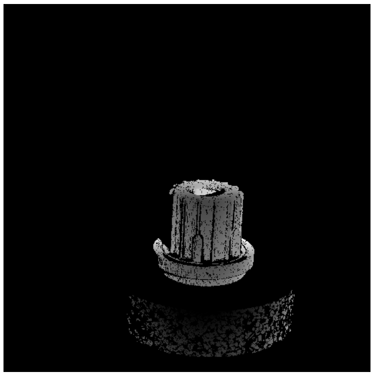


Figure 3.1: Depth Map of an object captured by Kinect

In view of this this kind of situation, a set of generated synthetic 3D scene via Unity is used as training dataset. The main advantage using generated scene is the complete information of all the information in the synthetic world. The depth map can be captured loss-free. The corresponding normal map can also be safely considered as ground truth. To construct the dataset, 22 3D models have been used and 1000 different scenes have been captured with different pose of each model. 3 kinds of images are recorded in each scene:

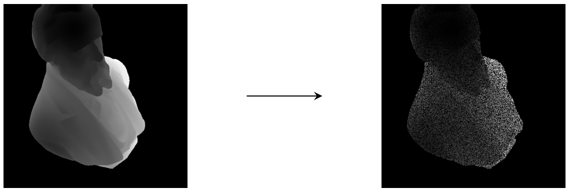
- Depth map
- Grayscale image
- Normal Map

3.1.1 Noise

The raw depth maps captured by Kinect usually have missing pixels. As shown in picture 3.1. Observing the depth map, the missing pixels can be divided into two groups.

- random missed single pixels
- missed dark areas

Since the input of evaluation is incomplete Kinect depth map, thus the input of training should have the similar pattern as well. That is, adding the similar noise to synthetic depth map. For the two kinds of depth noise, first can be simulated by adding uniformly distributed black pixels to the whole depth map, whereas the second can be dealed with a high-pass filter to filter out dark pixels and set them to 0.



Specifically, for each pixel with index (i, j) in a synthetic depth map D, if its value less than a threshold B, then set it to 0, i.e.

$$D(i,j) = \begin{cases} 0 & D(i,j) \leq B \\ D(i,j) & D(i,j) > B \end{cases}$$

for the rest of the pixels, the following two equal-opportunity situations will occur,

- pixel value remain same
- pixel value set to 0

3.2 Gated Convolution

Gated Convolution layer[20], the output of the layer with input size (N, C_{in}, H, W) and output $(N, C_{out}, H_{out}, W_{out})$ can be described as:

$$o(N_i, C_{o_j}) = \sigma(\sum_{k=0}^{C_{in}-1} w_g(C_{o_j}, k) \star i(N_i, k) + b_g(C_{o_j})) * \phi(\sum_{k=0}^{C_{in}-1} w_f(C_{o_j}, k) \star i(N_i, k) + b_f(C_{o_j})) \tag{3.1}$$

where ϕ is LeakyReLU function, σ is sigmoid function, thus the output values are in range $[0, 1]$, $\star$ is the valid 2D cross-correlation operator, N is batch size, C denotes a number of channels, H is a height of input planes in pixels, and W is width in pixels, $w(C_{o_j}, k)$ denotes the weight of j-th output channel corresponding k-th input channel, $i(N_i, k)$denotes the input of i-th batch corresponding k-th input channel, $b(C_{o_j})$ denotes the bias of j-th output channel.

3.3 Canny Edge Detection for Detail Enhancement

The inaccuracy part is usually concentrate in the coarse surface or drastic changed surface parts of the object. The corresponding part can be extracted separately via edge detector algorithms, like Canny Edge detector. Feed the edges to a special net for normal prediction might improve the accuracy further.

3.4 Image Guided normal inference

The normal inference can be guided by a RGB or gray-scale image, since the image is captured by passive method, it is fully dense comparing to depth map, which also has a complete view of the scene. The developed research about image feature extraction using CNN provide use a number of choice for this task. ResNet [7] is used in this paper as the backbone for grayscale image feature extraction.

3.5 Architecture

Based on the implementation mentioned above, we propose a Gated Convolutional Neural Network to perform guided normal inference. The architecture of trained network is shown in Figure 3.2. There are two stages in the downsampling and one stage in upsampling. First stage takes gray scale image as input then samples downward 3 times and extracts the features using 2D convolutional layers. The second stage takes 3D vertex as input then samples downward 3 times as well but extract the features using gated convolutional layers. Then concatenate two stages together and upsampling 3 times, two standard convolutional layers have been added at the end. The output normal map has the same size as the input 3d vertex map.

