ImgAdaPoinTr: Improving Point Cloud Completion via Images and Segmentation

Vaagn Chopuryan[1][a], Mikhail Kuznetsov[1,3][b], Vasilii Latonov[1][c] and Natalia Semenova[1,2][d]
[1]*Sberbank PJSC, 19 Vavilova St., Moscow 117312, Russia*
[2]*AIRI, Nizhniy Susalnyy Lane 5, Moscow 105064, Russia*
[3]*National Research Nuclear University MEPhI (Moscow Engineering Physics Institute), 31 Kashirskoe shosse, 31, Moscow 115409, Russia*

Keywords: Point Cloud Completion, Transformers, View-Guided, Segmentation.

Abstract: Point cloud completion is an essential task consisting of inferring and filling in missing parts of a 3D point cloud representation. In this paper, we present an ImgAdaPoinTr model, which extends the original Transformer encoder-decoder architecture by accurately incorporating visual information. Besides, we assumed using segmentation of 3D objects as a part of the pipeline due to acquiring an additional increase in performance. We also introduce the novel ImgPCN dataset generated by our rendering tool. The results show that our approach outperforms AdaPoinTr by average 2.9% and 10.3% in terms of Chamfer-Distance L1 and L2 metrics, respectively. The code and dataset are available via the link https://github.com/ImgAdaPoinTr.

1 INTRODUCTION

A point cloud is a collection of data points in a three-dimensional coordinate system, representing the external surface of an object or a space, often used in 3D modeling, computer graphics, and computer vision. The point cloud is a widely used data type that represents 3D models (Berger et al., 2016) and could be converted to the polygon mesh. The 3D scanning devices like LiDARs and RGB-D scanners, used for point cloud acquisition, have recently become more accessible (Bi et al., 2021). The amount of these devices applications increased (Raj et al., 2020). For example, point clouds are used in real-time motion capture (Li et al., 2022). As far as the clouds of points produced by 3D scanning devices suffers from incompleteness and inherent sparsity, the researchers and developers face point clouds processing problems (Zeng et al., 2022). These disadvantages caused by occlusions, device technical limitations, scanning angles and internal algorithms weaknesses. Moreover, 3D scanning device misses some regions of the scanned object due to the object's complicated shape.

[a] https://orcid.org/0009-0007-8205-6591
[b] https://orcid.org/0009-0006-1315-7711
[c] https://orcid.org/0000-0002-7810-8033
[d] https://orcid.org/0000-0003-4189-5739

Point clouds have numerous important applications, especially where information from 3D scanning sensors is integrated with data from other sensors, such as standard RGB cameras. For instance, self-driving cars use LiDAR in conjunction with cameras to enhance motion tracking and object recognition (Abbasi et al., 2022; Shen et al., 2021). In the field of robotics, data frequently originates from a variety of sensor types. Therefore, adopting a sensor fusion approach for point cloud completion is an essential and beneficial strategy.

The learning-based approaches using different data types for pipeline input are called multi-modal or cross-modal. The application of multimodal models in different areas of deep learning, including 3D data processing (Nichol et al., 2022; Poole et al., 2022) has shown impressive results in various tasks such as Zero-shot classification (Radford et al., 2021), generation from textual description (Ramesh et al., 2021) and others. For instance, in the paper XMFnet (Aiello et al., 2022) the information from the two modalities in a localized latent space was united for point cloud completion.

A view-guided solution for the task of point cloud completion is presented in the work (Zhang et al., 2021). The authors fuse information through a modality transfer technique. This technique involves the concatenation of input point clouds and a convolu-

Chopuryan, V., Kuznetsov, M., Latonov, V. and Semenova, N.
ImgAdaPoinTr: Improving Point Cloud Completion via Images and Segmentation.
DOI: 10.5220/0012398100003660

In *Proceedings of the 19th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2024) - Volume 4: VISAPP*, pages 47-58
ISBN: 978-989-758-679-8; ISSN: 2184-4321

tional network, and a part refinement stage, where outputs from convolutional networks utilized in the modality transfer are used. However, in this approach, the 3D models from the test set, were also presented in the training sample, which leads a high probability of data leakage and ambiguous results.

In this paper we introduce a novel architecture for point cloud completion based on the AdaPoinTr (Yu et al., 2023) architecture. The new approach incorporates AdaPoinTr pipeline and features derived from image rendered from 3D-models. The primary contributions of this work are the following:

- We propose the ImgAdaPoinTr model for the point cloud completion task. We integrated a Multihead-Attention layers to facilitate a seamless fusion of information from the image encoder and the Geometry-aware Transformer Encoder. The best performance was achieved by using our custom Cycle Loss.
- Our experiments show that using the segmentation module as a part of the pipeline is acquiring an additional increase in performance for specific classes. We propose the SegEncAdaPoinTr model and describe the conditions.
- Also, we present a rendering toolkit[1] supporting GPU parallel computing by which we obtained the ImgPCN dataset for this work.

2 RELATED WORK

Existing learning-based point cloud completion methods can be divided into following types:

2.1 Point-Based Methods

Point-based methods have been explored for point cloud completion, processing point clouds without converting them into other representations. PointNet and PointNet++ (Qi et al., 2017a; Qi et al., 2017b) are among the works in this area, addressing the direct processing of point clouds using symmetric functions to handle unordered point sets. FoldingNet (Yang et al., 2018) employs an autoencoder structure to transform 2D grids into 3D structures, facilitating the generation of 3D point clouds from encoded features. PoinTr (Yu et al., 2021) uses transformers to handle the irregularity and permutation invariance of point clouds.

A limitation of FoldingNet is its use of the codeword. The method constructs a 2D $N \times N$ grid and appends a codeword to each point within this grid. Subsequently, this vector is projected into a 3D space, followed by precise spatial positioning adjustments. The codeword poses a notable constraint, as it relies on a singular global vector to reconstruct all points, with a near-total lack of localized information.

PCN (Yuan et al., 2018) extends this approach. While it also uses a codeword, which is one for all points, it serves to produce a coarse point cloud. For each point in this initial cloud, the FoldingNet technique is applied, considering both the point's coordinates and the codeword. However, even with the regulation of coarse points using the EMD loss, but codeword doesn't change. In PSN, FoldingNet decoder operates under more advantageous conditions, focusing on reconstructing local regions in relation to the coarse points.

SnowflakeNet (Xiang et al., 2021) introduces a unique perspective, modeling the generation of point clouds akin to snowflake-like growth. This approach is based on specific points in 3D space, where point clouds are progressively generated from designated parent points by splitting parent point features through deconvolution.

