Shape Completion with Points in the Shadow

Bowen Zhang
Xi'an Jiaotong University
Xi'an, China
zhangbowen.wx@gmail.com

Xi Zhao*
Xi'an Jiaotong University
Xi'an, China
xi.zhao@mail.xjtu.edu.cn

He Wang
University of Leeds
Leeds, UK
realcrane@gmail.com

Ruizhen Hu
Shenzhen University
Shenzhen, China
ruizhen.hu@gmail.com

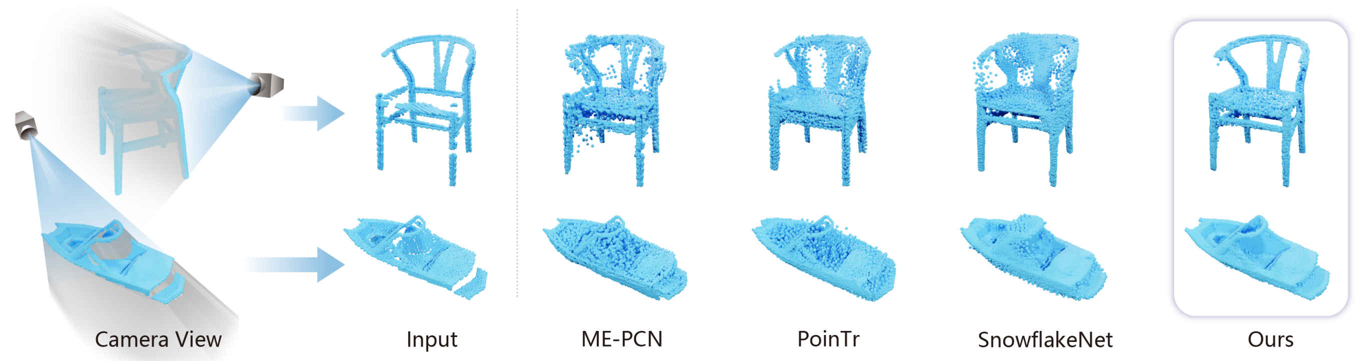

Figure 1: Given a single-view partial scan, our method considers the volume that is shadowed by the observed points and generates more complete and clean results compared with ME-PCN [Gong et al. 2021], PoinTr [Yu et al. 2021] and SnowflakeNet [Xiang et al. 2021].

ABSTRACT

Single-view point cloud completion aims to recover the full geometry of an object based on only limited observation, which is extremely hard due to the data sparsity and occlusion. The core challenge is to generate plausible geometries to fill the unobserved part of the object based on a partial scan, which is under-constrained and suffers from a huge solution space. Inspired by the classic shadow volume technique in computer graphics, we propose a new method to reduce the solution space effectively. Our method considers the camera a light source that casts rays toward the object. Such light rays build a reasonably constrained but sufficiently expressive basis for completion. The completion process is then formulated as a point displacement optimization problem. Points are initialized at the partial scan and then moved to their goal locations with two types of movements for each point: directional movements along the light rays and constrained local movement for shape refinement. We design neural networks to predict the ideal point movements to get the completion results. We demonstrate that our method is accurate, robust, and generalizable through exhaustive evaluation and comparison. Moreover, it outperforms state-of-the-art methods qualitatively and quantitatively on MVP datasets.

CCS CONCEPTS

• **Computing methodologies → Computer vision**.

KEYWORDS

point cloud, completion, view constrain, neural networks

ACM Reference Format:
Bowen Zhang, Xi Zhao, He Wang, and Ruizhen Hu. 2022. Shape Completion with Points in the Shadow. In *SIGGRAPH Asia 2022 Conference Papers (SA '22 Conference Papers), December 6–9, 2022, Daegu, Republic of Korea.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3550469.3555389

1 INTRODUCTION

3D point cloud has become popular with the fast development of depth cameras and 3D scanning devices. It has been employed to acquire geometries from small objects [Zeisl et al. 2013] to city-scale infrastructure [Lai et al. 2011] in various applications such as SLAM and self-driving cars. Although it is possible to acquire full

*Corresponding author: Xi Zhao, School of Computer Science and Technology, Xi'an Jiaotong University (zhaoxi.jade@gmail.com)

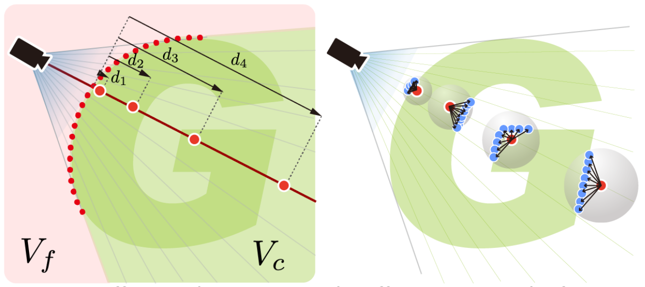

Figure 2: The completion along rays. (a)we duplicate the observed point multiple times to get the larger red points, and move them along the ray with offset d_i. (b) we split each red point multiple times to get the blue points and spread them to a neighborhood of the red point (the transparent circle).

observations via panoramic scanning in some scenarios [Kurkela et al. 2021], very often, it requires a completion step as only partial scans can be obtained in many cases [Fei et al. 2022].

Point cloud completion aims to generate complete shapes from a partial point cloud. The partial point cloud can be any incomplete data, but a prevalent type is the partial scan, especially the single view scan. Very recently, 3D models have been used to synthesize single view partial scans for point cloud completion. The well-known completion benchmarks, such as ShapeNet [Chang et al. 2015], Completion3D [Tchapmi et al. 2019], MVP [Pan et al. 2021], et al. are all simulated from ShapeNet models and provide single-view partial scans and the ground-truth point cloud for machine learning. Based on these datasets, research on point cloud completion, especially deep learning-based techniques, has grown rapidly in recent years. However, existing methods almost entirely rely on network's capacity to learn the correlation between the observed partial scans and the complete ground truth. This leads to an under-constrained solution space for filling in the missing geometry. This uncontrolled data-fitting can generate out-of-distribution geometries far from the ground truth or average over multiple possible missing geometries related to the same input scan.

