Improving Generalization Results for 3D Point Cloud Data Reconstruction From 2D Images

Jaineel Dalal
jdalal@stanford.edu

Rosie Yuyan Zou
rosiezou@stanford.edu

1 Abstract

Being able to reconstruct 3D models from 2D images is an important problem with a wide array of applications in Computer Vision and Robotics. In this project we investigate the application, limitations and generalization results of current state-of-the-art 3D reconstruction solutions. We also train with a different ground truth data pipeline and a novel point cloud consistency loss function that builds up on the existing work on Self-supervised Shape and Pose networks for Reconstructed Point Clouds in [1]. With this approach, we tackle the data problem that is quite common in 3D object reconstruction, by implementing a neural network architecture that can generate point clouds from a single real image. Our proposed approach replaces the data generation pipeline and the 3D loss function proposed in [1] with a custom ground truth pipeline and a novel 3D loss function that is significantly impacted by an object's geometric property. We were able to conclude that the current state-of-the-art 3D reconstruction network, which employs a purely self-supervised approach, does not perform well when evaluated against a ground truth point cloud data set with no uniform density. We also identified flaws in the 3D loss function design presented in the baseline. Lastly, we investigated the overall practicality of the baseline approach.

2 Prior Work

2.1 3D Reconstruction

Before deep learning, classic multi-view stereo algorithms relied on geometrical methods to formulate and understand the projections from 3D to 2D space [2]. Effective solutions either required multiple images captured using accurately calibrated cameras or depth images captured using expensive laser equipment, in tandem with a single camera to reconstruct 3D data. The classic stereo based techniques described in [3] for example, relied on matching features across images taken from slightly different angles which may not be practical or feasible in several situations. Effectively, it can be said that solving 3D reconstruction is a data problem where lack of good quality data can make it difficult to generate a good enough representation. With the advent of deep learning, single image based 3D object reconstruction methods have become increasingly popular. Early works on the same focused on full 3D supervision using 3D voxels, meshes or point clouds [4], [5], [6]. However such approaches still require large amounts of 3D data for training which can be hard and expensive to obtain. Some of the recent works [7], [8] have focused more on utilizing 2D multi-view images with color and object silhouettes for effective supervised training. However, this still involves multiple 2D views of the same 3D model along with the associated camera pose information at the training stage. Applying such restrictions in a practical setting could be a problem when such supervisory data is hard to obtain. In our implementation, we hope to use a semi-supervised approach which can work with 2D images and their corresponding point clouds directly in 3D space.

3 Dataset

3.1 Representation and Dataset selection

3D data required for training purposes can be represented in several formats such as depth images, point clouds, meshes and volumetric grids. We prefer point clouds for reconstruction since that preserves 3D and geometric information without much discretization [9]. Although, deep learning for point clouds comes with its own set of challenges such as high dimensionality and an unstructured problem [10], the field has been increasingly bolstered by the wide availability of public datasets for

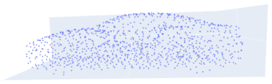
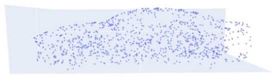
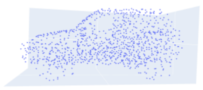
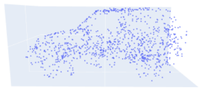
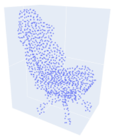
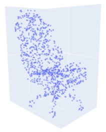
Category	CAD model	Uniform sampling (baseline)	Area based sampling (ours)
Car			
Car			
Chair			

Table 1: Different strategies for sampling 3D point clouds from a mesh model: Uniform sampling vs Area based sampling.

3D models such as ShapeNet [11], ModelNet [12], Kitti [13] and ScanNet [14]. For our use-case, we rely on synthetic datasets such as Shapenet [11] and ModelNet [12] since they consist of a single object in a 3D mesh and hence are easier to manipulate.