SeedFormer (Zhou et al., 2022) presents a novel shape representation termed "Patch Seeds" for point clouds. Additionally, it incorporates an innovative Upsample Transformer during the generation process.

AdaPoinTr (Yu et al., 2023) is a model that employs the Transformer (Vaswani et al., 2017). Its primary objective is to harness the robust sequence-to-sequence generation capabilities of the Transformer architecture for point cloud completion tasks. Initially, coarse points are derived from incomplete data, and after processing through the Transformer, they obtain a more comprehensive set of coarse points for the entire shape. Subsequently, a rebuild head is utilized to generate a refined point cloud.

2.2 Voxel-Based Methods

Voxel-based methods have been investigated for harnessing the spatial locality inherent in point clouds. Typically, these methods involve voxelizing 3D data into occupancy grids or distance fields, which then allows for the application of convolutional networks (Dai et al., 2017; Han et al., 2017). One limitation of this approach is the memory consumption of 3D data processing. Point-Voxel CNN (Liu et al., 2019b) and Relation-Shape CNN (Liu et al., 2019a) utilize convolution operations on voxelized data, targeting both local spatial details and broader dependencies within point cloud data. With increasing resolution, there is a corresponding rise in computational demands. To

[1] https://github.com/Pytorch3DGenRenderer

address this, GRNet (Xie et al., 2020) suggests the use of gridding operations and 3D CNNs for an initial coarse completion, followed by refinement processes to produce detailed structures.

2.3 View-Guided Models

View-guided models, such as ViPC (Zhang et al., 2021) and XMFNet (Aiello et al., 2022), integrate point clouds and images for 3D reconstruction. ViPC transforms 2D images into coarse point cloud, refines details, and fuses them into a comprehensive 3D representation. In contrast, XMFNet merges features from partial point clouds and images using graph-convolutional layers and attention mechanisms, reconstructing the missing parts. Notably, XMFNet's training process and its efficacy have been validated on the ShapeNet-ViPC dataset in various settings.

2.4 Generative Models

Generative models have been explored in the context of point cloud completion. PU-GAN (Li et al., 2019) and PU-Net (Yu et al., 2018) employ generative adversarial networks (GANs) and upsampling networks, respectively, to produce dense point clouds. Their objective is to ensure that the completed structures exhibit geometric consistency and detail. RL-GAN-Net (Sarmad et al., 2019) incorporates a reinforcement learning agent with a GAN network, with the goal of facilitating real-time point cloud shape completion. A common limitation observed in these models is the stability of their approaches, with instances of instability reported.

Other approaches, while potentially valuable in their own contexts, are not discussed here due to their limited relevance to our current work.

3 METHOD

3.1 Datasets

The ShapeNet-ViPC dataset, presented in the ViPC paper (Zhang et al., 2021), stands as the singular dataset for point cloud completion that incorporates images. The original authors, however, did not release the code for its generation. Addressing this absence, we introduce the ImgPCN dataset, which merges point cloud data with corresponding rendered images, targeting a multimodal strategy for point cloud completion and reconstruction tasks. For its creation, we employed our rendering tool, which will be detailed subsequently. For each 3D model, 24 views were produced at 15-degree intervals around a 35-degree elevation meridian in the ShapeNet-ViPC dataset manner.

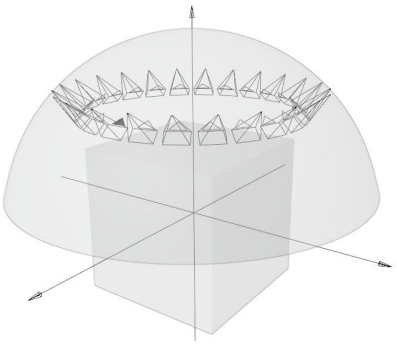

Figure 1: Camera points trajectory: a circle parallel to the sphere equator. The cube is a bounding box. Sphere is calculated so each camera captures bounding box with desired clearance.

3.2 Rendering Tool

Our rendering tool is built upon the PyTorch3D library (Johnson et al., 2020). Each 3D model's render is determined by a specific set of parameters, provided in respective configuration file.

For each rendered image the following rules are satisfied. First, the virtual camera focal axis crosses the center of model's bounding box. Second, the distance between camera and bounding box center (let us denote it by R) is calculated in such a way so the main bounding box diagonal is completely seen from any view angle.

The camera points of view lay on the specified trajectory, which lays in the sphere of R radius. This trajectory and the number of render points are set via the tool interface. We used a circle parallel to the sphere equator for dataset (see Figure 1).

The pipeline of our tool can be described as follows:

- Configuration file for each 3D model is generated. This process can be automated.
- Subsequently, a master configuration file for the overall generation is created by specifying the path to the directory containing the individual rendering configurations.
- An optimal distance R is calculated, ensuring its complete fit within the image frame. The formula 1 describes the calculation method.
- Optionally, there is a provision to parallelize the rendering process across multiple GPUs.

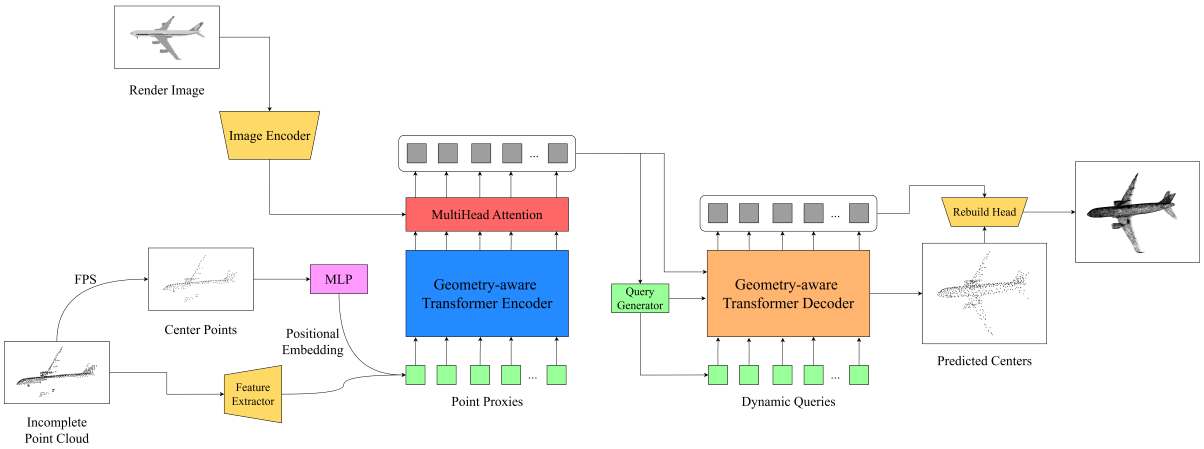

Figure 2: An extended pipeline scheme. Input partial point cloud is first downsampled to derive center points. Local features around these center points are extracted using a lightweight DGCNN. Simultaneously, the Image Encoder processes the visual data to extract image features. These features are fused with the local features extracted from the point cloud, incorporating the spatial intricacies of the visual data. With the addition of positional embedding to the fused features, they are channeled into a Geometry-aware Transformer Encoder-Decoder architecture. Through multi-head attention mechanisms, this structure predicts point proxies for the missing regions of the point cloud. Finally, the Rebuild Head refines these point proxies, completing the point cloud in a coarse-to-fine manner.