Our insight is that the solution space should be restricted and more discriminative by leveraging prior knowledge. A key observation is that if we know the camera pose for the partial scan, the space for the completion can be reduced by explicitly modeling the spatial relation between the camera and the observed geometry. Inspired by a computer graphics technique called shadow volume [Crow 1977], we can consider the camera as a point light facing the object, so that the unobserved geometry should be inside the volume that is shadowed by the object (the left column of Fig.1). This volume is naturally a reduced solution space for the missing geometry. To complete the geometry, we assume the observed part can shoot particles into the shadow volume, and they will eventually land on the missing surface of the object to help the shape completion. However, the shadow volume has an arbitrary shape, so we need to construct a convenient basis for possible particle movements.

We use a 2D example (Fig.2) to demonstrate our main idea. We define the shadowed volume as "candidate volume" (V_c) for the completion (the green area in Fig.2(a)). Inversely, the volume other than the candidate volume is the "forbidden volume" (V_f) for completion. Given a camera pose and the observed points P, We can approximate the candidate volume V_c by casting rays from the camera to each observed point. These rays span the entire space of V_c. We consider each ray as a completion basis, and apply the completion process along each ray with the following two steps. First, we put multiple duplicated points on each observed point and move them along the ray with distance d_i to produce the initial guesses of the unobserved points (the red points). The second step is refinement. We split each red point into multiple blue points. The final locations of the blue points are computed by first duplicating the red points multiple times and then spreading them within the neighborhood of the red point. We do not simply move the blue points around locally, but constrain the moving range adaptively for each point. The points farther away from the observed points have larger moving ranges.

Traditionally, Chamfer Distance(CD) is widely used for evaluating point cloud completion quality. However, CD is a single value that is not sufficiently discriminative and fine-grained for detailed completion analysis. We argue that completion should be assessed on observed and unobserved parts separately because the two parts reflect different aspects of completion. The reconstruction of the observed part shows the *fidelity* while the missing part shows the *plausibility*. Therefore, in addition to the traditional metrics such as CD and F-score, we employ a more fine-grained CD metric that separately evaluates the reconstruction quality of the observed and unobserved parts.

Overall, our method can reduce the solution space with the guidance of the camera view. Furthermore, with the two types of movements inside the reduced solution space, our method can retain the observed geometric details and generate the missing part simultaneously. Exhaustive evaluation and comparison show that our method outperforms the state-of-the-art methods both quantitatively and qualitatively. Formally, our contributions include:

(1) A novel formulation of single-view point cloud completion with highly-reduced solution space;
(2) An effective two-step completion method based on point movements with constrained direction and range;
(3) An intuitive fine-grained metric that separately evaluates the observed and unobserved parts.

2 RELATED WORK

Among the huge number of point cloud completion methods, the methods based on the point cloud representation and deep learning network are most related to our work. Considering the completion methods from the input's point of view, we can roughly classify the existing methods into two categories: one is the methods that only use the input implicitly. This type of method produces typically the final results based on the guidance of a global feature. The other is the methods that retain the input explicitly for producing the results. Besides, as we consider the completion process a displacement between point sets, we also introduce the displacement-related techniques for the point cloud.

2.1 Global Feature Guided Completion

This type of work extracts global features from partial input by using methods such as PointNet [Qi et al. 2017a], and then decodes this feature to generate the completion result. We call it "implicit" because the partial input affects the final results indirectly through the latent feature. Most point cloud completion work during the last few years falls to this type. The early methods such as [Yuan et al. 2019] apply an encoder-decoder network. Tchapmi et al. [Tchapmi et al. 2019] improved the decoder by using a hierarchical structure. Wen et al. [Wen et al. 2020] further apply the skip-attention mechanism to convey geometric information from partial input to the hierarchical decoder to improve the results. In [Xie et al. 2020], the authors use 3D grids as the intermediate representation to reduce the loss of structural and context details during completion.

These works use different ways to enforce the influence of structure and geometry details of partial input, but no matter what techniques they use, the influence of the guidance is limited. Because when generating complete shapes, the solution space is too large.

2.2 Input Points Retained Completion

To maximize the use of the input point cloud, researchers make efforts to retain the input points as much as possible. We classify the strategies of retaining input into two categories: keep all original input points in the result and merge the input into the middle result before refinement.

The most straightforward way to retain the input is to predict the missing part of the shape, and then put the input and generated part together to produce the completion result. The representative works of this kind are PF-Net [Huang et al. 2020] and PoinTr [Yu et al. 2021]. PF-Net uses a multi-scale generating network to generate missing regions. PoinTr uses a transformer encoder-decoder architecture to predict the point proxies for the missing part. The benchmark they use normally contains a random removal of parts from the shape. Although completion for this type of partial data is useful, it is not as general as the completion of a partial scan point cloud.

To deal with the single view partial scan data, Liu et al. [Liu et al. 2019] propose to put the input partial scan and the initial completion results together to further feed to the refinement process. Normally there is also a sub-sampling step to control the number of points generated. Such a "merge and sub-sampling" strategy has become popular for strengthening the influence of input data on the results. It has been used in many works, such as [Gong et al. 2021; Wang et al. 2020; Xia et al. 2021; Xiang et al. 2021; Zhang et al. 2020]. The interesting extension for "merge and sub-sampling" is to use the symmetry axis of the shape, and apply the "mirror" operation before merging the input point cloud [Wang et al. 2020; Xia et al. 2021].

Our method can be roughly considered as the second input retain strategy. However, we merge the input and the generated part differently. By predicting the offset of points along rays, both the observed and unobserved parts are processed uniformly in the system, so there is no need for the traditional "merge and sub-sampling" operation.

