Cross-modal Learning for Image-Guided Point Cloud Shape Completion

Anonymous Author(s)
Affiliation
Address
email

Abstract

In this paper we explore the recent topic of point cloud completion, guided by an auxiliary image. We show how it is possible to effectively combine the information from the two modalities in a localized latent space, thus avoiding the need for complex point cloud reconstruction methods from single views used by the state-of-the-art. We also investigate a novel self-supervised setting where the auxiliary image provides a supervisory signal to the training process by using a differentiable renderer on the completed point cloud to measure fidelity in the image space. Experiments show significant improvements over state-of-the-art supervised methods for both unimodal and multimodal completion. We also show the effectiveness of the self-supervised approach which outperforms a number of supervised methods and is competitive with the latest supervised models only exploiting point cloud information.

1 Introduction

The rise in popularity of 3D sensing technologies such as depth cameras, laser scanners, LiDARs, etc. is making the processing of point cloud data ever more important. The acquisitions produced by those instruments are often incomplete due to occlusions by objects in the environment, reflections, and viewing angles. This limits the exploitability of those data in tasks like scene understanding [1], robotic vision [2], autonomous driving [3] and many more. Completing a point cloud from partial observations is a challenging ill-posed inverse problem that requires strong prior knowledge about shapes to be effectively regularized.

At the same time, we know that humans are very proficient at mapping the visual concepts learnt from 2D images to understand the 3D world, and are able to successfully infer the shape of partial 3D objects from their 2D experiences. It is thus sensible to expect that point cloud completion techniques can benefit from 2D images to better characterize the 3D shape to be completed. Indeed, several applications of interest in robotic vision can take advantage of multimodal data where the 3D acquisitions of a depth-sensing instrument are paired with images from an RGB camera. It is also worth noting that the two modalities may be acquired from different vantage points, either thanks to disparities in the acquisition geometry or because the vantage point has changed with the passing of time. This makes it clear that the two modalities may carry complementary information and effectively fusing it is key to unlock better completion performance. Nevertheless, the literature on the topic of point cloud completion [4, 5, 6, 7, 8, 9, 10, 11] has largely focused on the single-modality problem, where only priors about 3D shapes are exploited. Only recently, image-guided completion has started to receive attention [12].

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

In this paper, we study how the side information offered by a single image can be used in addition to shape priors to complete a partial point cloud. While following the setting of ViPC [12], we extend the multimodal completion methodology in several different ways. First, ViPC [12] is bottlenecked by the need to estimate a coarse point cloud from the image via single-view reconstruction techniques to fuse the information. We avoid this task by proposing a novel architecture that performs fusion in a latent domain via cross-attention operations on fine-grained, localized representations of the two modalities, coupled with a flexible decoder that allows to complete areas of varied size. Moreover, the multimodal setting is uniquely poised for self-supervised learning. In fact, the input image, especially when captured from a different vantage point, may offer a supervisory signal to guide the completion of those areas occluded in the partial point cloud but visible in the image. This is especially interesting for practical applications where it could be difficult to have access to complete shapes, but significantly easier to have images from a different viewing angle. Therefore, we propose to augment the known 3D self-reconstruction losses with the exploitation of a differentiable renderer to measure the fidelity of the completed point cloud in the image space.

Our experiments show that the proposed model significantly outperforms the state-of-the-art on both the supervised and self-supervised settings. In particular, the addition of the rendering loss allows the self-supervised image-guided model to outperform several supervised baselines and to be competitive with the latest supervised models only exploiting point cloud information.

2 Related work

Point cloud completion 3D shape completion is a long-standing problem in computer vision. Early works devised explicit geometric descriptors or relied on shape retrieval from large datasets [13] [14] [15] [16]. Since the advent of neural networks operating on raw point cloud data, several models for the completion problem have been studied [17]. They are mostly based on the encoder-decoder architecture, pioneered by PCN [4], which was the first model that did not require any assumption of structure or annotation information about the underlying shape. TopNet [5] presents a hierarchical rooted tree structure that generates structured point clouds as a collection of its subsets. AtlasNet [6] and MSN [18], on the other hand, recreate the point cloud by assessing a set of parametric surface elements. Convolutional-based approaches ([19, 20]) use a voxelixed representation of shapes as input to 3D CNNs; nevertheless, this representation introduces undesirable approximations in the shape due to coordinate quantization effects. GRNet [21] uses techniques to represent point cloud onto a 3D grid, so that CNNs can be exploited, without losing structural information. Recently, VRCNet [8] has proposed a dual path architecture and a VAE-based relation enhancement module for probabilistic modeling. Architectures based on transformers have also been proposed. PointTr [9] changes the transformer block to take advantage of the inductive bias of 3D geometries, creating a geometry-aware block that models local geometry relations. Moreover, SnowflakeNet [22] generates child points by gradually splitting parent points by means of a Skip-Transformer that learns the appropriate splitting modes for particular regions. As a result, the network is able to predict highly detailed shape geometries. Finally, it is worth mentioning that the point cloud completion literature is split between two settings: one, as in the aforementioned works, where the partial input has the same number of points as the completed point cloud and another where the completed point cloud has more points than the input such as in [23, 24]. In this paper, we will consider the former setting.