Here we describe how optimal distance R is calculated:

$$R = \begin{cases} \frac{df}{wp}, & d > d_{min} \\ 0.68, & d \leq d_{min} \end{cases} \quad (1)$$

where d is a bounding box main diagonal, f is a camera focal length, w is an image width pixels number (here we suppose that desired image is squared) and p is a width portion that defines a clearance between the render border and model image. The focal length is calculated by formula: $f = \frac{w/2}{\tan(\varphi/2)}$, where φ is a camera field of view angle. The threshold $d_{min} = 0.78$ was obtained empirically.

3.3 Architecture Overview

The architecture proposed in this work is based on the AdaPoinTr model, which has demonstrated state-of-the-art results in point cloud completion by employing a Transformer-based encoder-decoder mechanism and dynamic query generation. While maintaining the core components of AdaPoinTr, novel elements are introduced. In particular, an Image Encoder and an additional layer in the Transformer architecture were added. The additional layer's aim is to integrate multimodal data and enhance the model's capacity to generate comprehensive point clouds.

The following sections provide a detailed overview of the architecture, highlighting both the inherited components from AdaPoinTr and our novel contributions.

Preprocessing and Local Feature Extraction. The initial phase involves the input partial point cloud preprocessing. The input point cloud is downsampled and specific center points are obtained. This step is required to avoid big time and RAM resources consumption. Subsequently, a lightweight Deep Graph Convolutional Neural Network (DGCNN) (Wang et al., 2019) is used to extract local features around center points, providing a foundational feature representation for subsequent processing stages.

Point Proxies and Positional Embedding. The extracted local features, in conjunction with positional embedding, are transformed into a sequence of point proxies. These proxies are a compact representation of local regions within the point clouds, encapsulating both spatial and feature information in a format amenable to processing via the subsequent Transformer architecture.

Integration of Image Encoder. In our novel contribution, an Image Encoder is introduced to extract pertinent information from rendered images. This encoder processes the renders, extracting features that encapsulate texture and lighting information, which are then integrated into the point cloud completion process.

Geometry-Aware Transformer Encoder-Decoder Architecture - Encoder. The Transformer encoder processes the sequence of point proxies, utilizing self-attention mechanisms to capture dependencies within the data, both at local and global scales.

Additional Layer. A novel layer, which employs Multihead Attention, is introduced to fuse the output

of the Transformer encoder with the output of the Image Encoder. This layer is pivotal in integrating information from the point cloud and image data, ensuring that the subsequent decoding process can leverage features from both modalities.

Dynamic Queries. Dynamic queries are generated by the model, which are utilized by the Transformer decoder to infer missing elements based on learnable pairwise interactions among features of the input point cloud and queries.

Decoder. The Transformer decoder, utilizing the encoded features and dynamic queries, predicts point proxies for the missing parts of the point cloud.

Point Cloud Completion and Refinement. Predicted Centers: The predicted point proxies are utilized to infer centers for the missing parts of the point cloud, providing a coarse completion.

Rebuild Head. A Multi-Layer Perceptron (MLP) and a Rebuild Head are employed to further complete the point cloud based on the predicted point proxies, ensuring a coarse-to-fine completion strategy.

Refinement. Subsequent to the initial completion, the model employs a series of refinement modules to further refine the final prediction of the point cloud, enhancing the accuracy and detail of the completed point cloud.

The completed pipeline scheme is depicted on Figure 2. In summary, while our architecture retains the robustness of the AdaPoinTr model, the introduction of the Image Encoder and the additional Multihead Attention layer represents our primary novel contributions, facilitating the integration of multimodal data and potentially enhancing the model's capacity to generate detailed and accurate point cloud completions. Further experimental validation, detailed in subsequent sections, elucidates the impact and efficacy of these novel components within the point cloud completion task.

3.4 Loss Function

We introduce a composite loss function based on the terms used in AdaPoinTr approach. Let us introduce terms used for loss function composition. We denote by C the predicted local centers set. More we denote by P the set of points of complete point cloud and by G the ground truth point cloud. Therefore the loss functions are defined by the formulas:

$$J_0 = \frac{1}{|C|}\sum_{c\in C}\min_{g\in G}\|c-g\| + \frac{1}{|G|}\sum_{g\in G}\min_{c\in C}\|g-c\| \quad (2)$$

$$J_1 = \frac{1}{|P|}\sum_{p\in P}\min_{g\in G}\|p-g\| + \frac{1}{|G|}\sum_{g\in G}\min_{p\in P}\|g-p\| \quad (3)$$

$$J_{denoise} = \frac{1}{|\hat{P}_i|}\sum_{c\in\hat{P}_i}\min_{g\in G^{gt}_{ci}}\|c-g\| + \\ + \frac{1}{|G^{gt}_{ci}|}\sum_{g\in G^{gt}_{ci}}\min_{c\in\hat{P}_i}\|g-c\| \quad (4)$$

where J_0, J_1, and $J_{denoise}$ represent the Chamfer Distance loss (CD) for local centers, complete point clouds, and the denoising loss, respectively. $\hat{P}_i$ and G^{gt}_{ci} represent the predicted and ground-truth local shapes, respectively.

We propose a special function λ_{coarse}, which is inspired by the Cycle Learning Rate Scheduler (Smith, 2017). We name this function Cycle Loss. The primary objective behind this function is to gradually diminish the influence of coarse point clouds during training as the model progressively learns superior representations. This function is used as the J_0 term weight in the composite loss function.