2.3 Displacement-based Point Cloud Manipulation

Yin et al. [Yin et al. 2018] propose a one-step deformation to deform one point cloud to another by using a cycle structure network. This method can be generally applied to the transformation between two domains. Another category of work is to use the point-wise displacement for manipulating the articulated point cloud shapes [Yan et al. 2020; Yi et al. 2018]. In [Yi et al. 2018], the pair-wise displacement is used to find the correspondence between points from two sets and further predict the segmentation and part-based motion. RPM-net [Yan et al. 2020] predicts a temporal sequence of point-wise displacement for the input shape to infer movable parts and generate motions.

PMP-Net [Wen et al. 2021], PMP-Net++ [Wen et al. 2022] and Front2Back [Yao et al. 2020] are more related to our work as they aims to do the completion. PMP-Net and PMP-Net++ enable multi-step movement of the input point set and use the least total moving distance loss to mimic the earth mover distance. Compared to them, we use viewpoint to reduce the solution space for the point's displacement problem, and constrain the movement of points with the rays. Front2Back [Yao et al. 2020] directly uses occlusion masks from the input and predicts the points on the "other side". It has difficulty in dealing with objects with complex structures as they cannot be simply described by opposite orthographic views. In contrast, our method is more flexible as we can generate different numbers of new points along the rays to better represent the geometry.

3 OUR METHOD

3.1 Problem Formulation

We formulate the completion process as a point displacement optimization problem. First, points are initialized at the partial scan. Then they are moved to their goal locations with two types of movements shown in Fig.2. We design a network which consists of two main modules for such a completion process: the offset prediction module is designed to predict the movement along rays, then the offset-constrained refinement module is designed to estimate the local refinement movements. We show the overall architecture of our method in Fig.3.

The input of our system contains a camera location, a partial scan and the rays. They are constructed as follows: assuming the camera is located at a 3D position $Cam = (x_c, y_c, z_c)$, and faces the object center. From such a camera configuration, we can get a partial scan containing N 3D points $P = \{p^i\}, (i = 1, ..., N)$. We define a set of rays along each p^i, which is represented by a vector from Cam to p^i: $R = \{\vec{r_i}\} = \{p^i - Cam\}, (i = 1, ..., N)$.

Offset prediction. This module computes the initial completion result P_o from partial scan P. We duplicate the input points multiple times and move these points along the rays with the predicted offset to get P_o. We can describe this process with:

$$p_o^{i,l} = p^i + \mathcal{F}_d^l(p^i) \cdot \vec{r_i} \tag{1}$$

where $i = 1, ..., N; l = 1, ..., L$. The function $\mathcal{F}_d : \mathbb{R}^3 \to \mathbb{R}$ computes L offsets for each camera ray. We provide more details of this module in Section 3.2.

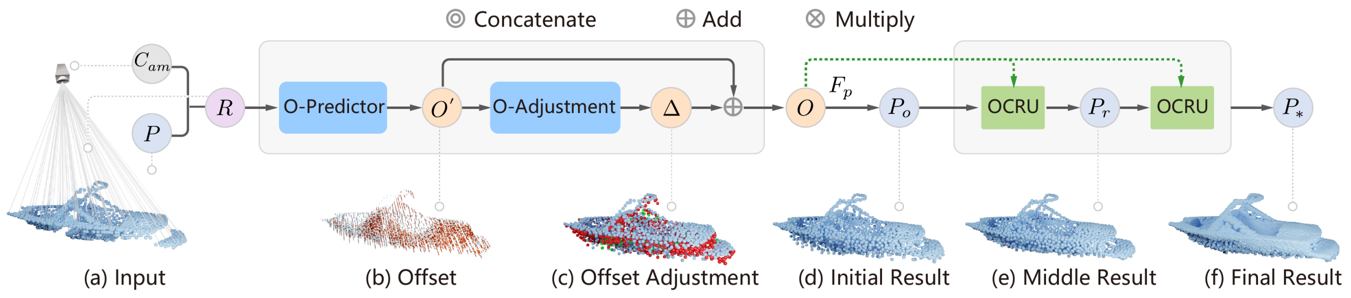

Figure 3: The overview of our method. Given the camera location and partial scan, we compute a batch of rays that is the network's input(a). For each ray, we first predict offset O' that is visualized by lines in (b), and then predict an adjustment for each offset that is visualized in (c), where green points represent positive adjustment; red points represent negative adjustment. With the initial completion result(d), we further apply a two-step refinement to get a smoother result (e) and final result (f).

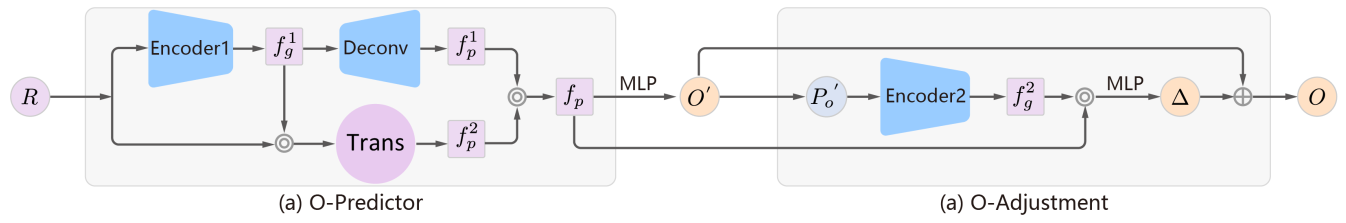

Figure 4: The structure of the offset prediction module. It contains the initial offset prediction (O-Predictor) and the offset adjustment (O-Adjustment).

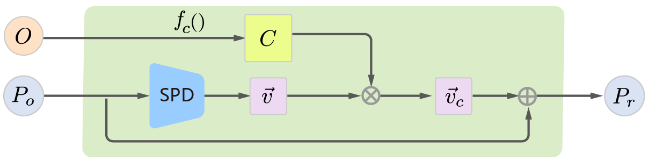

Figure 5: The structure of offset constrained refinement unit (OCRU). We use function f_c to compute the value C, which is used to constrain the range of movement $\vec{v}_c$ computed by SPD.