3.6 Add the light information

stores for every pixel the direction of incoming light. find a mapping H, such that $l_{in} = H(v, l_s)$ for pixel v, l_s is the position of light source, l_{in} is direction of incoming light of pixel v. Iterate all the pixels, we get the light direction map L.

Let I denotes image, N denotes normal map, L denotes light direction map. lambertian reflection

$$I = \rho N * L$$

where $*$ denotes scalar product, the albedo rho can be computed by

$$\rho = \frac{I}{N * L}$$

in NN, it is

$$\rho = conv2d(\frac{I}{N * L})$$

The corresponding loss term is

$$\|I - \rho(N * L)\|_{F_2}$$

in this case, the normal is supposed to be more crispy than the previous implement.

In a first step, we should compute initial albedos from the predicted normals and check if everything is correct. (Normals need to be in spatial space again, not in tangent space, and we need to use cameras extrinsics R and t from the data file to triangulate the vertex maps correctly)...

$$\rho = \frac{I}{N * L}$$

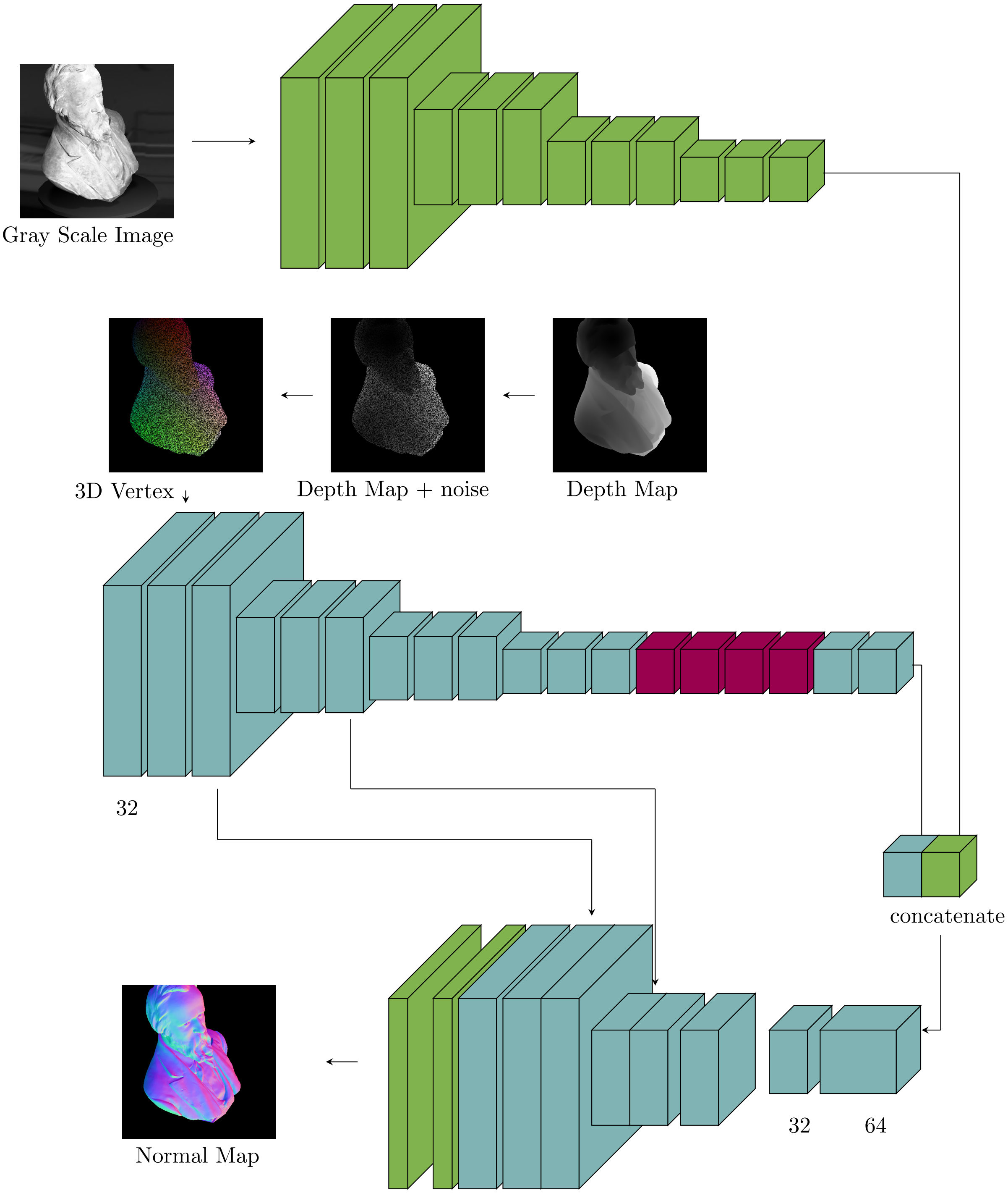


Figure 3.2: CNN Architecture

Chapter 4

Experiments

The model is trained with PyTorch 1.10.2, CUDA 10.2.89, GPU with single NVIDIA GEFORCE GTX 1080Ti.