Given parameters:

- s — step size, defines how often the λ_{coarse} coefficient should peak;
- c_{min} — minimal λ_{coarse} value;
- c_{max} — maximal λ_{coarse} value;
- γ — exponential scaling factor for the learning rate;
- n — the epoch index (epochs are enumerated with natural numbers);

The total loss is computed as follows:

$$cycle = \left\lfloor 1 + \frac{L}{2s} \right\rfloor \quad (5)$$

$$x = \left| \frac{L}{s} - 2\cdot cycle + 1 \right| \quad (6)$$

$$\lambda_{cr}(n) = c_{min} + (c_{max} - c_{min})\max(0,(1-x))\gamma^n \quad (7)$$

where $cycle$ computes the current cycle count, x calculates the absolute position within the current cycle, and λ_{cr} adjusts the impact of the coarse point cloud during training.

The total loss function is formulated as:

$$J_{PC} = \lambda_{cr}J_0 + J_1 + \lambda J_{denoise} \quad (8)$$

4 EXPERIMENTAL RESULTS

4.1 Experimental Settings

In this section we provide description, evaluation and analysis of performed experiments. All experiments

Table 1: Result on the PCN and ImgPCN(12 views) datasets. We use the CD-L1 (multiplied by 1000) to compare ImgAdaPoinTr with other methods. * Denotes the top result within its experimental group, surpassing the value in AdaPoinTr.

Name	AVG	Airplane	Cabinet	Car	Chair	Lamp	Sofa	Table	Boat	F@1%
FoldingNet	14.310	9.490	15.800	12.610	15.550	16.410	15.970	13.650	14.99	0.322
PCN	9.640	5.500	22.700	10.630	8.700	11.000	11.340	11.680	8.590	0.695
GRNet	8.830	6.450	10.370	9.450	9.410	7.960	10.510	8.440	8.044	0.708
SnowFlake	7.210	4.290	9.160	8.080	7.890	6.070	9.230	6.550	6.400	-
SeedFormer	6.740	3.850	9.050	8.060	7.060	5.210	8.850	6.050	5.850	-
AdaPoinTr	6.528	3.681	8.823	7.476	6.850	5.478	8.353	5.801	5.763	0.845
SegEncAdaPoinTr	6.569	*3.544	8.893	**7.404**	6.939	*5.189	8.544	6.179	5.862	*0.847
ImgAdaPoinTr	6.347	3.515	**8.540**	*7.425	6.677	**5.065**	**8.082**	5.714	5.756	0.857
ImgEncSegDecAPTr	**6.339**	**3.461**	8.646	*7.437	**6.616**	**5.065**	8.140	**5.685**	**5.660**	**0.859**

were carried out on the PCN and ImgPCN datasets. Both datasets consists of 30974 models from 8 categories: airplane, cabinet, car, chair, lamp, sofa, table, and watercraft. The complete point clouds are created by uniformly sampling 16384 points from the mesh surfaces, while the partial point clouds originate from back-projecting 2.5D depth images into a 3D space. A defining distinction between PCN and ImgPCN lies in the images procured through rendering aligned with meshes from ShapeNetCoreV1. Each object is rendered from 24 unique viewpoints, producing images with a resolution of 224x224 pixels.

Our approach adopted 256 points as input to the transformer-encoder, and 512 points for the transformer-decoder, with the number of queries being set at 512. In both the encoder and decoder, we employed a dimension of 384, with 6 heads, and a K value of 8. The transformer-encoder has a depth of 6, while the transformer-decoder delves deeper with a depth of 8. As our choice of image encoder, we opted for ResNet18. Although we experimented with alternatives like ResNet50 and ConvNext, as depicted in Table 3, ResNet18 emerged superior. The attention mechanism mirrored that of the XMFNet (Aiello et al., 2022), incorporating both self and cross attentions spanning two layers.

4.2 Main Result

Throughout our experiments, we converged on the ImgAdaPoinTr architecture, which emerged as the top-performing solution. ImgAdaPoinTr: ResNet18 Encoder with Cycle Loss. Various strategies, such as linear reduction and the Cycle Loss approach, were employed to strategically diminish the specified loss component. To assess our method, we compare it against several state-of-the-art methods using the PCN dataset and the CD-L1 metric. We evaluated our architecture using 12 random viewing points for each object from the test set. The results are presented in Table 1. As can be seen from Table 1, ImgAdaPoinTr and its enhanced versions outperform all other methods. For certain object classes, the improvement is as significant as 8.15% and average 2.9%.

The segmentator significantly impacts point cloud completion capabilities. Tables 4 and 5 demonstrate that the object classes, on which GDANet (Xu et al., 2021) was trained, benefit from the inclusion of the segmentator in the architecture. Specifically, SegEncAdaPoinTr attained the optimal performance on CD-L1 for car (where improvements are challenging) and showed a 32.43% enhancement on the CD-L2 metric for airplane.

The Figure 3 reveals that the airplane's wing edges are distinctly articulated, reflecting a kind attention to detail. For cars, there's enhanced detailing on the mirrors and wheels. The chair is presented as a unified object, devoid of any missing components.

4.3 Ablation Studies

We study the impact of another pipeline configurations to make sure that our comparison is as fair as possible. We consider 4 types of pipeline modifications on the base of using images and segmentation.

4.3.1 Enhancement Through Variable Loss

Throughout the training process, we progressively diminish the impact of the error associated with the coarse points cloud. We experimented with both linear and cyclic strategies. The cycle reduce approach yielded slightly better results. We marked it as AdaPoinTrVarLoss.

4.3.2 Enhancement Through Image Encoder Integration

- **ImgEncAdaPoinTr Resnet Encoder.** Enhance feature fusion by integrating a transformer layer

