Non Line of Sight Imaging

A deep learning approach to 3d point cloud reconstruction

Author: Jackrite To

Supervisor: Javier Grau, Markus Plack, Prof. Dr. Matthias B. Hullin

March 14, 2022

Contents

1 Introduction

1.1 Background on Non-line-of-sight imaging

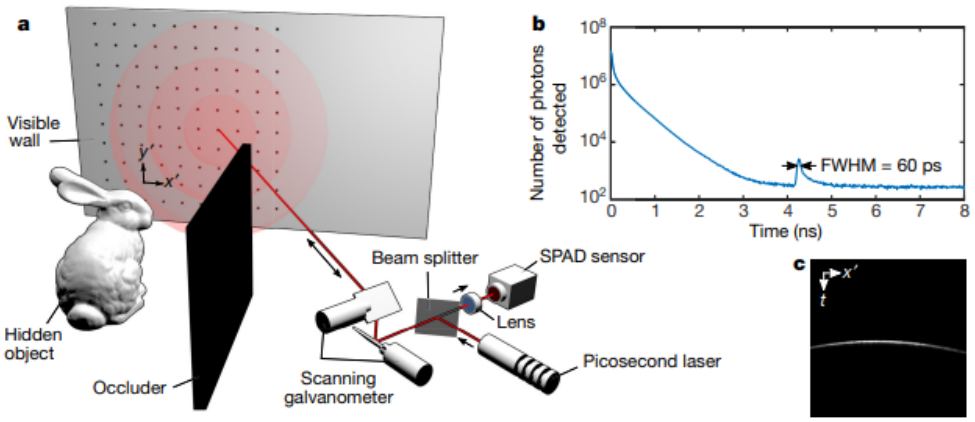


Figure 1: Non-line-of-sight setup
Picture from [6]

In Non-line-of-sight (NLOS) imaging, an object is hidden by an occluder and is not visible. It is facing a visible wall. A pulse of light is sent from a laser to the visible wall. It is reflected on the surface of the object. The light is reflected back from the object towards the wall. A sensor captures the number of photons emitted back from the wall to the sensor at successive time interval t. The wall acts as a camera which detects reflected light from the object. The result of the temporal response of the light measurement on several points on the wall is called a transient. More specifically, it is called a Time-Of-Flight (TOF) transient. From these transient measurements, the hidden object can be reconstructed in 3 dimensional space.

1.2 Goal of the experiment

From the transient, several methods exist to reconstruct the hidden 3d object such as in the approaches with diffuse mirrors [3] or with the light-cone transform [6]. Instead of implementing a specific reconstruction algorithm from transient to object, the approach in this lab was to implement a deep learning approach to reconstruct the 3d hidden object from transient. However, the TOF transient in NLOS are computationally intensive to process. In this lab, two reductions in dimensionality of the transient will be applied on the transient, one based on sinogram method and the other one based on Fermat path. 3 deep learning process with their respective transient representations are implemented: 3d transient (TOF transient), transient from fermat path, transient sinogram. A group of 3 students where each was responsible for a transient representation. These 3 different representations could then be compared in terms of efficiency and accuracy of 3d reconstruction. This report is focused mainly on the implementation of the deep learning approach from 3d transient.