Offset-constrained refinement. This module computes the final result P_* from the initial completion result P_o. We split each point of P_o into multiple points, and move them within a local neighborhood to improve the local geometry details. The equation of this step is:

$$p_*^{m,k} = p_o^m + \mathcal{F}_{\vec{v}}^k(p_o^m) \tag{2}$$

where $m = 1, ..., N_o, k = 1, ..., L_*$. N_o is the number of points in P_o. The function $\mathcal{F}_{\vec{v}} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ is for computing L_* local movements, in which the local movement of each point is constrained by its offset along the ray computed in the previous module. We provide more details of this module in Section 3.3.

3.2 Offset Prediction

To estimate the offset mapping $\mathcal{F}_d$ defined in Equation 1, we design a network which contains two parts: initial offset prediction (O-Predictor) and the offset adjustment (O-Adjustment). The network structure is shown in Fig.4.

O-Predictor. Since R contains not only the point positions, but also the orientations depending on the camera, modeling the correlations of both positions and orientations is crucial. We design a network module named O-Predictor to model the correlations for predicting offsets. The input of the network is ray R and the output is L offsets along each ray, denoted as $O' = \{d^{il}\}(i = 1, ..., N; l = 1, ..., L)$, where each offset d^{il} is a distance from p^i to a new point along the i_{th} ray in the positive direction.

Given rays R, we first extract the global correlations between all $\vec{r}_i \in R$ in a latent space using PointNet++ [Qi et al. 2017b] to get a global feature f_g^1, then transform the correlation into per-ray features f_p^1, both of which consider the orientation and position in R simultaneously. In parallel, we concatenate R and f_g^1 and pull it through a skip Transformer [Xiang et al. 2021] where the attention is solely based on the orientation similarity between rays, to get the per-ray orientation correlation feature f_p^2. Finally, f_p^1 and f_p^2 are concatenated with rays R, partial scan P and global feature f_g^1 to compute a feature f_p which is further fed into a MLP to compute the output offset O'. We apply Relu to the output of MLP to make sure all the offset values are non-negative.

O-Adjustment. O-Predictor alone can only provide rough offset estimation because learning the complex distribution of offsets for different shapes is non-trivial. We design an O-Adjustment module to improve the offset precision by modifying the predicted offset values. From the example shown in Fig.3, we can see that although the points at the top of the watercraft move too much away from the bar after the O-Predictor step(b), the O-Adjustment module move them back by assigning these points with negative adjustment(c).

The structure of the O-Adjustment module is as follows. After getting the offsets O' from the last module, we compute a first step completion results $P_{o'}$ by moving each input point with this offset along its ray. This process can be presented as $p^i + d^{il} \cdot \vec{r}_i$. Then we use the PCN encoder [Yuan et al. 2019] to extract the global feature f_g^2, which is concatenated with feature f_p and fed to MLP to predict an adjustment for each offset, denoted as Δ. The final offset O is the sum of O' and Δ . We again apply the Relu function to O to make sure the output offset is non-negative values. Finally, we convert the output of the offset O to the initial completion result P_o by moving points along rays.

3.3 Offset Constrained Refinement

The main problems with the initial result P_o are the non-uniform density and lack of local geometry details. The points in the input partial scan P may distribute unevenly due to the camera configuration relative to the shape, which results in non-uniformed distribution of corresponding rays. In addition, the points farther away from the camera are sparser because of the perspectivity of the rays. Therefore, refinement is needed to generate fine-grained geometry structures.

Preliminary experiments show that classic refinement methods such as folding operation [Yang et al. 2018] and the snowflake point deconvolution (SPD) [Xiang et al. 2021] tend to fill holes. However, if the holes are part of the object, filling them can damage the topology and structure of the shape. Besides, their refinement process cannot control the movement ranges of the points. For example, a point may move to the invalid space of the partial scan P, which leads to a noisy effect.

To address this problem, we propose an offset-constrained refinement module (Fig.5) to constrain the movement range during the refinement. We first apply a farthest point sampling (PFS) for P_o to make the points distribute more evenly. Then the result is further fed to the offset constrained refinement module, which contains two layers of refinement. Each layer is an offset-constrained refinement unit (OCRU) designed based on SPD. Here the output of SPD is a list of 3D movement vectors. We multiply the offset constraint value to each dimension of the SPD's output to apply the constraint. More specifically, the offset constraint value for j_{th} point in $u_{th}(u = 1, 2)$ OCRU is computed by:

$$C_u^j = f_c(O_j, u) = (O_j/2 + 0.03)/\alpha^{u-1} \tag{3}$$

where O_j is the overall offset for j_{th} point computed by the offset prediction module. α is a scale coefficient. In all the experiments, we set $\alpha = 1.5$.

With this constraint, the local details of the partial scan are kept. In the final result P_*, the points close to partial scan P have tiny offsets (the offsets are nearly zero), so the movement range of such points in the refinement module is very small. As a result, the observed geometry represented by such points is retained. The boundary points of the partial scan are also part of the observed points, so the generated points stem from them should not move far too. As a result, the geometry in the boundary area is also neat and clean. The movement range is more extensive for points generated around the far end of the rays. So points have more freedom to move during refinement. With such a strategy, we effectively retain the observed geometry and improve the unobserved part's quality.

3.4 Loss Function

We apply the shape loss $\mathcal{L}_{CD}$, which is Chamfer Distance (CD) to measure the shape difference between the completed results and the ground truth. The CD between point cloud P_1 and P_2 is computed by:

$$\mathcal{L}_{CD}(P_1, P_2) = \sum_{p \in P_1} \min_{q \in P_2} ||p - q||_2^2 + \sum_{q \in P_2} \min_{p \in P_1} ||q - p||_2^2 \tag{4}$$

During the training, we first pre-train the offset prediction module with loss function $\mathcal{L}_{CD}(P_{o'}, P_{\mathrm{gt}_1}) + \mathcal{L}_{CD}(P_o, P_{\mathrm{gt}_1})$, where the $P_{o'}$ and P_o are computed by the offset prediction module. Then, we keep the parameters in the offset prediction module fixed and pre-train the refinement module with loss function $\mathcal{L}_{CD}(P_r, P_{\mathrm{gt}_2}) + \mathcal{L}_{CD}(P_*, P_{\mathrm{gt}_3})$, where P_r and P_* are the middle and final result of the refinement module. Finally, we train two modules jointly with the same loss function as the second stage. P_{gt_1}, P_{gt_2} and P_{gt_3} are the ground truth point cloud which contains 8192, 2048 and 16384 points respectively. We provide other training details in the supplementary material.