3.2 Strategy

3D mesh model from ShapeNet is sampled into 1024 points to generate a 3D pointcloud. We rely on two sets of point clouds for both training and evaluation - one generated by the baseline and one generated by our custom ground truth pipeline. The baseline approach performs random sampling for 16k points on the surface of the object followed by farthest point uniform sampling to obtain the final set of 1024 points. With each pointcloud containing a fixed number of points, this may overrepresent smaller vertices and underrepresent larger areas. Owing to this, we designed our custom groundtruth pipeline based on the work done by [15] and [16] for generating point clouds where points are sampled based on the surface area of each triangular vertex in a given 3D mesh. This results in better coverage for larger areas as evident from table 1. The input training images are also rendered from the mesh model with a single random view per object instance. All experiments are performed on two representative ShapeNet categories - Car and Chair. Following the training/test split recommendation in [17] and [18], we split the ShapeNetCore_v1 dataset for both categories roughly into two halves - 50% for training and 50% for testing. The number of models per category ranges from 6.7k to 8k in the overall dataset.

4 Approach

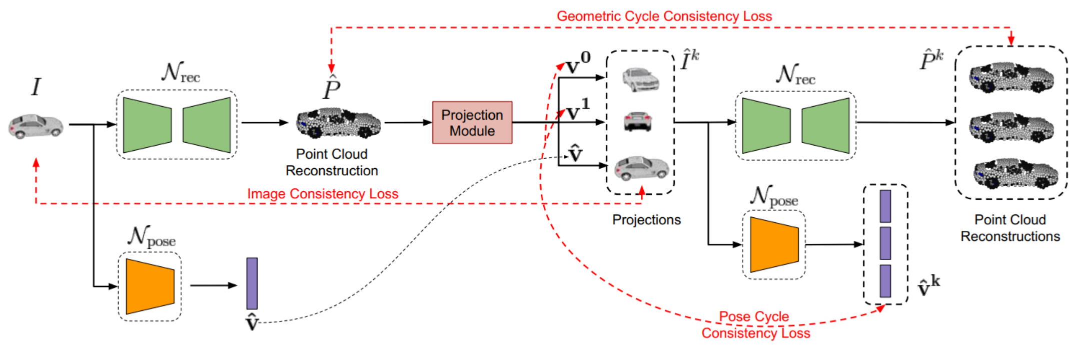
4.1 Baseline Model

Detailed studies of existing literature showed one common approach - an encoder-decoder network architecture design in conjunction with transfer learning. While our initial brainstorming phase identified 3D-PSRNet [19] and 3D-LMNet [20] as our baseline models, we concluded to use the model presented in [1] that builds upon the DIFFER model [17] which itself utilizes 3D-PSRNet [19] as its baseline. This paper tries to address a shortage of pose and angle information in the training data set which resonates very well with our original project motivation, owing to which we have decided to look into both the theoretical aspects as well as the implementation of this baseline model in order to come up with a design of our own.

4.2 Model Description

In our approach, we build on top of the self-supervised network design in [1] and introduce our own loss function, defined in 4.3. Note that the network was trained without the Pose Network due to logistical challenges outlined in 4.4.

Figure 1: Baseline Model Architecture Design [1]


Geometric Cycle Consistency Loss
I
$\mathcal{N}_{\text{rec}}$
$\hat{P}$
Point Cloud Reconstruction
Projection Module
$\mathbf{v}^0$
$\mathbf{v}^1$
$\hat{\mathbf{v}}$
$\hat{I}^k$
Projections
$\hat{P}^k$
Point Cloud Reconstructions
Image Consistency Loss
$\mathcal{N}_{\text{pose}}$
$\hat{\mathbf{v}}^k$
Pose Cycle Consistency Loss

4.3 Loss Functions

- **Baseline intuition**
  The overall baseline strategy consists of a two-fold network training process: one network for point cloud reconstruction ($\mathcal{N}_{rec}$), and one for pose estimation ($\mathcal{N}_{pose}$). Total loss function is a linear combination of both 2D and 3D discrepancy measures, with the influence of each being gated by regularization hyper-parameters. As demonstrated in the baseline, purely 2D-based loss functions could potentially lead to degenerate solutions where the paired planar 3D reconstruction could optimize for zero loss by continuing to generate the 2D reprojections from a constant viewpoint. [1] introduces a 3D-based Geometric Cycle Consistency Loss (GCC Loss) to mitigate this problem that's evaluated using the Chamfer distance which denotes the distance between two point clouds.