4.1 Neighbor Input

4.2 Normal Neural Network

The first neural network that worked well for normal prediction in this master thesis is introduced in this section. It is simply named Normal Neural Network, or NNN. The input is 512x512x3 3D vertex matrix and output is 512x512x3 normal map. The architecture is based on UNet [17] with 3 times downsampling and 3 times upsampling. Gated convolution layer is used as a substitution for standard convolution layer in order to handle the sparse input. The channel size is 32 through the whole network unless the last one. For the training detail, this model is trained on the whole dataset with 500 items, with learning rate 0.001, penalty 1.4. The model architecture is shown in Figure 4.1

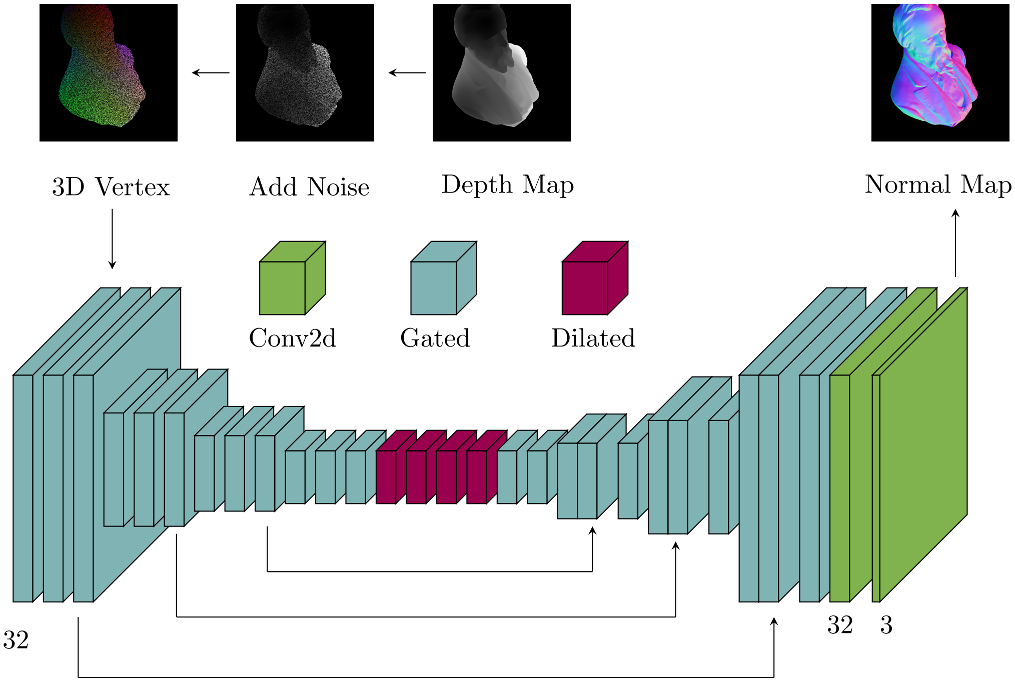


Figure 4.1: Basic Normal Neural Network model based on Gated Convolution layer and UNet architecture.

4.3 ResNet+Gated Convolution

4.4 Normalized Convolution

4.5 GaRes Model

Figure 4.5 shows the ground truth and predicted normal map of a Washington statue. In the predicted normal map, as shown on the right side, the relief on the side of stone chair is lack of sharpness compare to the ground truth on the left side.