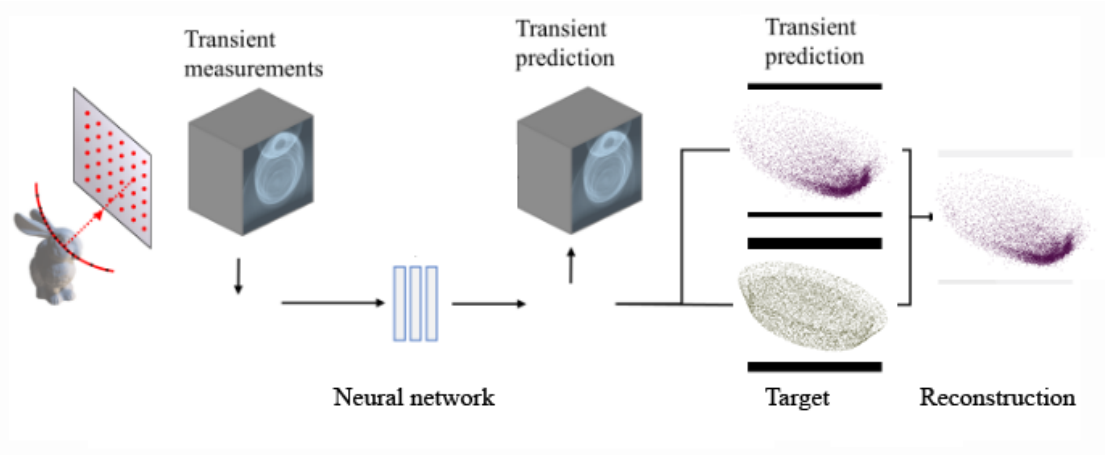


Figure 2: Deep learning pipeline
Picture modified from [8]

2 Data generation

2.1 Transient generation

Instead of a real physical non-line-of sight setup, the generation of transient measurements was done via an executable file called Tilib [4]. Tilib simulates the transient from 3d meshes object. As shown on figure 3 (left), its output is a transient of dimension (time*grid width*grid height). Tilib has multiple parameters such as the equivalent of temporal resolution, the interval in the temporal resolution, the grid size of the wall. Two different types of transients were generated with the following parameters:

A square of 32*32 points was sampled from the wall. The temporal response converted in distance that light travels started from 0.08. From this starting point, 256 intervals of 0.0096 meter were sampled. The resulting dimension of a transient file was (256*32*32).

A square of 64*64 points was sampled from the wall. The temporal response converted in distance that light travels started from 0.08. From this starting point, 512 intervals of 0.0048 meter were sampled. The resulting dimension of a transient file was (512*64*64).

The dimension (512*64*64) was used for the transient sinogram representation. The dimension (256*32*32) was used for the TOF transient and transient based on Fermat path.

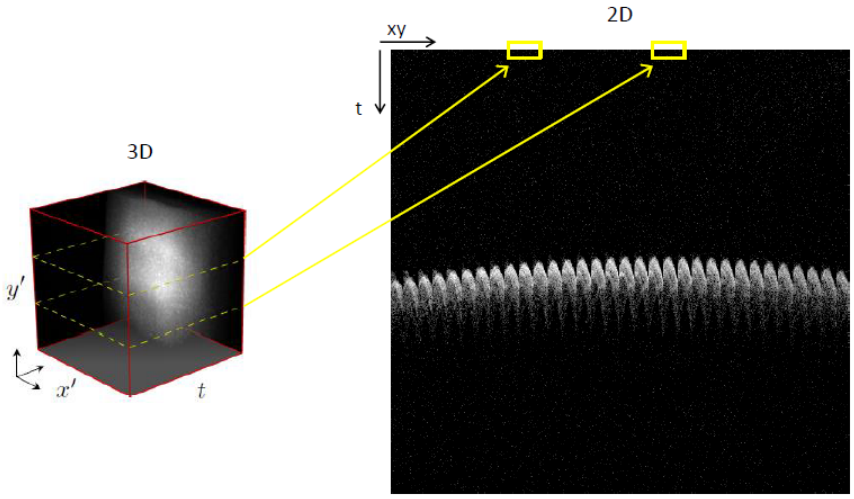


Figure 3: 3d transient

2.2 Preprocessing of objects

The mesh objects required for this experiments were collected from an archive of Princeton University (http://modelnet.cs.princeton.edu/). They have been pre-selected so that they did not contain too complicated independent faces such as flowers. All objects have been standardized so that they all could fit in a standardized box. Preprocessing consisted of repairing non-manifold edges and detecting inverted normals in the collection of objects. Non-manifold edges are edges that are shared by more than 2 faces. They have been repaired by splitting the vertices involved in an automatic manner on all objects collected. However, the objects provided by the Princeton archive did not always have normals of theirs faces consistently pointing outside of the objects, as shown on figure 4 (center). It can be solved by manually repairing each file individually as shown on figure 4 (right). No overall efficient solution have been found to address this problem on all files in a consistent and automatic manner. Therefore, these files were removed from the original data collection. With these 2 cleaning process 20 percent of objects were discarded from the original dataset and be fed to Tilib.