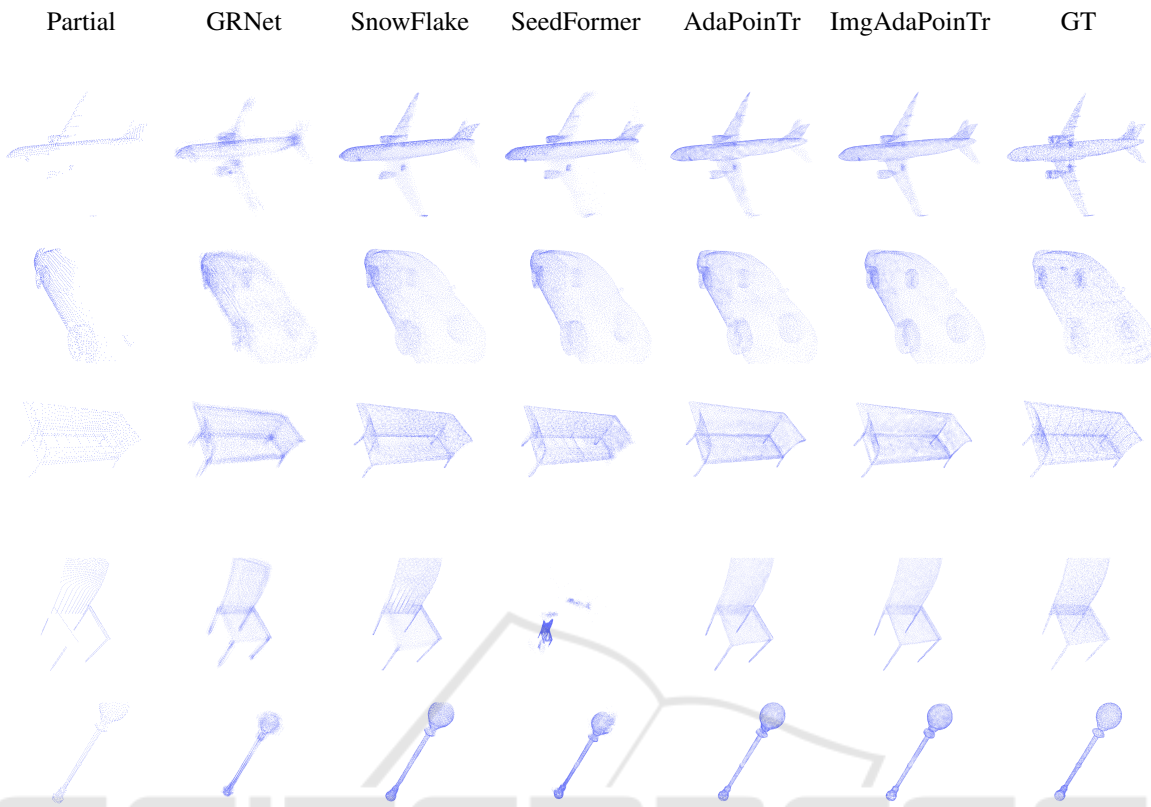

Figure 3: Visual comparison of point cloud completion on PCN dataset.

to fuse features from the primary pipeline, AdaPoinTr, and features derived from ResNet18. The transformer layer accepts the output from the transformer encoder, aiming to leverage the robustness of ResNet18 in conjunction with AdaPoinTr.

- **ImgCrossEncAdaPoinTr.** Fusing features from transformer Encoder using Cross-Encoder. Implement a Cross-Encoder layer (Reimers and Gurevych, 2019) to fuse features from the AdaPoinTr pipeline and ResNet18. This layer takes the output from the transformer encoder, aiming to harness the feature extraction capabilities of a Cross-Encoder.
- **ImgCrossDecAdaPoinTr.** Fusing features from transformer Decoder using Cross-Encoder. Similar to previous experiment, but use Cross-Encoder (Reimers and Gurevych, 2019) after transformer decoder.

4.3.3 Enhancement via Utilization of Pre-Trained Segmentator

The central thesis for the incorporation of the GDANet segmentator (Xu et al., 2021) is based on the hypothesis that a model pre-trained on a point cloud to comprehend the class of object parts, can infuse additional information conducive to the completion of generating absent object components.

- **SegEncAdaPoinTr.** Utilizing GDANet Encoder. We use GDANet segmentation model pre-trained on ShapeNet-part dataset. Utilizing a mask to filter by classes due to the relatively normal coverage on only 4 out of 8 classes in the PCN dataset, the layer accepts output from the transformer encoder.
- **SegEncDecAdaPoinTr.** Similar to SegEncAdaPoinTr but use fuse layer twice after transformer encoder and decoder layers.
- **SegCrossEncAdaPoinTr.** Integrate a Cross-Encoder layer to fuse features from AdaPoinTr and GDANet segmentator. Employing a mask for class filtering due to specific dataset constraints, the layer accepts output from the transformer encoder.
- **SegCrossDecAdaPoinTr.** Implement a Cross-Encoder layer (Reimers and Gurevych, 2019) for feature fusion from AdaPoinTr and GDANet transformer decoder.

Table 2: Ablation Study: Assessing the impact of architectural elements added to the AdaPoinTr (Evaluated using CD-L1). (A) is ImgEncAdaPoinTr.

Name	VarLoss	ImgEnc	AVG
AdaPoinTr			6.528
AdaPoinTrVarLoss	✓		6.482
(A)		✓	6.409
ImgAdaPoinTr	✓	✓	**6.347**

Table 3: Ablation study on different image encoders designs on the PCN Dataset (Evaluated using CD-L1).

Image Encoder Design	CD-L1(AVG)
ConvNext	6.636
ResNet50	6.426
ResNet18	**6.347**

- **TripleSegFoldNetAdaPoinTr.** Utilize L2-distance to identify nearest and farthest points, employing distances for sampling points and corresponding features as positive and negative examples, input into TripleLoss.

4.3.4 Enhancement Through Fusing Image Encoder Integration and Pre-Trained Segmentator

- **ImgEncSegDecAPTr.** Enhance feature fusion by integrating a transformer layer to fuse features from the primary pipeline, AdaPoinTr, and features derived from ResNet18. The transformer layer accepts the output from the transformer encoder, aiming to leverage the robustness of ResNet18 in conjunction with AdaPoinTr. Similar to SegEncDecAdaPoinTr, we utilize a fusion layer, but this occurs only after the transformer decoder layers.

4.4 Experiment Analysis

The comparison showed that the most efficacious results were obtained utilizing the ResNet Encoder coupled with Cycle Loss. This approach facilitated the attainment of a state-of-the-art result on the PCN dataset, surpassing AdaPoinTr.

In Table 1, a comparative analysis of ImgAdaPoinTr with other architectures is presented, focusing on the point completion task using the PCN dataset. The evaluation metric employed for this comparison is the Chamfer Distance L1 (CD-L1) and F-Score@1% averaged over all classes.

A detailed comparative analysis with AdaPoinTr (Yu et al., 2023), which served as the foundational model for our research, is subsequently provided. The entirety of the work can be bifurcated into two primary segments. The first encompasses the utilization of an image encoder and a variable loss function, while the second pertains to the incorporation of supplementary information derived from a pretrained segmentator.

Three pivotal experiments were conducted within the realm of image encoder utilization, which are presented down below.