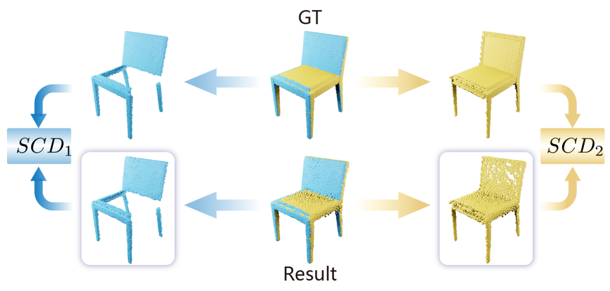

Figure 6: The computation of SCD_1 and SCD_2.

4 EXPERIMENTS

4.1 Dataset

We use the Multi-View Partial point cloud dataset (MVP) [Pan et al. 2021] in our experiments. MVP is computed based on 16 categories of 4000 CAD models. Twenty-six camera locations are sampled for each model to simulate the partial scan. In our experiment, we recompute the MVP dataset to record the camera configuration for each scan. Finally, we use 62400 partial-complete point cloud pairs for training and 41600 pairs for testing.

4.2 Evaluation Metric

We use CD, F-Score [Tatarchenko et al. 2019] and Density-aware CD (DCD) [Wu et al. 2021] to evaluate the completion results. CD

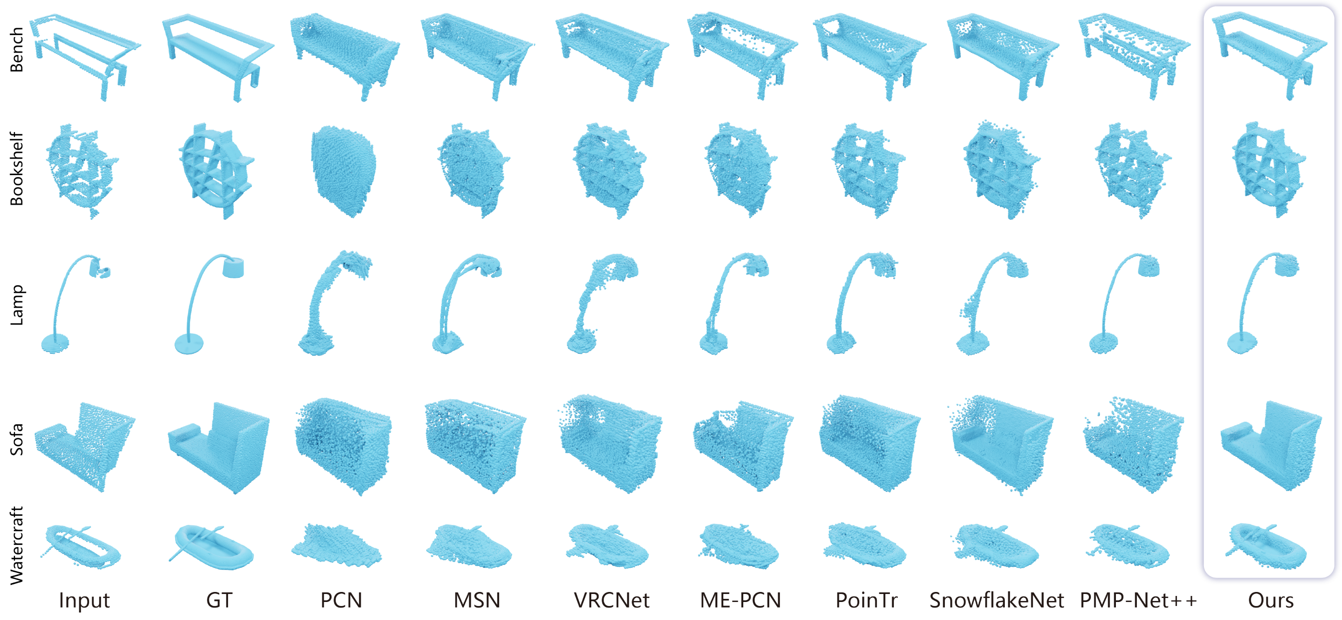

Figure 7: Visualized completion results for comparison on MVP dataset.

is sensitive to outliers but insensitive to the local density. F-score is used as a supplement to *CD* to evaluate the completion results. A higher F-score usually indicates better visual quality. *DCD* is also a metric for evaluating the overall point cloud distance, which is more sensitive to the density distribution of point clouds.

The goal of the completion task naturally includes two parts: retaining the geometry of the partial scan (task1) and generating a reasonable shape for the unobserved part (task2). Existing metrics are indiscriminative to the two parts, which suggests that they lack granularity. To provide sufficient details on the completion quality, we propose a new metric named "split *CD*" (*SCD*) that includes two values, SCD_1 and SCD_2.

To compute SCD_1 and SCD_2, we split the GT point cloud P_{gt} into two parts P^1_{gt} and P^2_{gt} based on the partial input P. More specifically, if a point in P_{gt} is in the neighbourhood of P, then it belongs to P^1_{gt}, otherwise it belongs to P^2_{gt}. For each point p in P, the neighborhood of p is defined as the spatial area within radius r ($r = 0.01$ in our setting), and the neighborhood of P is the union of the neighborhoods of all p in P. We next split the completion results into two parts based on P and P^2_{gt}: for each point p in a result P_*, we find a closest point p' in the point set $\{P, P^2_{\text{gt}}\}$. If p' belongs to P, then p belongs to P^1_*, otherwise P^2_*. Then the split *CD* is computed as $SCD_1 = \mathcal{L}_{CD}(P^1_*, P^1_{\text{gt}})$, $SCD_2 = \mathcal{L}_{CD}(P^2_*, P^2_{\text{gt}})$. This process is visualized in Fig.6.