View-guided completion Recently, the usage of auxiliary data to complement point cloud completion has been introduced by ViPC [12]. The idea is to help the reconstruction objective using side information available as a different imaging modality. In particular, ViPC assumes that an image corresponding to a view of the same object is also available for the completion task, and it exploits the image to retrieve the global shape information that is lacking in the incomplete point cloud. The image is processed by a pre-trained single-view reconstruction model, which estimates a coarse point cloud from the image, representing the entire shape. The key challenge in this setting is how to effectively combine features extracted from the two modalities. Unlike ViPC, our approach leverages direct fusion at a feature level, avoiding the need to explicitly reconstruct a coarse point cloud from a single image, a generally hard inverse problem in itself and full of pitfalls.

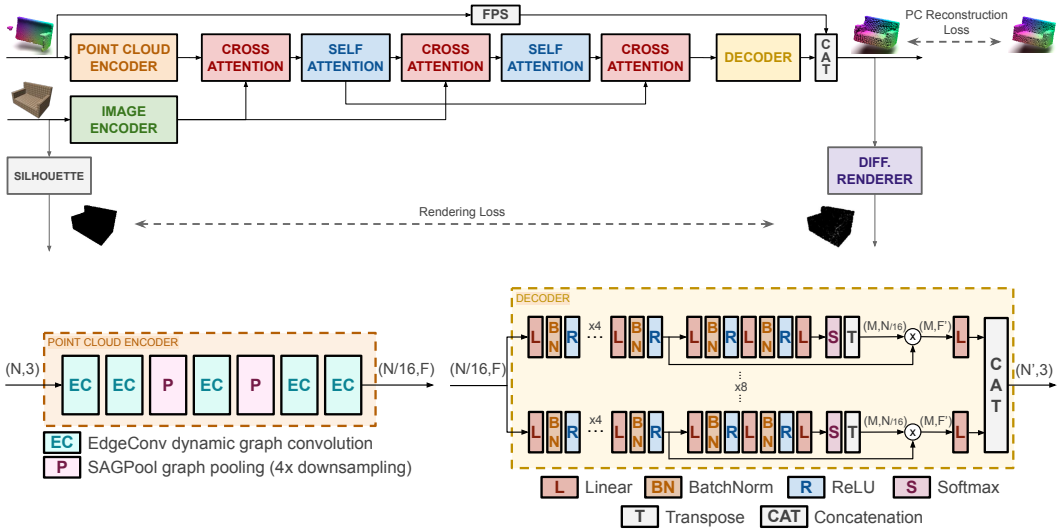

Figure 1: Architecture overview. Localized features from the partial point cloud and the input image are jointly processed via cross- and self-attentions. A decoder reconstructs the target number of points from the feature space with attention-based upsampling. The input partial point cloud is downsampled with farthest point sampling (FPS) and concatenated. Supervised training only uses the point cloud reconstruction loss with respect to the complete point cloud. Self-supervised training has a point cloud reconstruction loss with respect to a less partial point cloud and a rendering loss.

Self-supervised strategies for completion All the previously mentioned approaches rely on complete ground truth as a supervisory training signal. This may be difficult to obtain in real-word scenarios. Self-supervised training strategies avoid the need for retrieving such expensive ground truths. However, the amount of work on this topic is rather limited. Wang et al. [10] use resampling that removes further points from an already partial point cloud and mixup among shapes to train their completion network in a self-supervised manner. Similarly, Mittal et al. [11] also propose an inpainting procedure that leverages further partializations of partial point clouds. To the best of our knowledge, there is no work exploring the availability of a different modality, namely an image to provide a self-supervisory signal to the point cloud completion task.

3 Proposed Methods

We address the setting in which a partial point cloud needs to be completed with the assistance of an image of the object taken from a certain viewpoint. Our goal is to study how to leverage this side information in the most effective manner. To this purpose we study i) a supervised learning setting, for which we show that the image features can be effectively fused with those of the partial point cloud in a latent space; ii) a self-supervised setting based on the idea that the image may contain clues about the missing part and can thus serve as a (self-)supervisory signal. Fig. 1 shows an overview of the proposed method, named XMFnet (Cross-Modal Fusion network) which will be detailed in the next sections.

3.1 Architecture and supervised setting

At a high level, the architecture of XMFnet is composed by two modality-specific feature extractors that capture localized features of the input point cloud and image, summarized at a small number of points/pixels, followed by a sequence of cross-attention and self-attention operations that progressively merge the two feature spaces. Finally, a decoder upsamples this localized information to estimate a predefined number of points of the missing component. More formally, we denote the partial point cloud as $\mathbf{X} \in \mathbb{R}^{N \times 3}$, the input view as an image $\mathbf{I} \in \mathbb{R}^{P_x \times P_y \times 3}$ and the complete point cloud as $\mathbf{Y} \in \mathbb{R}^{N \times 3}$. The task of our model is to predict a complete shape $\hat{\mathbf{Y}} \in \mathbb{R}^{N \times 3}$ given $\mathbf{X}$ and $\mathbf{I}$ as inputs.

Notice we follow the conventional setting where the partial and complete point clouds have the same number of points, meaning that there is a resampling of the known part.

3.1.1 Point cloud and image encoder

The point cloud encoder should extract localized features from the partial shape $\mathbf{X}$. It is important to keep a degree of locality, i.e., associating features to a small number of points $N_X < N$ rather than a single global embedding because the information about the missing part that needs to be estimated by the entire model is also mostly localized. However, it is also important to have a sufficiently large receptive field to infer some global information about the object. For this reason, we adopt a graph-convolutional architecture with graph pooling operations. The architecture is a sequence of graph-convolutional layers (EdgeConv [25]) interleaved by pooling operations (Self-Attention Graph Pooling [26]) to reduce the cardinality of the point cloud. Pooling has the double purpose of expanding the receptive field to also include more global information and reduce the complexity of the subsequent cross-attention operations fusing the two modalities.