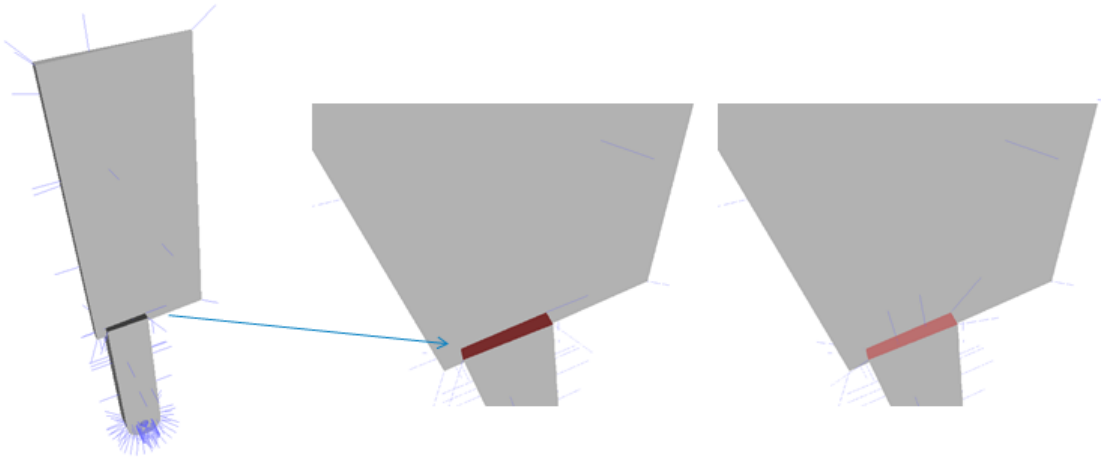


Figure 4: Manual correction of inverted normal of faces

2.3 Sinogram

The TOF transient from an hidden object is large and requires large computation. The idea behind transient sinogram was to sample only points along a circular path on the wall where the hidden object was reflected [5]. The goal was to find a sufficient small size of transient measurements to efficiently reconstruct the hidden object. Transient sinogram was made by sampling 360 points on a circular path on 512 temporal measurements of a 64*64 grid.

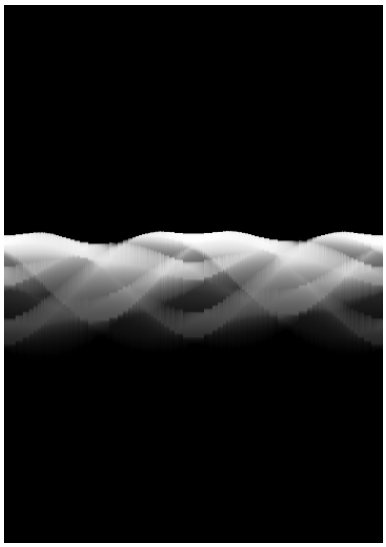
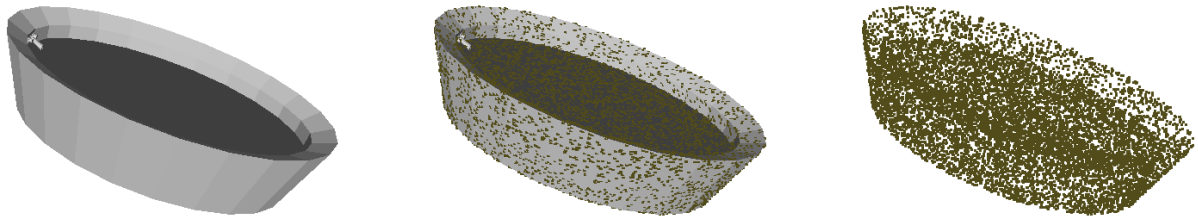