4.3 Comparison

We exhaustively compare our method with seven baseline methods: PCN [Yuan et al. 2019], MSN [Liu et al. 2019], ME-PCN [Gong et al. 2021], VRCNet [Pan et al. 2021], PoinTr [Yu et al. 2021], SnowflakeNet [Xiang et al. 2021], and PMPNet++ [Wen et al. 2022].

Table 1: Evaluation for each method. ***CD*** **and** ***SCD*** **are multiplied by 10^4.**

Methods	*CD*	F-Score	*DCD*	SCD_1	SCD_2
PCN	5.907	0.617	0.628	5.863	6.322
MSN	4.749	0.682	0.645	2.210	7.757
ME-PCN	4.680	0.662	0.658	2.086	7.341
VRCNet	4.780	0.741	0.539	2.471	6.350
PoinTr	3.882	0.715	0.613	1.709	6.664
PMP-Net++	3.381	0.687	0.696	1.391	6.954
SnowflakeNet	2.696	0.796	0.524	0.951	4.683
Ours	**2.419**	**0.800**	**0.513**	**0.646**	**3.896**

Table 2: Ablation study. ***CD*** **and** ***SCD*** **are multiplied by 10^4.**

Methods	*CD*	F-Score	*DCD*	SCD_1	SCD_2
No GT viewpoint	2.635	0.781	0.538	0.678	4.590
No adjustment	2.523	0.787	0.532	0.706	4.090
No offset constraint	2.730	0.779	0.540	0.827	4.355
Our final	**2.419**	**0.800**	**0.513**	**0.646**	**3.896**

Table 3: Ablation study on the number of points along rays in the offset prediction module. ***CD*** **and** ***SCD*** **are multiplied by 10^4.**

Point number	*CD*	F-Score	*DCD*	SCD_1	SCD_2
2	**2.375**	0.798	0.523	**0.608**	4.066
4 (Ours)	2.419	**0.800**	**0.513**	0.646	**3.896**
6	2.480	0.786	0.535	0.741	3.956

Quantitative comparison. The quantitative results are reported in Table 1. This table lists averaged metric values for all categories of shapes, and more detailed results are in the supplementary material. Overall, our method achieves the best performance in all metrics across all categories of objects. Lower *CD* and higher F-Score values indicate an overall better completion quality. A lower *DCD* value demonstrates that our method does not suffer from unbalanced local density compared with other baseline methods. The *SCD* show a more fine-grained analysis with SCD_1 and SCD_2. PCN has similar scores for the two parts, which suggests that the completion quality for the observed part and the unobserved part are quite similar. For other baseline methods, the SCD_1 values are reduced, but the SCD_2 values do not change much (sometimes even increase). This reveals that these methods implicitly focus on retaining the observed geometry, sometimes at the cost of the completion quality of the unobserved part. One possible reason is that they allocate too many points to input to ensure reconstruction but fewer points for the missing part. Finally, SnowflakeNet does reduce both of these values but not significantly on SCD_2 compared with our method. Nevertheless, our method reduces both *SCD* values, demonstrating that our results are not only overall better, but also have balanced improvement on both the observed and unobserved geometries.

Qualitative comparison. In Fig.7, we visually compare our method with other baseline methods on the MVP dataset. In general, our method can keep the geometric details of the partial input better. First, the partial scan boundary is not blurred during completion, which is crucial for model details such as the holes in the back of the chair, the bookshelve and the neck of the desk lamp. Besides, the concave areas in the input are also kept, such as the sitting area of the watercraft. The holes, boundaries and concave areas are vital geometric/topological features of an object, indicating their functionality or distinguishing them from similar objects. Keeping these details intact in the input is crucial for high-fidelity completion. Next, Our method can capture overall shape variations well across various shapes. It can deal with unique shapes that are rare in the training data. One example is the sofa with a rather asymmetric design, which is statistically rare in the data. However, our method can recognize and capture the overall asymmetry and recover the model precisely. In contrast, all other methods try to make the sofa symmetric during completion.

We show results with panoramic views in the supplementary video and more completion results for all categories in the supplementary material.

4.4 Ablation Studies

Here we evaluate the design of the network of our method. We consider the following ablation versions:

- No GT viewpoint. The main assumption of our method is that the viewpoint configuration is known. We assume that we do not have this information and predict the viewpoint from the partial scan.
- No offset adjustment. We test how our method performs when the offset adjustment step is removed.
- No offset constraint. We test how our method performs if we do the refinement without the offset constrain.

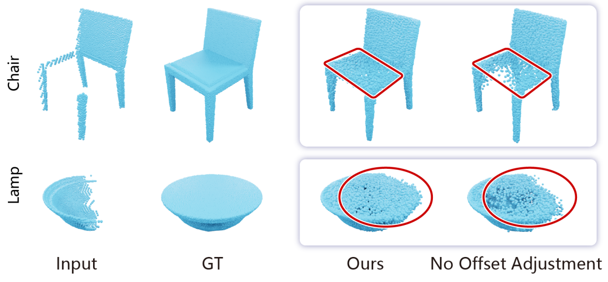

Figure 8: Ablation study: compare our method with the version that has no offset adjustment step.

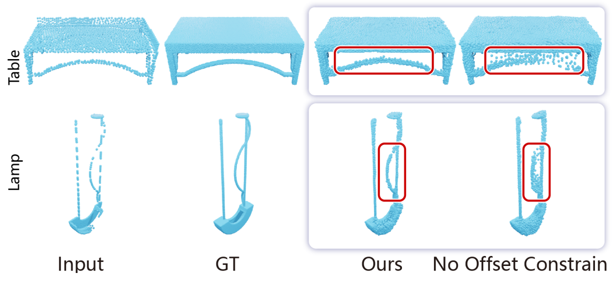

Figure 9: Ablation study: compare our method with the version that does not apply offset constrain during refinement.