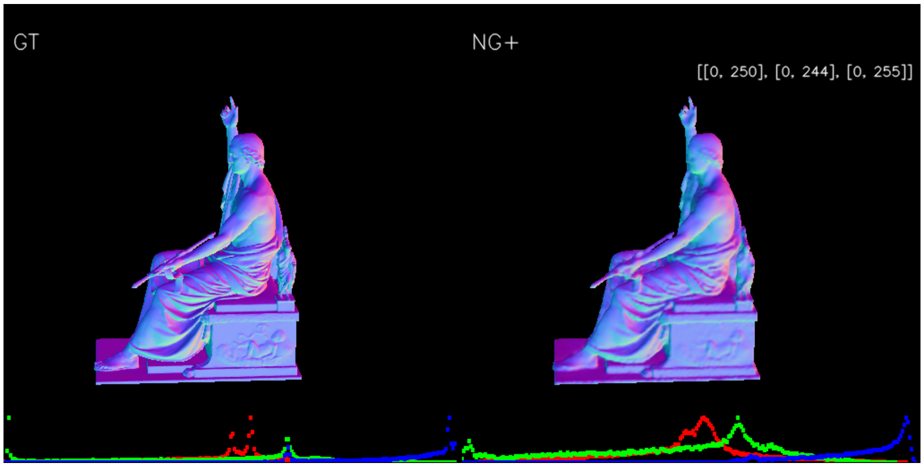


Figure 4.2: Left: Ground truth normal map; Right: Predicted normal map based on model ”NG”

To further visualize the error of predicted normal map, figure 4.5 shows the angle error of normal map. It is obvious to see, that the error goes higher in the coarse surface, like fingers, gown and relief. Oppositely, the error goes lower in the smooth surface, like the arm, face, and foot. The coarse surface are mainly the boundaries, or edges, which can be extract efficiently using edge detection algorithms, like Canny Edge detector. Figure 4.5 shows the detected edge of ground truth using Canny Edge Detector.

4.6 Detailed Gated ResNet Normal Neural Network

Use edge detector algorithms for further improvement.

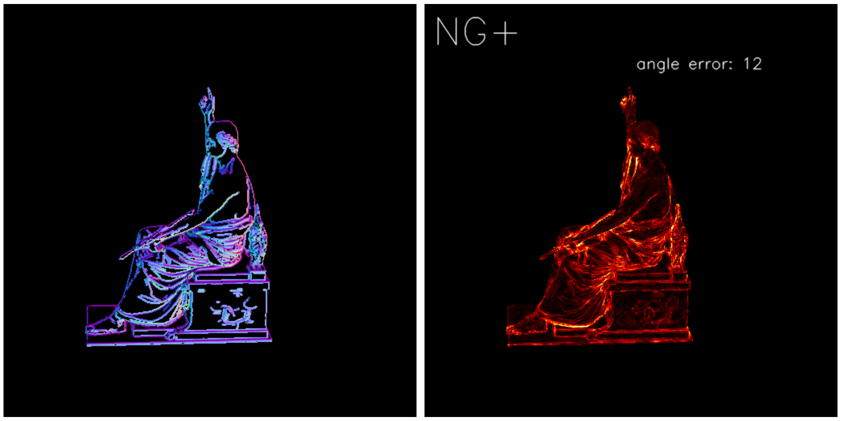


Figure 4.3: Left: Normal Map; Middle: Detected Edge of Normal Map using Canny Edge Detection Algorithm; Right: Error of predicted normal map.

Chapter 5

Discussion

5.1 Dataset Normalization

The vertices have been normalized before feed them into the model. The depth range of each scene is shown in Figure 5.1. The sizes of each training object are various, whereas it should be as an invariant value for the training model. Thus the normalization is required before feed training objects into the models. Figure 5.1 shows the fluctuation of extreme values and their ranges in 100 random training items. Table 5.1 gives the corresponding average values.

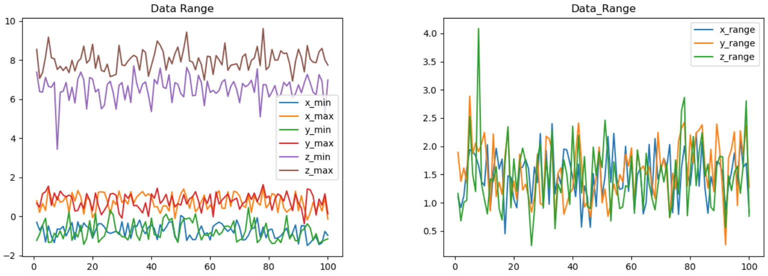


Figure 5.1: Left: Extreme value in 3 axis; Right: Vertex range in 3 axis

Axis	Range	Min	Max
X	1.48	-0.75	0.73
Y	1.56	-0.76	0.80
Z	1.47	6.53	8.00

Table 5.1: The ranges and extreme values of each axis. The extreme min and max values of both X axis and Y axis are close to -0.75 and 0.75 separately. The case for Z axis is 6.5 and 8.0 separately. However, the range of three axes are relatively similar, around 1.5.

For the normalization, first we translate the point to the original point as much as possible, then choose the range value of one axis as a scale factor, normalize it to a unit object.

$$U^a = (V^a - V^a_{min})/s \quad \text{for } a \text{ in } X, Y, Z \text{ axis} \tag{5.1}$$

where V^a_{min} denote the minimum value appeared in axis a, s is the range of an axis, which can be any one of X, Y, or Z axis.