Figure 5: Transient sinogram

2.4 Pointclouds as targets for the neural network

Meshes of a 3d Object were converted to point cloud using Poisson sampling disk distribution. Point cloud is a list of coordinates of points in a 3-dimensional space. For each object, 10406 points were sampled. They would serve as targets for the neural network.



(a) mesh object

(b) object and point cloud superposed

(c) point cloud

Figure 6: Point cloud generation with Poisson disk sampling

3 Deep learning approach

3.1 U-net neural network architecture

U-net neural network architecture has been chosen for the learning process. In 2015, this architecture was the best convolutional network method for segmenting neuronal structures on electron microscopic pictures [7]. In U-net , the encoder part contracts the information and the decoder up-samples it. Encoder and decoder are made of 4 successive blocks. In the encoder, each block doubles the number of channels and halves the dimension of the feature map. The decoder does the opposite of the encoder, it doubles the dimension and halves the number of channels. Each block is composed of a double 3d convolution with a Relu activation between them. In the encoder part, max-pooling is responsible for halving the dimension of the input image. In decoder, up-convolution doubles the dimension of the input image.At the end of the decoder part, a 3d convolution with kernel 3, stride 3 and padding 1 so that the final dimension of the output is (86*11*11*3) and finally rearranged to the shape (10406*3). Thus, in the prediction output of the model, each row contained the coordinates of one point in 3 dimensions.

The total number of parameters of the U-net model has been modulated by the successive series of double convolution block: the higher the number of channels, the higher the number of parameters. As shown in the table 1, the list of channel number (20,40,60,120) gave a total number of parameters of 8 822943.

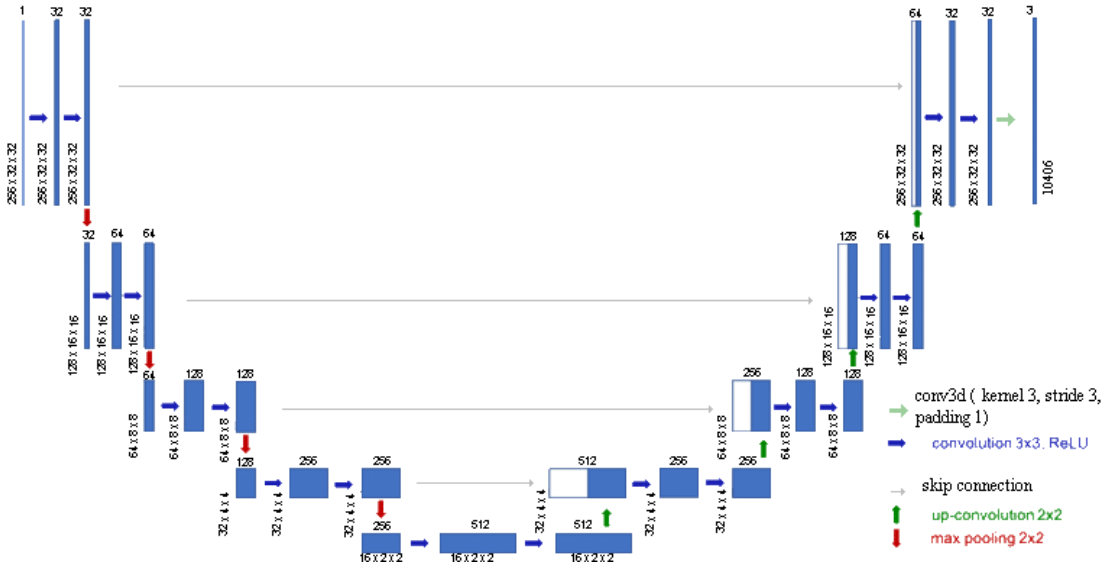


Figure 7: U-net architecture

Series of number of channels	N parameters
(8,16,32,64)	1418147
(20,40,60,120)	8822943
(32,64,128,256)	22580835