The quantitative results are shown in Table 2. We can see that the final version of the method has the overall best performance.

Without the viewpoint information, the quality of the final results is getting slightly worse. However, the average *CD* and *SCD* values are still better than all other baseline methods. Note that the prediction of viewpoint from partial input is not the focus of this paper. We found that with the imperfect viewpoint, our method still works and can produce results with satisfactory quality. The detail of the viewpoint prediction method and related experimental results can be found in the supplementary material.

When there is no offset adjustment step, the performance of our method drops. Some points cannot move to expected locations without the offset adjustment with a single step. Then the network tends to assign a small offset to points, so fewer points are generated in the missing area. This is why the SCD_2 is getting worse. In Fig.8 we can see that the one-step offset results look crude: there are holes (such as the chair) or incomplete parts (such as the half-size lamp) in the unobserved area. With the offset adjustment step, the completion results $P_{o'}$ and P_o are improved simultaneously, and so are the final completion results.

When there is no offset constraint in the refinement module, the performance of our method also drops. We can see that the SCD_1 value increases the most compared with other alternative solutions. Without the constraints, the points can move in larger ranges during refinement. Such movement especially makes the completion of the observed part worse because the local details tend to be blurred. We can see such effects in Fig.9. Without the constraints, the area under the table surface and the details of the lamp are blurred, while with the constraints, we get very neat and clean local details.

The number of points that move along each ray is also important for our method. We evaluate how many points should move along each ray during the completion. Theoretically, too few points may not be enough to represent an initial complete result, because the rays can be sparse. However, too many points may cause unnecessary fillings inside the object, leading to larger shape loss and higher computation complexity. So we try different numbers of points and compute the evaluation values for each case. The results are shown in Table 3. We found that using four points gets the overall best results with higher DCD, SCD_2 and F-score. Therefore, we use four points when implementing our experiments.

We also test our method with ablations in the offset prediction and adjustment steps regarding the transformer and the feature extraction. The details of these experiments can be found in the supplementary material.

5 CONCLUSION

This paper proposes a new framework for single view point cloud completion. The key insight is that the vast solution space can be drastically reduced when using shadow volumes and camera rays as the solution basis. This construction leads to two types of completion strategies by moving points for completion: one is moving points along rays, and the other is the local constrained movement for refinement. Moreover, we propose a more fine-grained metric for evaluation: the split CD (SCD), with which we can analyze the completion quality for the observed part and the unobserved part separately and jointly. Extensive evaluation and comparison demonstrate the superiority of our method over the current SOTA methods.

Limitations and future work. Our method relies on the details of the partial scan. If the quality of the scan is low, especially when the main structure or significant part of the shape is not captured, our method might have difficulties recovering the correct local details. Two representative examples are shown and discussed in the supplementary material. In the future, instead of generating one geometry, we will aim to generate a distribution of possible geometries where more prior knowledge or user's preferences can be considered. As mesh is a more useful data type, predicting implicit function along camera rays for reconstruction is also worth further exploration.

ACKNOWLEDGMENTS

This work was supported in part by National Natural Science Foundation of China (62072366, 61872250), Key R&D project of Shaanxi Province (2021QFY01-03HZ), China Postdoctoral Science Foundation Funded Project (2020M673407), Guangdong Natural Science Foundation (2021B1515020085) and Shenzhen Science and Technology Program (RCYX20210609103121030).

REFERENCES

Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. 2015. ShapeNet: An Information-Rich 3D Model Repository. *arXiv:1512.03012 [cs]* (Dec. 2015). http://arxiv.org/abs/1512.03012 arXiv: 1512.03012.

Franklin C. Crow. 1977. Shadow algorithms for computer graphics. *ACM SIGGRAPH Computer Graphics* 11, 2 (1977), 242–248. https://doi.org/10.1145/965141.563901

Ben Fei, Weidong Yang, Wenming Chen, Zhijun Li, Yikang Li, Tao Ma, Xing Hu, and Lipeng Ma. 2022. Comprehensive Review of Deep Learning-Based 3D Point Cloud Completion Processing and Analysis. *arXiv:2203.03311 [cs]* (March 2022). http://arxiv.org/abs/2203.03311 arXiv: 2203.03311 version: 2.

Bingchen Gong, Yinyu Nie, Yiqun Lin, Xiaoguang Han, and Yizhou Yu. 2021. ME-PCN: Point Completion Conditioned on Mask Emptiness. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 12488–12497.

Zitian Huang, Yikuan Yu, Jiawen Xu, Feng Ni, and Xinyi Le. 2020. PF-Net: Point Fractal Network for 3D Point Cloud Completion. *arXiv:2003.00410 [cs]* (March 2020). http://arxiv.org/abs/2003.00410 arXiv: 2003.00410.

Matti Kurkela, Mikko Maksimainen, Arttu Julin, Toni Rantanen, Juho-Pekka Virtanen, Juha Hyyppä, Matti Tapio Vaaja, and Hannu Hyyppä. 2021. Utilizing a Terrestrial Laser Scanner for 3D Luminance Measurement of Indoor Environments. *Journal of Imaging* 7, 5 (May 2021), 85. https://doi.org/10.3390/jimaging7050085

Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. 2011. A large-scale hierarchical multi-view RGB-D object dataset. In *2011 IEEE International Conference on Robotics and Automation*. 1817–1824. https://doi.org/10.1109/ICRA.2011.5980382 ISSN: 1050-4729.

Minghua Liu, Lu Sheng, Sheng Yang, Jing Shao, and Shi-Min Hu. 2019. Morphing and Sampling Network for Dense Point Cloud Completion. *arXiv:1912.00280 [cs]* (Nov. 2019). http://arxiv.org/abs/1912.00280 arXiv: 1912.00280.