$$\mathcal{L}_{gcc} = \sum_{i=1}^{k} d_{Ch}(\hat{P}, \hat{P}^i)$$

  In our opinion, lack of ground truth and a self-supervised learning approach may not be an optimal way to reconstruct from a 2D image. In baseline, the Chamfer distance is calculated between a pair of point clouds in a pairwise manner and summed over all k reconstructions of a given image where the first reconstruction acts as pseudo ground truth. There are two key design flaws in this formulation: (1) Note that early-stage reconstructions are bound to be inaccurate. Without ground truth data, the discrepancies between two inaccurate reconstructions of the same image from different viewpoints make it hard for the network to learn useful parameters in early stages. (2) Structural properties of the point clouds are disregarded since all points are considered in the calculation of the Chamfer Distance. Intuitively, for a point cloud reconstruction to be qualitatively accurate, not every single point needs to be considered. Instead, it is more necessary for keypoints that outline the overall shape of the object to be accurate.

- **K Random Octant Loss (KRO Loss)**
  In order to address the geometric structure and degeneracy problem with point clouds, we propose a novel loss function that leverages both geometric and structural properties of 3D objects. For a given pair of (Y_i, G_i), where (Y_i) is the ground truth 3D point cloud of the i-th observation and G_i is the generated point cloud, perform the following steps on both Y_i and G_i:

  – Draw a positive integer k from $unif(1, max\_k)$ where max_k is a hyper parameter that can be tuned.

  – Get the volume occupied by the object, and divide each axis into k equal segments. Note that there is no need to calculate volume during computation. This step will give us k^3 equal-sized octants.

  – Find the geometric center of each octant and the point cloud closest to it. We call it the anchor point $a_j^{(i)}$

  – Find the geometric center of the object and the point cloud closest to it. We call it the center point $c^{(i)}$

Our objective is to minimize

$$\mathcal{L}_{kro} = \sum_{i=1}^{m} \mathcal{D}_{GT}^{(i)} - \mathcal{D}^{(i)}{}_{sim}\ , \textit{ where } \mathcal{D}^{i} = \sum_{j=1}^{k^3} \left\| a_j^{(i)} - c^{(i)} \right\|^2$$

- **Final Loss Function**
  The final loss function is a linear combination of component loss functions proposed in the baseline and the KRO Loss function proposed in 4.3. More detailed definitions of image loss ($\mathcal{L}_I$) and mask loss ($\mathcal{L}_M$) can be found in the baseline paper [1].

$$\mathcal{L} = \alpha(\mathcal{L}_I + \mathcal{L}_M) + \beta\mathcal{L}_{gcc} + \lambda\mathcal{L}_{kro}$$

4.4 Experiments

Baseline model was trained for roughly 400k iterations for each of the three object categories - car, chair and plane. Due to time and hardware constraints, we limited ourselves to training uptil 9k-18k iterations for two object categories - car and chair with and without KRO loss. We also ran into several data dependency issues in the baseline implementation and had to work our way around missing pose and mask data. With pose information missing, we switched off PoseNet which in turn also provided valuable data for ablation study since we could concentrate our efforts solely on the performance of the reconstruction network. Table 2 describes our experiments for the same. All experiments

Experiment number	Category	Num iterations	With/Without KRO loss
1	Cars	18k	Without
2	Cars	18k	With
3	Chairs	9k	Without
4	Chairs	9k	With

Table 2: Ablation study with number of training iterations and novel KRO loss function

involving KRO Loss were performed with and without our custom GT data. For experiments without GT data, the self-supervised baseline approach of "pseudo ground truth" reprojections was used instead. Additionally, we also evaluated the intermediate models against the baseline ground truth data as well as our version.