Table 1: Number of parameters of the U-net model

3.2 Chamfer distance loss

$$d_{CD}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2 + \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2^2$$

Figure 8: Chamfer distance formula

The Chamfer distance compares two point clouds S1 and S2. For each point in each point cloud, the algorithm finds the closest point in the other point cloud. The sum of all this closest point squared in each point cloud is the Chamfer distance [2].

3.3 Choice of hyper-parameters

Two datasets were gathered: One containing 12 objects and the other one 1308. A first training has been done on the 12 objects dataset to find the best hyper-parameters leading to an efficient reconstruction. The hyper-parameters found have then been used for the training and validation process in the 1308 objects dataset.

The optimizer chosen for the training was Adaptive Moment Estimation (Adam), it has been shown that this optimizer outperformed the other alternatives with U-net architecture [1]. Three different learning rates have been used 1e-4, 1e-5, 1e-6. The epochs used were 300 and 400. To choose the batch size, the number chosen was the maximum number of objects that is possible to fit in a batch until the GPU run out of memory. For the 1308 dataset, 26 objects were able to fit in a batch. To choose the best model according to its capacity, three different models as shown in table 1 with their respective capacity are trained. At convergence at the end of training, the model chosen among the three will be the one which had sufficient low number of parameters without preventing reaching a good loss reconstruction.

4 Results

4.1 Dataset of 12 objects

4.1.1 Training

N parameters	learning rate	batch size	epoch	loss	loss/N parameters
1418147	0.0001	12	400	1.8e-3	1.3e-9
8822943	0.0001	12	400	5.2e-4	5.9e-11
22580835	0.0001	12	400	3.32e-4	1.47e-11

Table 2: Training loss with hyper-parameters for 12 objects

In table 2, the model with 22580835 parameters achieved the lowest loss (3.32e-4), the worst model with 1418147 parameters the highest (1.8e-3). To compare these three models a simple heuristic was applied by calculating the ratio loss/number parameters. According to these ratios, the model with 8822943 parameters was close to the best loss reachable (3.32e-4) and further from the worst model with 1418147. Therefore, this model attained a relatively good loss with lesser computational lost. Moreover, this model could also be comparable with the model made by another student for processing the representation of transient based on Fermat path which contained 7763395 parameters.

On figure 9, the training loss for 12 objects with learning rate 1e-4 , 400 epochs reached convergence fast without noise. These hyper-parameters would then be applied for the training on the 1308 objects dataset.

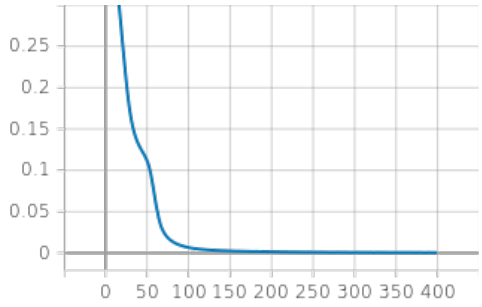


Figure 9: Training loss for 12 objects with learning rate 1e-4 , 400 epochs

4.1.2 Point cloud reconstruction

As shown on figure 10, for a loss of 5.2e-4, a point cloud prediction from the input 3d transient in training dataset was similar to the point cloud target. Most of the points in the prediction were in the same plan as the ones in the target. The shape of the target was preserved in the prediction. The viewer could understand the object of the prediction. A loss of 5.2e-4 was then a good threshold starting point for a good point cloud reconstruction of an object.