Liang Pan, Xinyi Chen, Zhongang Cai, Junzhe Zhang, Haiyu Zhao, Shuai Yi, and Ziwei Liu. 2021. Variational Relational Point Completion Network. *arXiv:2104.10154 [cs]* (April 2021). http://arxiv.org/abs/2104.10154 arXiv: 2104.10154.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 652–660.

Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. 2017b. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Advances in Neural Information Processing Systems*, Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/hash/d8bf84be3800d12f74d8b05e9b89836f-Abstract.html

Maxim Tatarchenko, Stephan R. Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. 2019. What Do Single-View 3D Reconstruction Networks Learn?. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 3400–3409. https://doi.org/10.1109/CVPR.2019.00352 ISSN: 2575-7075.

Lyne P. Tchapmi, Vineet Kosaraju, Hamid Rezatofighi, Ian Reid, and Silvio Savarese. 2019. TopNet: Structural Point Cloud Decoder. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Long Beach, CA, USA, 383–392. https://doi.org/10.1109/CVPR.2019.00047

Xiaogang Wang, Marcelo H. Ang, and Gim Hee Lee. 2020. Cascaded Refinement Network for Point Cloud Completion. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Seattle, WA, USA, 787–796. https://doi.org/10.1109/CVPR42600.2020.00087

Xin Wen, Tianyang Li, Zhizhong Han, and Yu-Shen Liu. 2020. Point Cloud Completion by Skip-Attention Network With Hierarchical Folding. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Seattle, WA, USA, 1936–1945. https://doi.org/10.1109/CVPR42600.2020.00201

Xin Wen, Peng Xiang, Zhizhong Han, Yan-Pei Cao, Pengfei Wan, Wen Zheng, and Yu-Shen Liu. 2021. PMP-Net: Point Cloud Completion by Learning Multi-step Point Moving Paths. *arXiv:2012.03408 [cs]* (June 2021). http://arxiv.org/abs/2012.03408 arXiv: 2012.03408.

Xin Wen, Peng Xiang, Zhizhong Han, Yan-Pei Cao, Pengfei Wan, Wen Zheng, and Yu-Shen Liu. 2022. PMP-Net++: Point Cloud Completion by Transformer-Enhanced Multi-step Point Moving Paths. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022), 1–1. https://doi.org/10.1109/TPAMI.2022.3159003

Tong Wu, Liang Pan, Junzhe Zhang, Tai Wang, Ziwei Liu, and Dahua Lin. 2021. Density-aware Chamfer Distance as a Comprehensive Metric for Point Cloud Completion. (Nov. 2021). https://arxiv.org/abs/2111.12702v1

Y. Xia, Y. Xia, W. Li, R. Song, K. Cao, and U Stilla. 2021. ASFM-Net: Asymmetrical Siamese Feature Matching Network for Point Completion. (2021). https://doi.org/10.48550/arXiv.2104.09587

Peng Xiang, Xin Wen, Yu-Shen Liu, Yan-Pei Cao, Pengfei Wan, Wen Zheng, and Zhizhong Han. 2021. SnowflakeNet: Point Cloud Completion by Snowflake Point Deconvolution with Skip-Transformer. *arXiv:2108.04444 [cs]* (Oct. 2021). http://arxiv.org/abs/2108.04444 arXiv: 2108.04444.

Haozhe Xie, Hongxun Yao, Shangchen Zhou, Jiageng Mao, Shengping Zhang, and Wenxiu Sun. 2020. GRNet: Gridding Residual Network for Dense Point Cloud Completion. *arXiv:2006.03761 [cs, eess]* (July 2020). http://arxiv.org/abs/2006.03761 arXiv: 2006.03761.

Z. Yan, R. Hu, X Yan, L. Chen, O Van Kaick, H. Zhang, and H. Huang. 2020. RPM-Net: Recurrent Prediction of Motion and Parts from Point Cloud. (2020). https://doi.org/10.1145/3355089.3356573

Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. 2018. FoldingNet: Point Cloud Auto-encoder via Deep Grid Deformation. *arXiv:1712.07262 [cs]* (April 2018). http://arxiv.org/abs/1712.07262 arXiv: 1712.07262.

Yuan Yao, Nico Schertler, Enrique Rosales, Helge Rhodin, Leonid Sigal, and Alla Sheffer. 2020. Front2back: Single view 3d shape reconstruction via front to back prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 531–540.

Li Yi, Haibin Huang, Difan Liu, Evangelos Kalogerakis, Hao Su, and Leonidas Guibas. 2018. Deep Part Induction from Articulated Object Pairs. *Acm Transactions on Graphics* 37, 6 (2018), 1–15. https://doi.org/10.1145/3272127.3275027

Kangxue Yin, Hui Huang, Daniel Cohen-Or, and Hao Zhang. 2018. P2P-NET: bidirectional point displacement net for shape transform. *ACM Transactions on Graphics* 37, 4 (July 2018), 1–13. https://doi.org/10.1145/3197517.3201288

Xumin Yu, Yongming Rao, Ziyi Wang, Zuyan Liu, Jiwen Lu, and Jie Zhou. 2021. PoinTr: Diverse Point Cloud Completion with Geometry-Aware Transformers. *arXiv:2108.08839 [cs]* (Aug. 2021). http://arxiv.org/abs/2108.08839 arXiv: 2108.08839.

Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. 2019. PCN: Point Completion Network. *arXiv:1808.00671 [cs]* (Sept. 2019). http://arxiv.org/abs/1808.00671 arXiv: 1808.00671.

Bernhard Zeisl, Kevin Köser, and Marc Pollefeys. 2013. Automatic Registration of RGB-D Scans via Salient Directions. In *2013 IEEE International Conference on Computer Vision*. 2808–2815. https://doi.org/10.1109/ICCV.2013.349 ISSN: 2380-7504.

W. Zhang, Q. Yan, and C. Xiao. 2020. Detail Preserved Point Cloud Completion via Separated Feature Aggregation. (2020). https://doi.org/10.1007/978-3-030-58595-2_31