Any network that extracts features from an image can be utilized as encoder for view $\mathbf{I}$. The design principles follow those of the point-cloud encoder, i.e., features localized at a subset $N_I < P_x P_y$ of the image pixels obtained from a sufficiently large receptive field are produced as output.

We will refer to the features produced by the point cloud encoder as $\mathbf{H}_X \in \mathbb{R}^{N_X \times F_X}$ and by the image encoder as $\mathbf{H}_I \in \mathbb{R}^{N_I \times F_I}$.

3.1.2 Modality fusion

Once we have collected localized information from the two modalities, we need to combine them effectively to capture their complementary information, despite the obvious domain gap. The attention mechanism is particularly suited to find correspondences between the features of a region of the point cloud and a region of the image. The cross-attention layer in our architecture uses the Transformer's multihead attention mechanism [27]. The point cloud features are projected to form the query tensor, while the image features are projected to form the key and value tensors, and then attention mechanism aggregates the features from different image regions according to the weights determined by the cross-correlation between the two modalities. More formally:

$$\mathbf{Q}_X = \mathbf{H}_X \mathbf{W}_Q, \quad \mathbf{K}_I = \mathbf{H}_I \mathbf{W}_K, \quad \mathbf{V}_I = \mathbf{H}_I \mathbf{W}_V \tag{1}$$

$$\mathbf{H}_{\text{fused}} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{F}}\right)\mathbf{V} \tag{2}$$

being $\mathbf{W}_Q \in \mathbb{R}^{F_X \times F}$, $\mathbf{W}_K, \mathbf{W}_V \in \mathbb{R}^{F_I \times F}$ the projection weights. The fused features $\mathbf{H}_{\text{fused}} \in \mathbb{R}^{N_X \times F}$ produced by the cross-attention mechanism can be regarded as the original point cloud features enriched by the image features.

The XMFnet architecture depicted in Fig. 1 shows a self-attention layer after the cross-attention fusion. The goal of this operation is to have a permutation-invariant transformation of the features with a global receptive field so that any information from the image not properly integrated can be rectified. Self-attention works exactly like Eq. (2) except for the fact that $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ are all different projections of the same features. Furthermore, a sequence of multiple cross- and self-attention layers can be used to more effectively integrate the information from the two modalities via a "slow" fusion. We remark that at the end of this sequence we use a special cross-attention layer that merges information from the end and the beginning of the sequence allowing better flexibility in the decision of the desired abstraction level (higher-level features cross-attend lower-level features).

3.1.3 Decoder and Supervised Loss

The decoder is a crucial component of our architecture as it should take the joint feature embedding and learn to reconstruct a complete point cloud preserving both global and local structure. To be precise, the decoder seeks to estimate the positions of a number of points that upper bounds the size

of the missing part, so that they can be concatenated to a version of the input partial point cloud subsampled by means of farthest point sampling (FPS). This mechanism is reminiscent of what is done in the setting with a variable number of points where only the missing part is estimated [23, 24]. For example, in our experiments, we upper bounded the size of the missing part to 50% of the total number of points, thus having N' points estimated by the decoder concatenated to N' points from the subsampled input. However, our method allows to be flexible and handle more incomplete inputs by simply tuning the desired ratio of points to be estimated and points provided from the partial input. Typically, the latent space where we perform feature fusion is much more localized to constrain complexity and allow higher-level features so that $N_X \ll N'$, thus requiring the decoder to upsample the feature field. We perform this operation by using a number of attention-based operations, inspired by the work in [28], that convert features to points in parallel with the idea that each branch specializes on the reconstruction of a sub-region of the missing part. The structure is depicted in Fig.1. More formally, calling $\mathbf{H} \in \mathbb{R}^{N_X \times F}$ the features provided to the decoder, the output $\hat{\mathbf{Y}}_i \in \mathbb{R}^{N'/K \times 3}$ of each of the K branches is computed as:

$$\mathbf{Z}_i = \text{MLP}_i^{\text{proj}}(\mathbf{H}) \qquad i = 1, \dots, K \tag{3}$$

$$\hat{\mathbf{Y}}_i = \left(\text{softmax}(\text{MLP}_i^{\text{dec}}(\mathbf{Z}_i))^T \mathbf{Z}_i\right) \mathbf{W}_{\text{out},i} \qquad i = 1, \dots, K \tag{4}$$

where $\text{MLP}_i^{\text{proj}} : \mathbb{R}^F \to \mathbb{R}^{F'}$, $\text{MLP}_i^{\text{dec}} : \mathbb{R}^{F'} \to \mathbb{R}^{N'/K}$ are multilayer perceptrons with different weights for each branch, projecting features to K subspaces and generating attention weights for the resampling process, respectively. $\mathbf{W}_{\text{out},i} \in \mathbb{R}^{F' \times 3}$ is projection matrix to 3D space. Finally, the completed point cloud is generated by concatenation of the outputs of all decoder branches and the partial input subsampled by FPS as:

$$\hat{\mathbf{Y}} = \left[\hat{\mathbf{Y}}_1, \hat{\mathbf{Y}}_2, \dots, \hat{\mathbf{Y}}_K, \text{FPS}(\mathbf{X})\right]. \tag{5}$$