However, the points in the prediction were not evenly distributed as in the target. They were clustered on one side of the object. It is assumed that this side were facing the wall during the transient generation. Therefore, it was the visible part that the wall of the NLOS setup could "see".

(a) prediction (b) target (c) prediction and target superposed

Figure 10: Point cloud reconstruction from training set

4.2 Dataset of the 1308 dataset

4.2.1 Training and validation

The 1308 objects dataset was splitted in two sets: the training and validation set with a ratio of 0.2 (80 percent of object in training, 20 percent in validation set). On figures 11 and 12, the training and validation curves were very wavy and noisy. It meant that

the learning of the model had difficulty to reach convergence, it bounced back around the true convergence point. For the validation loss curve, the noise was worse and the curve was not stable. It was explained by the high learning rate and the presence of objects in the validation set very different in terms of shape to the objects in the training set. Nevertheless on table 3, at 300 epochs the training loss had reached around 5.65 e-4 which was similar to the loss on the 12 objects dataset. Therefore another training was done at a lower learning rate of 1e-5. As shown on figures 13, the noise in training and validation curves were severely reduced. This reduction confirmed the requirement of a lower learning rate. However, the price to do a lower learning rate was to reach convergence for longer epochs. On table 3, at 400 epochs, the training loss did not reach 5.2 e-4.

Model capacity	**learning rate**	**batch size**	**epoch**	**train loss**	**validation loss**
8822943	0.0001	26	300	5.65e-4	4.3e-3
8822943	0.00001	26	400	9.9e-4	2.3e-3