4.4.1 Development of the Core Strategy

AdaPoinTr was augmented with a variable loss function, adjusting the coefficient value associated with the loss obtained during the coarse generation phase. As shown in the Table 2 this modification yielded a performance enhancement of 0.7%. Subsequently, an image encoder, specifically ResNet18, was integrated into AdaPoinTr, and its output was fused with the output from the transformer encoder, resulting in a performance boost of 1.82%. In Table 2 is (A). The final iteration combined both the variable loss function and the image encoder, leading to an aggregate improvement of 2.77%, which surpasses the cumulative impact of the individual components. As illustrated in Tables 6 in terms of CD-L1 metric, ImgAdaPoinTr improves the result by 7.54% the most in the Lamp category. In Table 7 based on the CD-L2 metric, the best performance is observed for the Lamp and Airplane categories, with scores of 27.55% and 25.71%, respectively. Conversely, there is only a minimal improvement observed in the categories of cars and boats measured using the CD-L1 metric. We have a hypothesis that achieving significant improvement for car is challenging, given the numerous objects within them where the scale of incomplete and complete point clouds does not align. As illustrated in Figure 4, we observed a reduction in both the median and maximum values for the CD-L1 and CD-L2 metric.

As illustrated in Figure 4, we observed a reduction in both the median and maximum values for the CD-L1 metric.

4.4.2 Image Encoders Designs

The second experiment revolved around the selection of the optimal image encoder. Three encoder architectures, namely ResNet18, ResNet50, and ConvNext, were evaluated. The findings, as tabulated in Table 3, revealed that ResNet18 outperformed the others, with ConvNext deteriorating the CD-L1 metric by 1.65% and ResNet50 enhancing it by 1.56%.

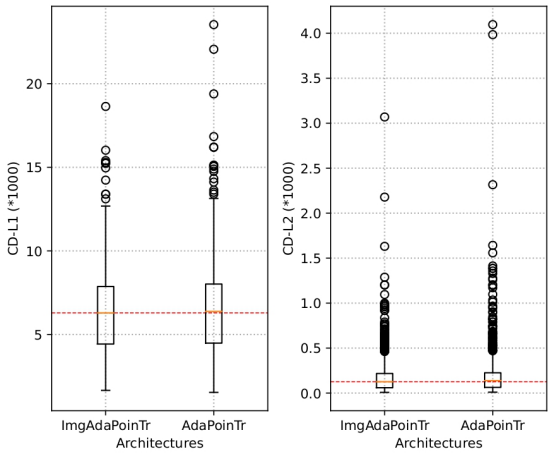

Figure 4: The difference in results between ImgAdaPoinTr and AdaPoinTr on the test dataset using the CD-L1 and CD-L2 metrics multiplied by 1000.

Table 4: Percentage Overlap of objects Between ShapeNet-Part and PCN Datasets.

Lamp	Airplane	Chair	Table	Car
66.69	66.50	55.10	61.25	12.05

4.4.3 Design of the Image Encoder's Integration

The third experiment was dedicated to assessing the architectural design of the image encoder's integration. Various strategies were explored, such as introducing the image encoder's output in the rebuild head as a global feature, post the transformer encoder layer, post the transformer decoder layer, and a simultaneous integration in both encoder and decoder, similar to the XMFNet architecture (Aiello et al., 2022). The most promising results were obtained when the image encoder's output was introduced post the transformer encoder. An alternative method for fusion is cross-encoder. Tables 6, 7 and 8 (presented in Appendix) provide an exhaustive comparative analysis of these methodologies based on metrics like Chamfer Distance L1 (CD-L1), Chamfer Distance L2 (CD-L2) and Earth Mover's Distance (EMD). All methods showed an improvement in metrics compared to AdaPoinTr. When comparing ImgEncAdaPoinTr and ImgCrossEncAdaPoinTr, it's clear that ImgEncAdaPoinTr generally outperforms based on the CD-L1 metric. It slightly underperforms only in the Car and Airplane categories, but excels in the others.

4.4.4 Segmentator

Next, we intend to describe the experimental results with the segmentator. In our investigations, we explored the potential of leveraging segmentation techniques to enhance point completion tasks. The underlying hypothesis was predicated on the notion that the vector representation of an individual point, employed for object part classification, would inherently encapsulate pertinent information conducive to the point completion task.

Table 5: Distribution of object segments within each category. Values in percentages.

Seg. ID	Airpl.	Car	Chair	Lamp	Table
1	45.60	5.49	34.17	16.37	73.12
2	32.34	6.90	41.12	60.20	23.11
3	12.80	16.43	21.57	1.23	3.77
4	9.26	71.18	3.14	22.20	-

As shown in Table 6, the conducted experiments did not yield improvements in the CD-L1 metric on average across all objects. However, when considering specific classes, for instance, SegEncAdaPoinTr improved by 3.86%, 0.96% (the best result among all experiments), and 5.27% for airplane, car, and lamp, respectively. We have a proposed explanation for this behavior, which we will detail below. We would like to highlight that at least one variant from the segmentator architectures enhances results for the following classes: airplane, car, chair (marginally), and lamp. In Table 7 when evaluated against the CD-L2 metric, the SegCrossDecAdaPoinTr model exhibited a notable improvement of 2.58% across all objects and 11.43%, 6.93%, and 16.33% for the airplane, chair, and lamp categories, respectively. It is noteworthy that all segmentation-based experiments rendered improvements for the airplane and lamp categories, with the exception of TripleSegFoldNetAdaPoinTr for the chair category. Furthermore, it is imperative to highlight that the employed segmentator was trained on a distinct dataset, albeit derived from ShapeNet. Table 4 displays the percentage overlap between ShapeNet-Part (Yi et al., 2016) and PCN datasets. A discernible pattern emerges from this data: the positive outcomes attributed to the segmentator align predominantly with object categories that exhibit dataset overlap between ShapeNet-Part and PCN. As illustrated in Table 8, SegEncAdaPoinTr outperformed other variants across the Earth Mover's Distance (EMD) metric, establishing itself as the most efficacious model in our segmentation-based experiments. SegCrossDecAdaPoinTr, while slightly inferior, still demonstrated improvements across most metrics compared to AdaPoinTr.

Among our experimental endeavors, we did not discern any substantial impact on outcomes when masking objects from classes present in PCN but absent in ShapeNet-part (Yi et al., 2016). However, we have a hypothesis that augmenting the number of object part classes and ensuring comprehensive overlap

in segmentation might yield more pronounced results in future investigations. We assume that having a diverse array of segments for each object category, without pronounced imbalances, is crucial. As observed in Table 6, segmentation didn't enhance the outcomes for tables. This can be linked to the distribution where over 96% of the table points are associated with two classes, and fewer than 4% with a third class. This distribution is detailed in Table 5.