Supervised training is performed using the L1 Chamfer Distance (CD) between the generated shapes and the ground truth shapes, defined as follows:

$$\mathcal{L}_{\text{CD}}(\mathbf{Y}, \hat{\mathbf{Y}}) = \frac{1}{2N} \sum_{\mathbf{y} \in \mathbf{Y}} \min_{\hat{\mathbf{y}} \in \hat{\mathbf{Y}}} \|\mathbf{y} - \hat{\mathbf{y}}\| + \frac{1}{2N} \sum_{\hat{\mathbf{y}} \in \hat{\mathbf{Y}}} \min_{\mathbf{y} \in \mathbf{Y}} \|\hat{\mathbf{y}} - \mathbf{y}\|. \tag{6}$$

3.2 Self-supervised setting

The multimodal completion problem addressed in this paper is uniquely poised for self-supervised learning. In fact, the image available as input may contain complementary information with respect to the point cloud and, crucially, cues about the missing part. This is especially true if the image is collected from a different viewpoint or at a different time with respect to the point cloud, resulting in different kinds of occlusions.

Existing architectures for self-supervised completion [10] [11] rely solely on point cloud supervision, due to their unimodal nature. The key insight of the proposed method (Fig. 2) is to supplement completion losses on points with a loss measuring a reprojection error in the image space. In particular, we measure whether the reconstructed point cloud produced by the architecture described in Sec. 3.1 leads to an image similar to the input one, when captured from the correct viewpoint.

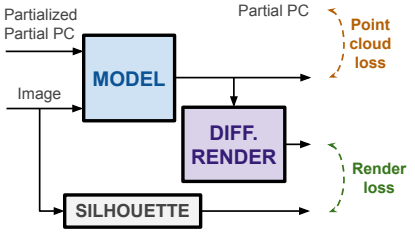

Figure 2: Self-supervised training.

In order to measure this information and use it in the training process, we include a differentiable rendering module based on alpha compositing [29] which generates a rasterized version of the object, employing provided camera parameters. In order to ensure consistency with the input image, intrinsic and extrinsic camera parameters may be estimated with a number of well-known methods [30] [31]

[32]. In order to minimize the domain shift between the input image and the result of the rendering process, we work on silhouettes, i.e., binary masks of objects. The differentiable renderer produces a soft silhouette with continuous values, while the input image is directly binarized. Inevitable inaccuracies in the camera parameters and rendering process will typically yield unreliable borders of the silhouette. Therefore, we also compute a border mask with a simple edge detector (Laplacian of Gaussian) and discount the loss function by a factor $\varepsilon < 1$ for the pixels in the mask. In summary, our rendering loss is defined as:

$$\mathcal{L}_{\text{render}} = \left\| \mathbf{M} \odot \left[R(\hat{\mathbf{Y}}) - S(\mathbf{I}) \right] \right\|_1, \qquad \mathbf{M}_{i,j} = \begin{cases} \varepsilon & \text{if } (i,j) \in \text{edge} \\ 1 & \text{otherwise} \end{cases} \tag{7}$$

being R the differentiable silhouette renderer and S the silhouette binarizer. We remark that [33] proposes to use rendering to improve performance in point cloud completion, aiding the learning process through an image domain supervision. However, their approach is supervised and is based on rendering the ground truth and generated point cloud in depth-maps with different view-points.

In addition to the rendering loss, our self-supervised framework also adopts self-supervision in the point cloud domain. In particular, we use a combination of the resampling and mixup approaches proposed in [10, 11]. Resampling consists in removing random portions of the original partial input point cloud, yielding even more partial shapes. As a result, the original partial input is employed as a pseudo-ground truth. Mixup combines a pair of partial shapes weighed according to Beta distribution in an attempt at increasing the complexity of the shapes processed by the network. Differently from [10], we also have images associated with a partial shape in our setting. Hence, we also mix the images in such a way that the mixup technique is carried out symmetrically for the two modalities.

We remark that the rendering loss is comparatively weaker than the point cloud loss and, by itself, has a number of ambiguities due to the lack of depth information and the use of silhouettes. For this reason, it is important to combine it with the point cloud loss. We found that the density-aware Chamfer Distance (DCD) [34], a version of CD that is more sensitive to non-uniform point distributions is superior in this self-supervised scenario to regularize the ambiguities of the rendering loss. Our overall self-supervised training procedure is therefore as follows. We alternate between a step that optimizes the point cloud loss consisting in a weighted CD:

$$\mathcal{L}_{\text{PC}} = \frac{1-\beta}{2N} \sum_{\mathbf{y} \in \mathbf{Y}} \min_{\hat{\mathbf{y}} \in \hat{\mathbf{Y}}} \|\mathbf{y} - \hat{\mathbf{y}}\| + \frac{\beta}{2N} \sum_{\hat{\mathbf{y}} \in \hat{\mathbf{Y}}} \min_{\mathbf{y} \in \mathbf{Y}} \|\hat{\mathbf{y}} - \mathbf{y}\|, \tag{8}$$

and a step optimizing a combination of DCD and rendering loss:

$$\mathcal{L}_{\text{I}} = \left[\frac{1}{2N} \sum_{\mathbf{y} \in \mathbf{Y}} \left(1 - \frac{1}{N} e^{-\alpha \|\mathbf{y} - \mathbf{w}\|_2} \right) + \frac{1}{2N} \sum_{\hat{\mathbf{y}} \in \hat{\mathbf{Y}}} \left(1 - \frac{1}{N} e^{-\alpha \|\hat{\mathbf{y}} - \mathbf{z}\|_2} \right) \right] + \lambda \mathcal{L}_{\text{render}} \tag{9}$$

where $\mathbf{w} = \arg\min_{\hat{\mathbf{y}} \in \hat{\mathbf{Y}}} \|\mathbf{y} - \hat{\mathbf{y}}\|_2$ and $\mathbf{z} = \arg\min_{\mathbf{y} \in \mathbf{Y}} \|\mathbf{y} - \hat{\mathbf{y}}\|_2$. Notice that $\mathcal{L}_{\text{PC}}$ and the DCD part of $\mathcal{L}_{\text{I}}$ use resampling and mixup for point clouds, while $\mathcal{L}_{\text{render}}$ does not ($\mathcal{L}_{\text{I}}$ is computed with full minibatch for the point-cloud part and half minibatch with the original partials for rendering).

4 Experimental results

4.1 Experimental Settings and Implementation Details

All the experiments are conducted on the ShapeNet-ViPC[12] dataset. The dataset contains 38,328 objects from 13 categories; for each object it comprises 24 partial point clouds with occlusions generated under 24 viewpoints, using the same settings as ShapeNetRendering [35]. The input and ground truth point clouds contains $N = 2048$ points each. Each 3D shape is rotated to the pose corresponding to a certain view point after being normalized within the bounding sphere with radius of 1. Images are generated from the 24 view points of ShapeNetRendering and have a resolution of 224×224 pixels. For all the experiments in this paper, we employ the same selection used in [12]: we used 31,650 objects from eight categories, with 80% of them for training and 20% for testing.

Table 1: Mean Chamfer Distance per point ($\times 10^{-3}$). ShapeNet-ViPC dataset, supervised.

Methods	Avg	Airplane	Cabinet	Car	Chair	Lamp	Sofa	Table	Watercraft
AtlasNet [6]	6.062	5.032	6.414	4.868	8.161	7.182	6.023	6.561	4.261
FoldingNet [39]	6.271	5.242	6.958	5.307	8.823	6.504	6.368	7.080	3.882
PCN [4]	5.619	4.246	6.409	4.840	7.441	6.331	5.668	6.508	3.510
TopNet [5]	4.976	3.710	5.629	4.530	6.391	5.547	5.281	5.381	3.350
ECG [7]	4.957	2.952	6.721	5.243	5.867	4.602	6.813	4.332	3.127
VRC-Net [8]	4.598	2.813	6.108	4.932	5.342	4.103	6.614	3.953	2.925
ViPC [12]	3.308	1.760	4.558	3.183	2.476	2.867	4.481	4.990	2.197
XMFnet	**1.443**	**0.572**	**1.980**	**1.754**	**1.403**	**1.810**	**1.702**	**1.386**	**0.945**

The partial point cloud is downsampled by farthest point sampling to $N' = 1024$ points and concatenated to the output of the decoder that produces $N' = 1024$ points leading to a completed point cloud with 2048 points. The decoder has $K = 8$ branches, each of them producing $M = 128$ points. The point cloud encoder employs EdgeConv and SAGPooling layers; the EdgeConv layers selects $k = 20$ nearest neighbors, while the two pooling layers use $k = 16$ and $k = 6$ nearest neighbors, respectively. The original point cloud is overall downsampled by a factor of 16, resulting in $N_X = 128$ points with $F_X = 256$ features. The image encoder is built with a ResNet18 [36] as backbone, it extracts $N_I = 14 \times 14 = 196$ pixels with $F_I = 256$ features. The multihead attention has 4 attention heads, with embedding size $F = 256$. In the $\mathcal{L}_{\text{I}}$ loss we use $\lambda = 0.15$. The mask factor for the edge detector has been set to $\varepsilon = 0.4$.

The differentiable renderer has been implemented with PyTorch3D[37]. The proposed framework is implemented in PyTorch and trained on an Nvidia V100 GPU. Class-specific training is performed for all models, using the Adam optimizer [38] for roughly 200 epochs with a batch size of 128. The learning rate is initialized to 0.001 and reduced by a factor of 10 at epoch 25 and 125.

4.2 Main Results

4.2.1 Supervised Learning

We first compare XMFnet against several baselines for the point cloud completion under supervised learning. Since the new multimodal setting with an auxiliary image has been introduced only recently, ViPC [12] represents the only method fully comparable to ours. However, we also report the results of a number of state-of-the-art architectures for completion with only point cloud input, when retrained on the ViPC dataset. AtlasNet[6] reconstructs a point cloud by estimating parametric surface elements. FoldingNet[39] is a 2-D grid based auto-encoder. PCN[4] is an encoder-decoder framework that reconstructs the point cloud in a coarse-to-fine manner. TopNet[5] has a rooted tree structure in the decoder. ECG[7] is an edge-aware completion method based on Graph Convolutions. VRC-Net[8] is the most recent method adopting a VAE-based model with a dual path architecture and probabilistic modeling. In line with the previous evaluation protocols, we use CD and F-score [40] as metrics for the reconstruction quality. Before evaluating the CD, we normalize the output of the models to fit into the unit sphere. Table 1 reports the experimental results and shows XMFnet outperforming the other techniques by a significant margin. While part of this gain with respect to state-of-the-art models for point cloud completion can be attributed to the use of the input image, it is worth noting that we also report significant improvements over the multimodal ViPC. This highlights the sub-optimality of performing modality fusion by resorting to estimating a coarse point cloud from the single input image, as in ViPC, rather than working in a latent feature space. Qualitative comparisons[1] are shown in Fig. 4. Our method is capable of producing cleaner completions than the other baselines, with fewer outliers and a more uniform point distribution. Additional visualizations and F-Score results are reported in the supplementary material.