Table 3: Dataset of 1308 object: train and validation loss

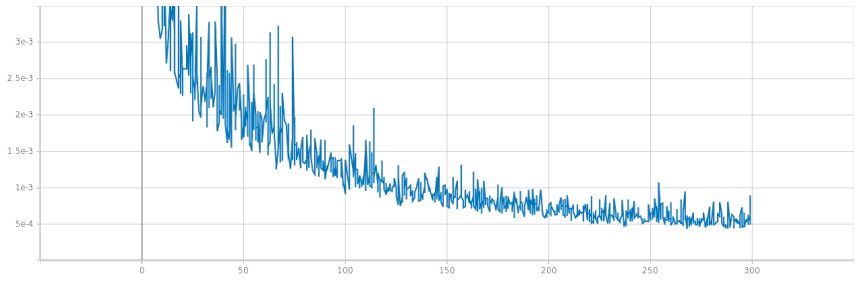


Figure 11: Training loss with learning rate 1e-4 and 300 epochs

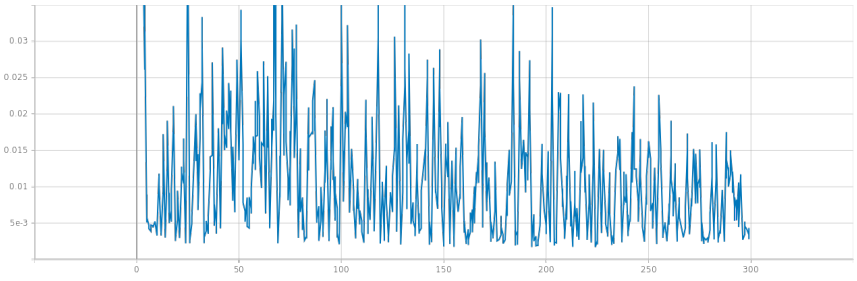


Figure 12: Validation loss with learning rate 1e-4 and 300 epochs

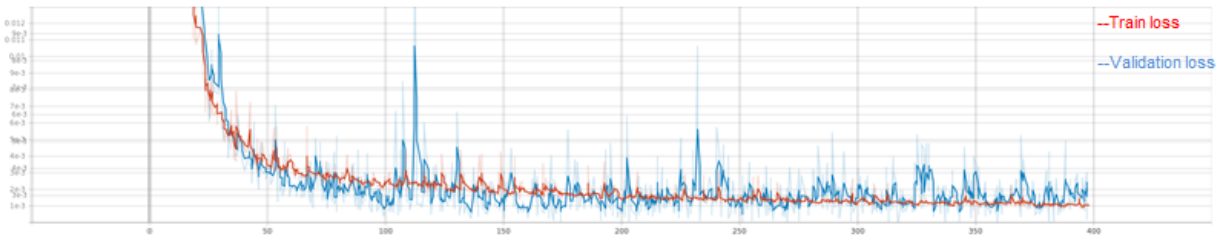


Figure 13: Training and validation loss with learning rate 1e-5 and 400 epochs

4.2.2 Point cloud reconstruction

On table 3, at 400 epochs, the validation loss was 2.3 e-3 which was a higher value than the previous threshold (5.2 e-4) for a good reconstruction. As shown on figure 14, the prediction point cloud compared to the target point cloud was bad. There were lot of point which were not even on the same plan as the target. The bottom of the chair of the target was not reconstructed in the prediction. Instead, it seemed that four feet of chair were wrongly reconstructed in the prediction. The phenomenon of clustering of point on one side of the prediction appeared also here. It can be concluded, that there were not enough objects in the dataset so that the model was able to correctly predict object that it had never seen.

(a) prediction (b) target (c) prediction and target superposed

Figure 14: Point cloud reconstruction from validation set

5 Future work

There were still room for improvements for these deep learning approaches. Since 20 percent of files were discarded because they might contain faulty inverted normals. An efficient algorithm must be found to detect normals of faces of the object that are not pointing consistently outwards and also in the same direction as of its neighbors.

The U-net model must be fine-tuned so that the points in the point cloud prediction would be evenly distributed. This can be done by modifying the Chamfer distance formula. If points predicted would form a cluster, their position would be penalized. Therefore the loss would be higher than the original one.

In NLOS, the lights from the object are reflected towards the wall. From the point of view of the wall, the viewer can only detect faces of the object facing the wall. All faces behind these first faces upfront are thus hidden and therefore not visible from the wall. What can be done to improve the deep learning model is to crop out the non-visible part of the target object. Only the visible part of the object will be kept, it will improve the loss during training.

A deep learning training with transient sinogram still remains to be done. Therefore, comparison of the three different deep learning approaches with their respective transient representation(3d, sinogram, fermat path) can then be achieved. The best model will be the one having the less number of parameters and input measurements for an efficient object reconstruction.

References

[1] Vatsala Anand, Sheifali Gupta, Deepika Koundal, Soumya Ranjan Nayak, Paolo Barsocchi, and Akash Kumar Bhoi. Modified u-net architecture for segmentation of skin lesion. *Sensors*, 22(3):867, 2022.

[2] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017.

[3] Felix Heide, Lei Xiao, Wolfgang Heidrich, and Matthias B Hullin. Diffuse mirrors: 3d reconstruction from diffuse indirect illumination using inexpensive time-of-flight sensors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3222–3229, 2014.

[4] Julian Iseringhausen and Matthias B Hullin. Non-line-of-sight reconstruction using efficient transient rendering. *ACM Transactions on Graphics (TOG)*, 39(1):1–14, 2020.

[5] Mariko Isogawa, Dorian Chan, Ye Yuan, Kris Kitani, and Matthew O'Toole. Efficient non-line-of-sight imaging from transient sinograms. In *European Conference on Computer Vision*, pages 193–208. Springer, 2020.

[6] Matthew O'Toole, David B Lindell, and Gordon Wetzstein. Confocal non-line-of-sight imaging based on the light-cone transform. *Nature*, 555(7696):338–341, 2018.

[7] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[8] Siyuan Shen, Zi Wang, Ping Liu, Zhengqing Pan, Ruiqian Li, Tian Gao, Shiying Li, and Jingyi Yu. Non-line-of-sight imaging via neural transient fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(7):2257–2268, 2021.