4.5 Observations

Observations made after running experiments 1 and 2 from table 2:

1. While evaluating point clouds against pseudo ground truth, KRO Loss generates numeric issues (NaN errors after epoch 1) without a multiplier suggesting that the network was becoming unstable during learning right after the first iteration. The similar 3D-based loss function GCC loss, does not have any such issues with an initial λ value of 10000. The numeric issue got resolved after multiplying KRO Loss with the same λ. Both losses show zero loss upto the 4th decimal place. However, multiplying the loss values by 1000 before printing and modifying the string format to display upto 12 decimals confirmed that as expected both loss values were indeed non-zero, just extremely small.
2. When evaluating point clouds against our ground truth, introducing KRO loss makes the overall loss values grow at an exponential pace, to a point where it triggers floating point value overflow and slows down the training process significantly. We encountered this problem while running Experiment 2, so we decided to halt the program at around $iters = 40$, complete Experiment 3 and forgo Experiment 4.

4.6 Results

- During training, while it appears that KRO loss has a slightly decreasing trend, as shown in Figure 5 in Appendix, the nominal value remains very high when evaluated against ground truth data. This makes sense because in a purely self-supervised network, early stages of the renderings will be highly inaccurate. Hence, evaluating an inaccurate rendering against another equally inaccurate rendering would yield seemingly high loss values. This would also explain why the initial λ for GCC loss was set to 10000 in the baseline. Simultaneously, this also proves that even with a high λ, 3D-based loss functions are not as useful without ground truth data.

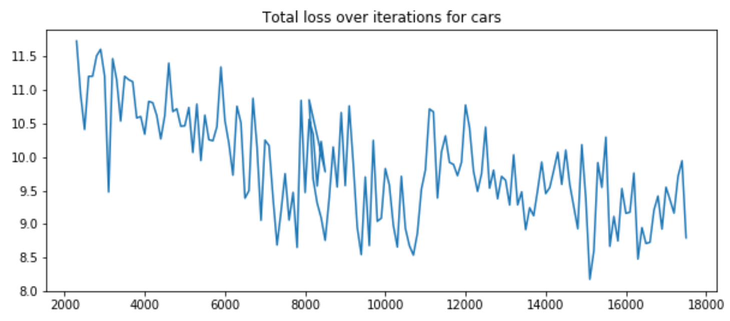


Figure 2: Total Loss Over Epochs for Cars, without KRO Loss

Model Type	Chamfer Distance	Fwd Distance	Bwd Distance	Epochs Trained	GT Type
Car	19.287	15.057	4.230	28k	Baseline
Car	72.508	66.881	5.627	28k	Ours
Chair	22.362	4.238	18.124	9k	Baseline
Chair	29.902	5.255	24.647	9k	Ours

Table 3: Evaluation Summary Table, showing both Chamfer Distance and its component values (forward and backward distances) evaluated against both the baseline and our own ground truth point clouds. We were able to train the car model for another 10k epochs after our initial experiment design.

- The above point is further supported by observing the output at training time. We plotted the intermediate point cloud reconstructions and observed that the reconstructed 3D point cloud data points are highly inaccurate and scaled down. This results in near-zero values in 3D loss calculations (both for GCC and KRO loss functions). However, symmetry loss is unaffected, due to the fact that it is a calculation based on the object itself and is not a property of the reconstruction. This leads us to believe that the timing of evaluating 3D-based loss functions should be more carefully studied.
- Evaluation of the intermediate baseline models yield worse metrics when compared with our custom ground truth point clouds, which we believe to be a more realistic density estimate compared to the near-uniform sampling provided by baseline.

4.7 Conclusions

- **3D-based loss functions are not useful in early stage training**
  This conclusion is supported by our observation of near-zero 3D geometric cycle consistency loss and KRO loss values in early iterations. We verified that the loss values were in fact non-zero, when we multiplied them by 1000 before printing them out to 12 decimals. This was further validated by plotting early-stage point cloud renderings. Even though the point cloud data was normalized and scaled, the rendered point clouds appear to be a 2D cluster with no defined geometry, with coordinates being as small as micrometers thereby pointing to a potential degeneracy during early stage training.
- **Practicality of purely self-supervised network** As shown in the evaluation of various model types, a purely self-supervised network performs poorly when evaluated against our non-uniformly distributed ground truth point cloud. Additionally, the training process requires a large number of epochs (upto 400k iterations with extremely small learning rate (5e-4) for training on a single object category. Given that the set of observable objects is virtually of an infinite size when geometry is considered, the baseline approach does not generalize well.