[1] Visualizations for ViPC [12] have been kindly provided by the original authors.

Table 2: Mean Chamfer Distance per point ($\times 10^{-3}$). ShapeNet-ViPC dataset.

Methods	Airplane	Lamp	Watercraft
AtlasNet [6]	5.032	7.182	4.261
FoldingNet [39]	6.504	6.368	3.882
PCN [4]	4.246	6.331	3.510
TopNet [5]	3.710	5.547	3.350
ECG [7]	2.952	4.602	3.127
VRC-Net [8]	2.813	4.103	2.925
VRC-Net(self-sup.)	4.315	8.023	7.259
ViPC [12]	1.760	2.867	2.197
XMFnet (sup.)	0.572	1.810	0.945
XMFnet (self-sup.)	2.426	6.269	3.423

Table 3: Unimodal vs. Multimodal completion (supervised)

Method	Avg	Airplane	Cabinet	Lamp
Unimodal	1.570	0.626	2.114	1.980
Multimodal	**1.470**	**0.572**	**1.973**	**1.810**
↳ **best view**	1.223	0.545	1.426	1.697
↳ **worst view**	1.819	0.722	2.621	2.115

Table 4: Ablation study for the Self-Supervised method (*airplane*)

Resampling	Mixup	Rendering	CD(10^{-3})
✓	✗	✗	4.568
✓	✓	✗	4.239
✓	✓	✓	**2.426**

4.2.2 Self-Supervised

We notice that we are the first to propose a self-supervised training strategy for multimodal completion, so the setting in which supervisory information can be gathered from an input image is unexplored. For this reason, we compare to a number of unimodal and multimodal supervised methods as well as a self-supervised version using resampling and mixup of a unimodal state-of-the-art model[2], reported in Table 2. We notice that our self-supervised method outperforms a number of supervised baselines and it is close to the performance of the most recent unimodal supervised models. Examples of visualizations are reported in the supplementary material.

4.3 Ablation Studies

In order to verify the effectiveness of the proposed design, we study the impact of the auxiliary image input on the completion performance. In order to ensure a comparison as fair as possible, the version of XMFnet that uses only point cloud information has the image encoder removed and the cross-attention blocks replaced with self-attention ones. The unimodal architecture is then trained with the same settings as the multimodal one, and the results are reported in Table 3. The results show that the addition of the image input provides a significant improvement in performance. Notice that besides the result averaged over all the possible 24 views, we also report the performance with the worst view and the best view. Indeed, we are interested in investigating how the viewpoint of the image affects completion. Fig. 3 reports the CD for different views for a specific point cloud. It is clear that some views provide complementary information due to their different vantage point and allow to substantially improve over the average result. A small number of "bad" views leads to results comparable to the unimodal case.

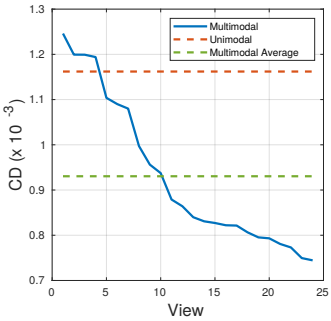

Figure 3: Impact of image contribution as function of point of view, sorted by reconstruction CD, for sample *airplane a2c2ad021c80f08c973e06016b069172*, supervised setting.

Furthermore, it is interesting to study the impact of our novel rendering loss in the self-supervised setting. We noticed that it allows the training process to have a faster and smoother convergence and that the overall completion performance is increased from both a

[2]We remark the difficulty in reproducing several published methods in the self-supervised setting. The code for [10, 11] was not available. The code to reproduce ViPC [12] is also incomplete so we cannot retrain the most sensible self-supervised baseline of ViPC + Resampling + Mixup.

