TAMECloud: Test-time Adaptation with MEta-auxiliary Optimization for Point Cloud Completion

Author Name
Affiliation
email@example.com

Abstract

Point cloud completion is crucial for improving the accuracy and utility of 3D reconstruction in applications such as robotics and augmented reality. However, existing methods often face challenges when generalizing to novel objects or adapting to diverse patterns of incompleteness that arise in real-world scenarios. To overcome this limitation, we present TAMECloud, a novel framework that leverages **t**est-time **a**daptation with **me**ta-auxiliary optimization for point **cloud** completion. Unlike conventional approaches that rely on static models, our method employs meta-auxiliary learning, comprising a primary completion task and two self-supervised auxiliary tasks, each specifically designed to enhance point cloud completion. By eliminating redundant structures and sharing weights both between the primary and auxiliary tasks, as well as among the auxiliary tasks themselves, TAMECloud streamlines the network and maximizes weight updates, enhancing generalization for specific point cloud input. Furthermore, an auto-weighted loss mechanism dynamically balances the contributions of auxiliary losses, mitigating the risk of negative transfer and optimizing multi-task performance. Experimental results demonstrate that TAMECloud outperforms state-of-the-art point cloud completion methods across various datasets, showcasing fast and accurate adaptation to diverse and unseen scenarios.

1 Introduction

The rapid advancement of 3D sensing technology has enabled a wide range of applications in autonomous driving [Yang *et al.*, 2024], robotics [Liu *et al.*, 2024], and augmented reality [Wang *et al.*, 2023]. Among those applications, point cloud representations have become widely adopted, as they are the most common raw data format generated by 3D sensors. However, point clouds captured by 3D sensors are often incomplete due to factors like self-occlusion, limited sensor resolution, and environmental influences such as light reflections or physical obstructions. This inherent incompleteness presents significant challenges for downstream tasks, including object recognition, scene reconstruction, and robotic action planning. To address these challenges, point cloud completion aims to infer the full shape of an object from its partial observations.

Benefiting from the recent advances in deep learning, point cloud completion has made considerable progress. Existing approaches have primarily focused on designing various network architectures that can handle the unordered and unstructured point cloud data. For example, PointNet [Qi *et al.*, 2017a] and its variants [Qi *et al.*, 2017b] have been widely used to build encoder-decoder architectures for the completion task. However, these methods often struggle to preserve fine-grained details, due to reliance on max-pooling operations. More recently, transformer-based models [Yu *et al.*, 2021; Li *et al.*, 2023; Wang *et al.*, 2024] have emerged, offering enhanced global interaction modeling through the attention mechanism.

Despite these advancements, existing methods still struggle to generalize effectively to unseen and novel objects. This limitation is especially critical in real-world applications, where the point clouds come from diverse objects and exhibit varying patterns of incompleteness introduced by different 3D sensors. While one potential solution could be collecting extensive datasets that cover a broad range of objects and missing patterns, this approach is costly, time-consuming, and may not fully address the generalization issue. To tackle these challenges, we propose a shift towards a novel learning paradigm that emphasizes adaptability and robustness.

In this paper, we introduce TAMECloud, a novel network architecture with multi-auxiliary branches, combining meta-auxiliary learning with test-time adaptation to enable effective point cloud completion across diverse and unseen scenarios. Specifically, the primary task ($Pri.$) focuses on completing incomplete point clouds by mapping partial observations to their full 3D structures. Moreover, we design two auxiliary tasks: Aux_{mae} which utilizes a Masked Autoencoder to reconstruct the original point cloud from a partially masked input, enhancing the model's ability to understand the underlying geometry; Aux_{ups} which performs upsampling to improve density and resolution, refining the geometric details of the completed structures. To further optimize the network, we incorporate meta-auxiliary learning [Finn *et al.*, 2017] to ensure that minimizing auxiliary loss during inference directly

benefits the primary task. Furthermore, we introduce an auto-weighted loss mechanism to dynamically adjust the contributions of auxiliary losses. This allows the model to balance multiple tasks and mitigate negative transfer [Vafaeikia *et al.*, 2020], which can occur when ineffective information is exchanged during the learning process. Our main contributions can be summarized as follows:

- We leverage meta-learning to achieve test-time adaptation for point cloud completion. TAMECloud enables fast and accurate adaptation to target domains, resulting in significant performance gains.
- We enhance auxiliary tasks for point cloud completion by implementing improved weight sharing among them through a meta-learning manner. This allows TAMECloud to dynamically balance the contributions of auxiliary tasks and update primary network weights during inference, eliminating the need for difficult and costly manual adjustments.
- TAMECloud significantly outperforms existing methods across a wide range of datasets. To the best of our knowledge, this is the first instance where a meta-auxiliary learning method with dynamic weight adjustment is utilized to improve point cloud completion.