4.8 Future Work

Since we turned off Pose Network portion of the baseline model for our training, we are inclined to learn how the early-stage intermediate models will be affected with the same, with and without KRO Loss function (evaluated against both GT and Pseudo GT). This will allow us to fully understand the efficacy of the standalone Pose Network block. In addition to the current formulation of KRO Loss, we would also like to explore a second variation, where instead of partitioning the axes in k equal sizes, the entire volume is partitioned into k randomly sized volumes. This second variation could also have a regularization term that uses angle information, $\theta_{x,y,z}^{(i)}$, where $\theta^{(i)}$ represents the angle between the i-th anchor point and the center point for each random octant.

4.9 Contributions

Jaineel Dalal developed the end to end ground truth generation, data augmentation and the visualization pipeline . Rosie Zou developed the KRO loss function and evaluated performance with baseline. Both of them spent countless nights fixing code and missing data from the baseline implementation. Both contributed to reproducing results and writing proposal, milestone and the final report.

4.10 Acknowledgement

We would like to thank Jo - our project TA for his timely insights and help with our data reconstruction and training pipeline. We would also like to acknowledge our families for their continued support over the past few months while we focused on managing our work duties along with part-time studies. Thank you for your support and patience.

References

[1] K L Navaneet, Ansu Mathew, Shashank Kashyap, Wei-Chih Hung, Varun Jampani, and R Venkatesh Babu. From image collections to point clouds with self-supervised shape and pose networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[2] Xianfeng Han, Hamid Laga, and Mohammed Bennamoun. Image-based 3d object reconstruction: State-of-the-art and trends in the deep learning era. *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[3] Richard Hartley and Andrew Zisserman. *Two-View Geometry*, page 237–238. Cambridge University Press, 2 edition, 2004.

[4] Rohit Girdhar, David F. Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects, 2016.

[5] Haoqiang Fan, Hao Su, and Leonidas Guibas. A point set generation network for 3d object reconstruction from a single image, 2016.

[6] Christopher B. Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction, 2016.

[7] Rui Zhu, Hamed Kiani Galoogahi, Chaoyang Wang, and Simon Lucey. Rethinking reprojection: Closing the loop for pose-aware shapereconstruction from a single image, 2017.

[8] Xinchen Yan, Jimei Yang, Ersin Yumer, Yijie Guo, and Honglak Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision, 2017.

[9] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3d point clouds: A survey, 2020.

[10] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *CoRR*, abs/1612.00593, 2016.

[11] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[12] Kashi Venkatesh Vishwanath, Diwaker Gupta, Amin Vahdat, and Ken Yocum. Modelnet: Towards a datacenter emulation environment. In *2009 IEEE Ninth International Conference on Peer-to-Peer Computing*, pages 81–82. IEEE, 2009.

[13] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[14] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[15] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation, 2017.

[16] David Stutz and Andreas Geiger. Learning 3d shape completion from laser scan data with weak supervision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, 2018.

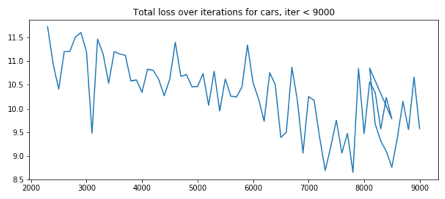
[17] K L Navaneet, Priyanka Mandikal, Varun Jampani, and R Venkatesh Babu. DIFFER: Moving beyond 3d reconstruction with differentiable feature rendering. In *CVPR Workshops*, 2019.

[18] Evangelos Kalogerakis, Melinos Averkiou, Subhransu Maji, and Siddhartha Chaudhuri. 3d shape segmentation with projective convolutional networks, 2017.