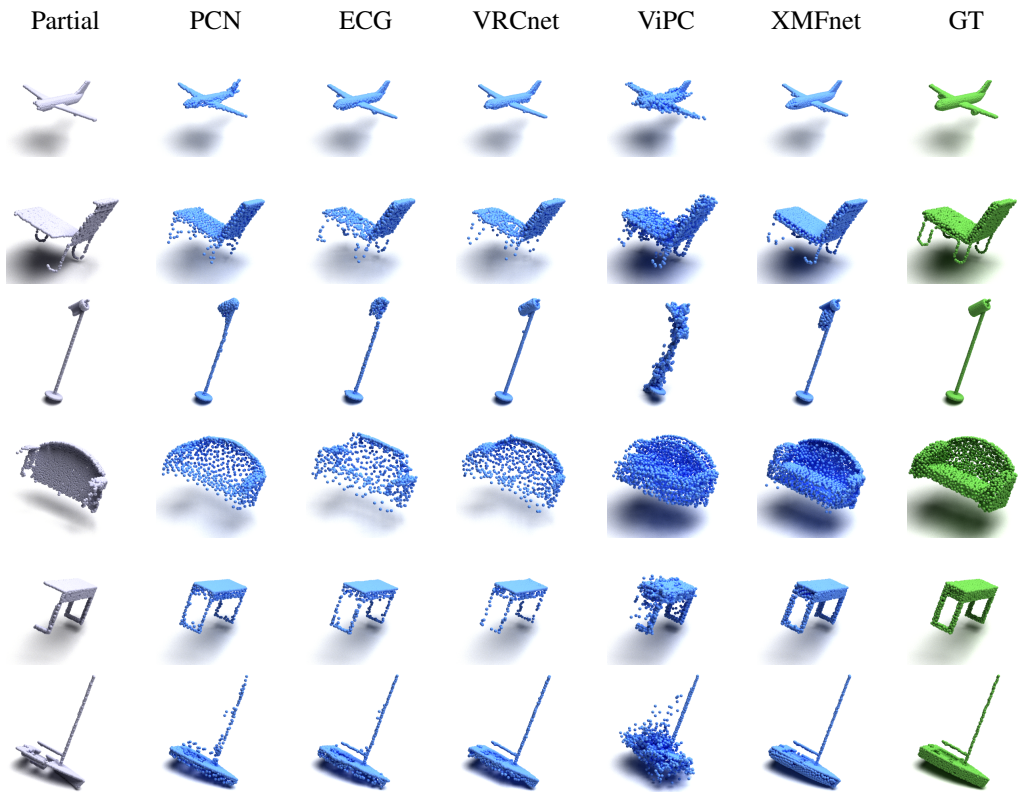

Figure 4: Qualitative comparison of completed point clouds for different classes.

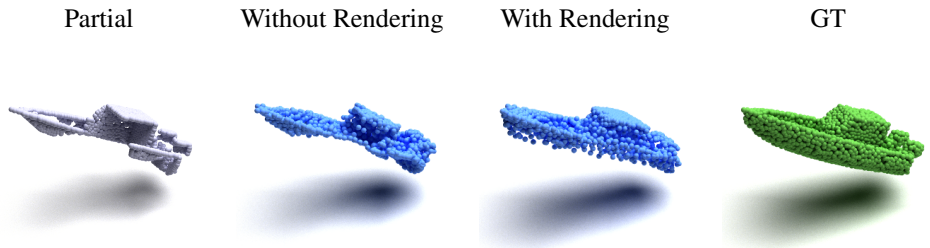

Figure 5: Qualitative visualization of the effect of the proposed self-supervised rendering loss. The sample without rendering has CD $= 7.812$, the one with rendering has CD$= 3.743$.

qualitative and quantitative point of view. A qualitative comparison between the self-supervised strategy with and without the rendering module can be visualized in Fig. 5 and quantitative results are reported in Table 4.

5 Conclusions

In this paper, we explored the topic of point cloud completion guided by an auxiliary image, discovering that effective fusion can be achieved in a latent space via cross-attention. Our method achieves state-of-the-art results on the ShapeNetViPC-Dataset. Moreover, we showed how this setting lends itself to self-supervised learning where the image can be used for supervision via a differentiable rendering approach. The main limitation of our study is the use of a synthetic dataset. Future work will focus on the challenges of the completion problem on real data, such as the presence of acquisition noise, the effect of the background in the auxiliary images, etc..

References

[1] J. Hou, A. Dai, and M. Nießner, "3D-SIS: 3D semantic instance segmentation of RGB-D scans," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4421–4430.

[2] J. Varley, C. DeChant, A. Richardson, J. Ruales, and P. Allen, "Shape completion enabled robotic grasping," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2017, pp. 2442–2447.

[3] Y. Li, L. Ma, Z. Zhong, F. Liu, M. A. Chapman, D. Cao, and J. Li, "Deep learning for lidar point clouds in autonomous driving: A review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 8, pp. 3412–3432, 2020.

[4] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert, "PCN: Point completion network," in *2018 International Conference on 3D Vision (3DV)*. IEEE, 2018, pp. 728–737.

[5] L. P. Tchapmi, V. Kosaraju, H. Rezatofighi, I. Reid, and S. Savarese, "Topnet: Structural point cloud decoder," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 383–392.

[6] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, "A papier-mâché approach to learning 3d surface generation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 216–224.

[7] L. Pan, "ECG: Edge-aware Point Cloud Completion with Graph Convolution," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4392–4398, 2020.

[8] L. Pan, X. Chen, Z. Cai, J. Zhang, H. Zhao, S. Yi, and Z. Liu, "Variational relational point completion network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8524–8533.

[9] X. Yu, Y. Rao, Z. Wang, Z. Liu, J. Lu, and J. Zhou, "PoinTr: Diverse point cloud completion with geometry-aware transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 498–12 507.

[10] X. Wang, M. H. Ang, and G. Lee, "Cascaded refinement network for point cloud completion with self-supervision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[11] H. Mittal, B. Okorn, A. Jangid, and D. Held, "Self-Supervised Point Cloud Completion via Inpainting," in *British Machine Vision Conference*, 2021.

[12] X. Zhang, Y. Feng, S. Li, C. Zou, H. Wan, X. Zhao, Y. Guo, and Y. Gao, "View-guided point cloud completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 890–15 899.

[13] M. Sung, V. G. Kim, R. Angst, and L. Guibas, "Data-driven structural priors for shape completion," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 6, pp. 1–11, 2015.