5 CONCLUSIONS

This study presented the line of models for point cloud completion by accurately incorporating a view-guided approach and segmentation. This method utilizes images and incomplete point clouds to address the task. The ImgAdaPoinTr performs better for all classes of 3D objects considered in comparison with baselines. The best results were received by ImgEncSegDecAPTr, which is enchanced by fusion of image features and segmentation simultaneously. We also introduce the ImgPCN dataset, generated via our open-source rendering tool, which provides a new resource for evaluating point cloud completion techniques. Due to the revealed limitations of existing pre-trained segmentation models, we plan to widen ImgPCN with segmentation markdown and set up a precise experiment for fusing.

REFERENCES

Abbasi, R., Bashir, A. K., Alyamani, H. J., Amin, F., Doh, J., and Chen, J. (2022). Lidar point cloud compression, processing and learning for autonomous driving. *IEEE Transactions on Intelligent Transportation Systems*, 24(1):962–979.

Aiello, E., Valsesia, D., and Magli, E. (2022). Cross-modal learning for image-guided point cloud shape completion. *Advances in Neural Information Processing Systems*, 35:37349–37362.

Berger, M., Tagliasacchi, A., Seversky, L., Alliez, P., Guennebaud, G., Levine, J., Sharf, A., and Silva, C. (2016). A Survey of Surface Reconstruction from Point Clouds. *Computer Graphics Forum*, page 27.

Bi, S., Yuan, C., Liu, C., Cheng, J., Wang, W., and Cai, Y. (2021). A survey of low-cost 3d laser scanning technology. *Applied Sciences*, 11(9).

Dai, A., Ruizhongtai Qi, C., and Nießner, M. (2017). Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5868–5877.

Han, X., Li, Z., Huang, H., Kalogerakis, E., and Yu, Y. (2017). High-resolution shape completion using deep neural networks for global structure and local geometry inference. In *Proceedings of the IEEE international conference on computer vision*, pages 85–93.

Johnson, J., Ravi, N., Reizenstein, J., Novotny, D., Tulsiani, S., Lassner, C., and Branson, S. (2020). Accelerating 3d deep learning with pytorch3d. In *SIGGRAPH Asia 2020 Courses*, SA '20, New York, NY, USA. Association for Computing Machinery.

Li, J., Zhang, J., Wang, Z., Shen, S., Wen, C., Ma, Y., Xu, L., Yu, J., and Wang, C. (2022). Lidarcap: Long-range markerless 3d human motion capture with lidar point clouds. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20470–20480, Los Alamitos, CA, USA. IEEE Computer Society.

Li, R., Li, X., Fu, C.-W., Cohen-Or, D., and Heng, P.-A. (2019). Pu-gan: a point cloud upsampling adversarial network. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7203–7212.

Liu, Y., Fan, B., Xiang, S., and Pan, C. (2019a). Relation-shape convolutional neural network for point cloud analysis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8895–8904.

Liu, Z., Tang, H., Lin, Y., and Han, S. (2019b). Point-voxel cnn for efficient 3d deep learning. *Advances in Neural Information Processing Systems*, 32.

Nichol, A., Jun, H., Dhariwal, P., Mishkin, P., and Chen, M. (2022). Point-e: A system for generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*.

Poole, B., Jain, A., Barron, J. T., and Mildenhall, B. (2022). Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*.

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017a). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660.

Qi, C. R., Yi, L., Su, H., and Guibas, L. J. (2017b). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Raj, T., Hashim, F. H., Huddin, A. B., Ibrahim, M. F., and Hussain, A. (2020). A survey on lidar scanning mechanisms. *Electronics*, 9(5).

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. (2021). Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR.

Reimers, N. and Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In

Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Sarmad, M., Lee, H. J., and Kim, Y. M. (2019). Rl-gan-net: A reinforcement learning agent controlled gan network for real-time point cloud shape completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5898–5907.

Shen, Z., Liang, H., Lin, L., Wang, Z., Huang, W., and Yu, J. (2021). Fast ground segmentation for 3d lidar point cloud based on jump-convolution-process. *Remote Sensing*, 13(16):3239.

Smith, L. N. (2017). Cyclical learning rates for training neural networks. In *2017 IEEE winter conference on applications of computer vision (WACV)*, pages 464–472. IEEE.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M., and Solomon, J. M. (2019). Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (tog)*, 38(5):1–12.

Xiang, P., Wen, X., Liu, Y.-S., Cao, Y.-P., Wan, P., Zheng, W., and Han, Z. (2021). Snowflakenet: Point cloud completion by snowflake point deconvolution with skip-transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5499–5509.

Xie, H., Yao, H., Zhou, S., Mao, J., Zhang, S., and Sun, W. (2020). Grnet: Gridding residual network for dense point cloud completion. In *European Conference on Computer Vision*, pages 365–381. Springer.

Xu, M., Zhang, J., Zhou, Z., Xu, M., Qi, X., and Qiao, Y. (2021). Learning geometry-disentangled representation for complementary understanding of 3d object point cloud. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 3056–3064.

Yang, Y., Feng, C., Shen, Y., and Tian, D. (2018). Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 206–215.

Yi, L., Kim, V. G., Ceylan, D., Shen, I.-C., Yan, M., Su, H., Lu, C., Huang, Q., Sheffer, A., and Guibas, L. (2016). A scalable active framework for region annotation in 3d shape collections. *SIGGRAPH Asia*.

Yu, L., Li, X., Fu, C.-W., Cohen-Or, D., and Heng, P.-A. (2018). Pu-net: Point cloud upsampling network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2790–2799.

Yu, X., Rao, Y., Wang, Z., Liu, Z., Lu, J., and Zhou, J. (2021). Pointr: Diverse point cloud completion with geometry-aware transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12498–12507.

Yu, X., Rao, Y., Wang, Z., Lu, J., and Zhou, J. (2023). Adapointr: Diverse point cloud completion with adaptive geometry-aware transformers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45:14114–14130.

Yuan, W., Khot, T., Held, D., Mertz, C., and Hebert, M. (2018). Pcn: Point completion network. In *2018 international conference on 3D vision (3DV)*, pages 728–737. IEEE.

Zeng, J., Wang, D., and Chen, P. (2022). A survey on transformers for point cloud processing: An updated overview. *IEEE Access*, 10:86510–86527.

Zhang, X., Feng, Y., Li, S., Zou, C., Wan, H., Zhao, X., Guo, Y., and Gao, Y. (2021). View-guided point cloud completion. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15885–15894, Nashville, TN, USA. IEEE.