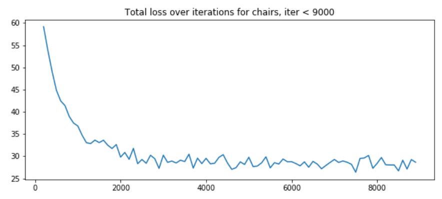
[19] Priyanka Mandikal, K L Navaneet, and R Venkatesh Babu. 3D-PSRNet: Part segmented 3D point cloud reconstruction from a single image. In *3D Reconstruction Meets Semantics Workshop (ECCVW)*, 2018.

[20] Priyanka Mandikal, KL Navaneet, Mayank Agarwal, and R Venkatesh Babu. 3d-lmnet: Latent embedding matching for accurate and diverse 3d point cloud reconstruction from a single image. *arXiv preprint arXiv:1807.07796*, 2018.

Appendix A

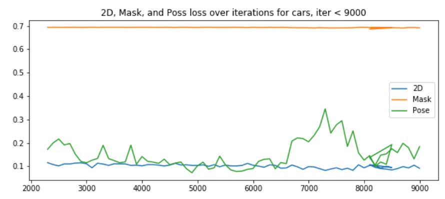


(a) Cars

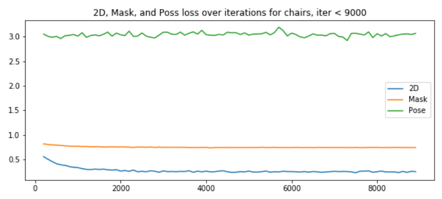


(b) Chairs

Figure 3: Total Losses over Epochs for Cars and Chairs, without KRO Loss



(a) Cars



(b) Chairs

Figure 4: Partial Losses over Epochs for Cars and Chairs, without KRO Loss

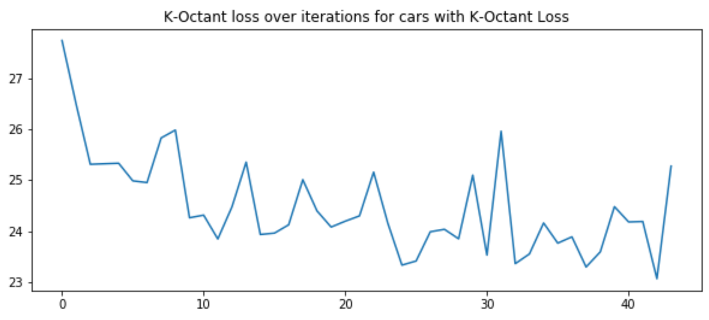


Figure 5: KRO Loss Over Epochs for Cars

Appendix B

Hyper-parameters used for training

- Reconstruction network architecture overview (structure only)
  - 4 CNN layers.
  - 4 FC layers.
  - Filter size: 3x33, stride of 2.
  - ReLU activation for all layers except the last FC layer which has Linear activation.
  - L2 regularizer.
- Training
  - gpu = 1
  - dataset = shapenet
  - 3d loss type = init model
  - categ = car, chair
  - loss = bce
  - affinity_loss (enabled)
  - optimise_pose
  - lambda_ae = 100
  - lambda_ae_mask = 100
  - lambda_mask_fwd = 1e-4
  - lambda_mask_bwd = 1e-4
  - lambda_3d = 10000
  - lambda_ae_pose = 1
  - lambda_mask_pose = 1
  - lambda_pose = 1
  - number of random projections = 4
  - iterations = 28000
  - k_random_octant_loss (enabled and disabled, for stratified testing)
  - use_gt_pcl (enabled and disabled, for stratified testing)
  - num observations = 50% of total available observations
  - Optimizer: Adam Optimizer with 0.0005 learning rate
  - Optimization algorithm: Mini batch gradient descent with batch size of 2
- Testing
  - num points in point cloud = 1024
  - num observations = 50% of total available observations
  - Optimizer for computing Chamfer distance: Adam Optimizer with 0.001 learning rate
  - eval metric: Chamfer distance