[14] N. J. Mitra, L. J. Guibas, and M. Pauly, "Partial and approximate symmetry detection for 3d geometry," *ACM Transactions on Graphics (TOG)*, vol. 25, no. 3, pp. 560–568, 2006.

[15] M. Pauly, N. J. Mitra, J. Wallner, H. Pottmann, and L. J. Guibas, "Discovering structural regularity in 3d geometry," *ACM Transactions on Graphics (TOG)*, vol. 27, no. 3, pp. 1–11, 2008.

[16] R. Schnabel, P. Degener, and R. Klein, "Completion and reconstruction with primitive shapes," in *Computer Graphics Forum*, vol. 28, no. 2. Wiley Online Library, 2009, pp. 503–512.

[17] B. Fei, W. Yang, W. Chen, Z. Li, Y. Li, T. Ma, X. Hu, and L. Ma, "Comprehensive review of deep learning-based 3d point clouds completion processing and analysis," *arXiv preprint arXiv:2203.03311*, 2022.

[18] M. Liu, L. Sheng, S. Yang, J. Shao, and S.-M. Hu, "Morphing and sampling network for dense point cloud completion," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 07, 2020, pp. 11 596–11 603.

[19] A. Dai, C. Ruizhongtai Qi, and M. Nießner, "Shape completion using 3D-encoder-predictor CNNs and shape synthesis," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5868–5877.

[20] X. Han, Z. Li, H. Huang, E. Kalogerakis, and Y. Yu, "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*, 2017, pp. 85–93.

[21] H. Xie, H. Yao, S. Zhou, J. Mao, S. Zhang, and W. Sun, "GRNet: Gridding residual network for dense point cloud completion," in *European Conference on Computer Vision*. Springer, 2020, pp. 365–381.

[22] P. Xiang, X. Wen, Y.-S. Liu, Y.-P. Cao, P. Wan, W. Zheng, and Z. Han, "Snowflakenet: Point cloud completion by snowflake point deconvolution with skip-transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5499–5509.

[23] Z. Huang, Y. Yu, J. Xu, F. Ni, and X. Le, "Pf-net: Point fractal network for 3d point cloud completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7662–7670.

[24] A. Alliegro, D. Valsesia, G. Fracastoro, E. Magli, and T. Tommasi, "Denoise and contrast for category agnostic shape completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4629–4638.

[25] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions on Graphics (TOG)*, 2019.

[26] J. Lee, I. Lee, and J. Kang, "Self-attention graph pooling," in *International conference on machine learning*. PMLR, 2019, pp. 3734–3743.

[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[28] K. Zhang, X. Yang, Y. Wu, and C. Jin, "Attention-based transformation from latent features to point clouds," *arXiv preprint arXiv:2112.05324*, 2021.

[29] O. Wiles, G. Gkioxari, R. Szeliski, and J. Johnson, "Synsin: End-to-end view synthesis from a single image," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7467–7477.

[30] E. Insafutdinov and A. Dosovitskiy, "Unsupervised learning of shape and pose with differentiable point clouds," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[31] A. Grabner, Y. Wang, P. Zhang, P. Guo, T. Xiao, P. Vajda, P. M. Roth, and V. Lepetit, "Geometric correspondence fields: Learned differentiable rendering for 3d pose refinement in the wild," in *European Conference on Computer Vision*. Springer, 2020, pp. 102–119.

[32] A. Palazzi, L. Bergamini, S. Calderara, and R. Cucchiara, "End-to-end 6-dof object pose estimation through differentiable rasterization," in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018, pp. 0–0.

[33] C. Xie, C. Wang, B. Zhang, H. Yang, D. Chen, and F. Wen, "Style-based point generator with adversarial rendering for point cloud completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4619–4628.

[34] T. Wu, L. Pan, J. Zhang, T. Wang, Z. Liu, and D. Lin, "Balanced Chamfer Distance as a Comprehensive Metric for Point Cloud Completion," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[35] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An Information-Rich 3D Model Repository," Stanford University — Princeton University — Toyota Technological Institute at Chicago, Tech. Rep. arXiv:1512.03012 [cs.GR], 2015.

[36] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[37] N. Ravi, J. Reizenstein, D. Novotný, T. Gordon, W.-Y. Lo, J. Johnson, and G. Gkioxari, "Accelerating 3d deep learning with pytorch3d," *SIGGRAPH Asia 2020 Courses*, 2020.

[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Internation Conference on Learning Representations*, 2015.

[39] Y. Yang, C. Feng, Y. Shen, and D. Tian, "FoldingNet: Point Cloud Auto-Encoder via Deep Grid Deformation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 206–215.

[40] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun, "Tanks and temples: Benchmarking large-scale scene reconstruction," *ACM Transactions on Graphics (TOG)*, vol. 36, pp. 1 – 13, 2017.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] We discussed the main limitations in Section 5.
   (c) Did you discuss any potential negative societal impacts of your work? [No] Our work is foundational research, we do not see a direct connection towards malicious usage of shape completion method presented in the paper.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] . We included a preliminary version of the code in the supplementary material, we will release it upon the paper accepted.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 4

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] We followed the typical evaluation setting of point cloud completion papers: we reported the best results.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 4

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

   (a) If your work uses existing assets, did you cite the creators? [Yes]

   (b) Did you mention the license of the assets? [Yes]

   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]