Zhou, H., Cao, Y., Chu, W., Zhu, J., Lu, T., Tai, Y., and Wang, C. (2022). Seedformer: Patch seeds based point cloud completion with upsample transformer. In *European conference on computer vision*, pages 416–432. Springer.

APPENDIX

We present the tables with our experimental results of ablation study on the next page. For last breaking results see our https://github.com/ImgAdaPoinTr

Table 6: Result of experiments on the ImgPCN dataset. We use the CD-L1 (multiplied by 1000) to compare AdaPoinTr with other our experiments. We conducted separate evaluations: one with architectures that incorporate an image encoder and another with a segmentator. For methods incorporating an image encoder, we present two rows. In the top row, the metric is assessed using 6 random viewing points for each object from the test set. For the bottom row, the number of random viewing points is 12. * Denotes the top result within its experimental group, surpassing the value in AdaPoinTr.

Name	AVG	Airplane	Cabinet	Car	Chair	Lamp	Sofa	Table	Watercraft
AdaPoinTr	6.528	3.681	8.823	7.476	6.850	5.478	8.353	5.801	5.763
AdaPoinTrVarLoss	6.482	3.614	8.746	7.422	6.845	5.277	8.365	5.831	5.755
ImgEncAdaPoinTr	6.414	3.596	8.701	7.517	**6.614**	5.217	8.165	5.785	5.714
	6.409	3.586	8.713	7.489	6.621	5.218	8.155	5.783	**5.709**
ImgAdaPoinTr (our)	6.356	3.520	**8.533**	7.437	6.690	5.086	8.105	5.715	5.762
	6.347	**3.515**	8.540	*7.425	6.677	**5.065**	**8.082**	**5.714**	5.756
ImgCrossEncAdaPoinTr	6.487	3.573	8.868	7.429	6.692	5.439	8.217	5.890	5.787
	6.482	3.571	8.864	7.448	6.683	5.426	8.199	5.883	5.787
SegCrossEncAdaPoinTr	6.660	3.610	8.970	7.529	6.872	5.567	8.567	6.256	5.915
SegCrossDecAdaPoinTr	6.549	3.649	8.882	7.519	*6.808	5.311	8.522	5.924	5.780
SegEncAdaPoinTr	6.569	*3.544	8.893	**7.404**	6.939	*5.189	8.544	6.179	5.862
SegEncDecAdaPoinTr	6.595	3.649	9.003	7.526	6.830	5.383	8.525	6.058	5.784
TripleSegFoldNetAdaPoinTr	6.658	3.722	8.980	7.561	7.009	5.329	8.725	6.079	5.856

Table 7: Result of experiments on the ImgPCN dataset. We use the CD-L2 (multiplied by 1000) to compare AdaPoinTr with other our experiments. We conducted separate evaluations: one with architectures that incorporate an image encoder and another with a segmentator. * Denotes the top result within its experimental group, surpassing the value in AdaPoinTr.

Name	AVG	Airplane	Cabinet	Car	Chair	Lamp	Sofa	Table	Watercraft
AdaPoinTr	0.194	0.070	0.319	0.185	0.202	0.196	0.305	0.140	0.133
AdaPoinTrVarLoss	0.185	0.055	0.297	**0.181**	0.199	0.164	0.310	0.143	**0.131**
ImgEncAdaPoinTr	0.183	0.055	0.303	0.204	**0.175**	0.160	0.287	0.150	0.132
	0.183	0.054	0.304	0.200	0.177	0.159	0.286	0.149	0.132
ImgAdaPoinTr (our)	0.176	0.053	**0.287**	0.194	0.185	0.145	0.268	0.140	0.138
	0.174	*0.052	**0.287**	0.192	0.183	**0.142**	**0.262**	0.138	0.137
ImgCrossEncAdaPoinTr	0.192	0.053	0.324	0.192	0.182	0.183	0.303	0.161	0.137
	0.191	0.053	0.323	0.195	0.182	0.180	0.300	0.159	0.137
SegCrossEncAdaPoinTr	0.204	0.050	*0.315	0.187	0.190	0.193	0.323	0.235	0.138
SegCrossDecAdaPoinTr	*0.189	0.062	0.320	0.190	0.188	0.164	0.313	0.141	0.134
SegEncAdaPoinTr	0.205	**0.048**	0.318	*0.182	0.201	*0.161	0.358	0.227	0.144
SegEncDecAdaPoinTr	0.197	0.056	0.326	0.190	*0.186	0.164	0.341	0.180	0.135
TripleSegFoldNetAdaPoinTr	0.209	0.066	0.332	0.191	0.226	0.166	0.379	0.172	0.141

Table 8: Result of experiments on the ImgPCN dataset. We use the EMD (multiplied by 1000) to compare AdaPoinTr with other our experiments. We conducted separate evaluations: one with architectures that incorporate an image encoder and another with a segmentator. * Denotes the top result within its experimental group, surpassing the value in AdaPoinTr.

Name	AVG	Airplane	Cabinet	Car	Chair	Lamp	Sofa	Table	Watercraft
AdaPoinTr	24.46	14.12	34.14	30.69	22.36	20.86	32.80	18.09	22.60
AdaPoinTrVarLoss	24.17	13.86	33.48	30.37	22.20	20.47	32.89	17.78	22.31
ImgEncAdaPoinTr	23.38	14.11	31.22	29.41	21.01	20.85	31.47	16.93	22.07
	23.40	14.09	31.20	29.38	21.09	20.94	31.53	16.91	22.06
ImgAdaPoinTr (our)	22.79	13.39	30.54	29.20	20.94	20.09	30.09	16.63	21.46
	22.78	**13.37**	**30.53**	**29.17**	**20.93**	20.10	**30.14**	**16.58**	21.46
ImgCrossEncAdaPoinTr	23.19	13.75	31.77	29.76	21.10	19.75	30.76	17.30	21.35
	23.20	13.76	31.78	29.79	21.14	*19.75	30.79	17.29	**21.32**
SegCrossEncAdaPoinTr	25.05	14.17	34.34	30.61	22.47	23.48	33.58	19.01	22.75
SegCrossDecAdaPoinTr	23.98	13.90	33.20	*29.68	22.36	20.53	32.27	*17.96	21.95
SegEncAdaPoinTr	*23.74	*13.74	*33.05	29.80	*22.02	**19.05**	*32.02	18.28	22.00
SegEncDecAdaPoinTr	24.57	14.34	33.40	30.93	22.78	21.70	32.50	18.46	22.48
TripleSegFoldNetAdaPoinTr	24.50	15.20	32.95	29.91	23.26	20.28	32.13	20.45	*21.81