Chapter 6

Formular

RGB image will be stored as gray value Scene using following equation:

$$gray : \frac{r + 2g + b}{4}$$

6.0.1 Normal from k neighbors

Given a point p locating on plane Π, calculate the normal n of plane Π.

First, find the nearest k neighbors $p_1, p_2, ..., p_k$ of point p using KNN-algorithms. The plane Π containing point p can be fitted using the neighbors of point p. Then the normal is available immediately.

Assume all the neighbors of point p are in plane $\Pi = ax + by + cz + d = 0$. Since we only need calculate the normal, thus with out loss of generation, we can set displacement $d = 0$. Then the normal $\mathbf{n} = (a, b, c)^T$.

Since all the neighbors of point p are located on plane Π, thus we have

$$P_{k\times 3} \cdot \mathbf{n}_{3\times 1} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

In order to avoid trivial solution, one more constraint should be added

$$\|\mathbf{n}_{3\times 1}\|_2^2 = 1$$

, which also let the normal to be a unit vector. In order to calculate a valid normal, 3 points are required at least. For the sake of robust, more points can be used to reduce the measuring error. In this case, the equation system is over-determined, which can be modeled as following optimization problem

$$\begin{aligned} \min \quad & \|P\mathbf{n}\|^2 \\ \text{s.t.} \quad & \|\mathbf{n}\|^2 = 1 \end{aligned} \tag{6.1}$$

Let the decomposition of $P = U\Sigma V^T$, The solution i.e. normal is the last column of V.

6.0.2 Normalized Convolution

Bibliography

[1] Yizhak Ben-Shabat, Michael Lindenbaum, and Anath Fischer. Nesti-net: Normal estimation for unstructured 3d point clouds using convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[2] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches, 2014.

[3] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network, 2014.

[4] Abdelrahman Eldesokey, Michael Felsberg, Karl Holmquist, and Mikael Persson. Uncertainty-aware cnns for depth completion: Uncertainty from beginning to end, 2020.

[5] Abdelrahman Eldesokey, Michael Felsberg, and Fahad Shahbaz Khan. Confidence propagation through CNNs for guided sparse depth regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(10):2423–2436, oct 2020.

[6] David F. Fouhey, Abhinav Gupta, and Martial Hebert. Data-driven 3d primitives for single image understanding. In *2013 IEEE International Conference on Computer Vision*, pages 3392–3399, 2013.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[8] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[9] S. Holzer, R. B. Rusu, M. Dixon, S. Gedikli, and N. Navab. Adaptive neighborhood selection for real-time surface normal estimation from organized point cloud data using integral images. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2684–2689, 2012.

[10] Jiashen Hua and Xiaojin Gong. A normalized convolutional neural network for guided sparse depth upsampling. In *Proceedings of the 27th*

International Joint Conference on Artificial Intelligence, IJCAI'18, page 2283–2290. AAAI Press, 2018.

[11] Hans Knutsson and Carl-Fredrik Westin. Normalized and difierential convolution methods for interpolation and filtering of incomplete and uncertain data. 01 1993.

[12] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks, 2016.

[13] Aaron van den Oord, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu. Conditional image generation with pixelcnn decoders, 2016.

[14] Fei Qi, Junyu Han, Pengjin Wang, Guangming Shi, and Fu Li. Structure guided fusion for depth map inpainting. *Pattern Recognition Letters*, 34(1):70–76, 2013. Extracting Semantics from Multi-Spectrum Video.

[15] Xiaojuan Qi, Renjie Liao, Zhengzhe Liu, Raquel Urtasun, and Jiaya Jia. Geonet: Geometric neural network for joint depth and surface normal estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[16] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement, 2018.

[17] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[18] Mingxing Tan, Ruoming Pang, and Quoc V. Le. Efficientdet: Scalable and efficient object detection. 2019.

[19] Na-Eun Yang, Yong-Gon Kim, and Rae-Hong Park. Depth hole filling using the depth distribution of neighboring regions of depth holes in the kinect sensor. In *2012 IEEE International Conference on Signal Processing, Communication and Computing (ICSPCC 2012)*, pages 658–661, 2012.

[20] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas Huang. Free-form image inpainting with gated convolution, 2018.

[21] Jun Zhou, Wei Jin, Mingjie Wang, Xiuping Liu, Zhiyang Li, and Zhaobin Liu. Fast and accurate normal estimation for point cloud via patch stitching, 2021.