2 Related Work

Point Cloud Completion. Recent studies on point cloud completion focus on designing innovative network architectures to improve accuracy and robustness. Notable methods [Yuan *et al.*, 2018; Tchapmi *et al.*, 2019; Yu *et al.*, 2021; Xiang *et al.*, 2021] have made significant advancements and established a strong foundation for future research. However, many of these approaches overlook the challenge of domain shift, leading to limited generalization across different datasets and application scenarios. To address this, we tackle point cloud completion from the perspective of meta-learning and auxiliary learning strategies. This approach is particularly challenging due to the variability in datasets and the noise inherent in real-world environments.

Test-time Adaptation. Test-time adaptation (TTA) is an effective strategy for improving model generalization across diverse data distributions. Recent methods successfully apply TTA with self-supervised auxiliary tasks in domains such as dynamic scene deblurring [Chi *et al.*, 2021; He *et al.*, 2025] and optical flow estimation [Ayyoubzadeh *et al.*, 2023]. In the context of 3D tasks, TTA has been explored for point cloud upsampling, registration, and segmentation [Hatem *et al.*, 2023a,b; Zou *et al.*, 2025]. However, these approaches often struggle to adapt the primary task effectively, as auxiliary tasks can introduce noise or inefficiencies [Vafaeikia *et al.*, 2020; Cui and Sun, 2021]. To address these challenges in point cloud completion, we propose a meta-auxiliary learning approach that employs lightweight auxiliary tasks for efficient fine-tuning, enabling dynamic adaptation to unseen data within a few iterations.

Meta-Auxiliary Learning. Meta-learning methods, such as model-agnostic meta-learning (MAML) [Finn *et al.*, 2017], have gained popularity for their ability to accelerate convergence and avoid local optima [Gui *et al.*, 2018; Chi *et al.*, 2022]. Leveraging these strengths, meta-learning can be combined with auxiliary learning to achieve test-time adaptation, leading to impressive results across various visual tasks. However, existing methods that utilize multiple auxiliary tasks [Liu *et al.*, 2023; Hu *et al.*, 2024] often overlook the automatic relative weighting of task losses, while manual adjustment is both challenging and costly. To address this, our model introduces meta-auxiliary learning, which dynamically adjusts the weighting of loss functions to balance the contributions of each task. When applied to point cloud completion, our approach enables efficient test-time adaptation and achieves significant performance improvements.

3 Method

In this section, we introduce TAMECloud, a novel framwork for point cloud completion that effectively adapts to varying varying patterns of missing data. As illustrated in Fig. 1, the method features a network architecture with a primary branch for the main completion task and two auxiliary branches for self-supervised reconstruction, detailed in Sec. 3.1. TAMECloud combines meta-auxiliary learning with test-time adaptation to dynamically tailor model parameters based on the input. Sec. 3.2 describes the meta-auxiliary training and meta-testing integrated with test-time adaptation. Sec. 3.3 presents an adaptive algorithm that dynamically optimizes the loss weights of the auxiliary branches.

3.1 Network Architecture

Primary Branch. Given a partial and unordered point cloud input $\mathcal{P} \in \mathbb{R}^{M\times 3}$, where M is the number of points, the primary task is to generate a complete point cloud $\mathcal{C} \in \mathbb{R}^{N\times 3}$, where $N > M$. In the primary branch shown in Fig.1 (a), we adopt SnowflakeNet [Xiang *et al.*, 2021] as the baseline completion network. The encoder $\mathcal{M}_E$ adopts architectures from [Qi *et al.*, 2017a] and [Zhao *et al.*, 2021], effectively extracting local and global features to create shape code. Its broad applicability in point cloud tasks ensures strong adaptability and robust support for our auxiliary branch. Additionally, $\mathcal{M}_E$ enables the sharing of learnable weights between primary and auxiliary tasks, denoted $\phi_{shar\text{-}p}$. The decoder $\mathcal{M}_D$ is optimized separately from the auxiliary branch, using a coarse-to-fine deconvolution network to generate the predicted point cloud $\mathcal{C}$.

The objective for the primary task is to minimize the reconstruction error between the predicted point cloud and the ground truth $\mathcal{G}$. We adopt Chamfer Distance (CD) between two point sets $\mathcal{X}$ and $\mathcal{Y}$ is defined as:

$$\mathcal{L}_{CD}(\mathcal{X},\mathcal{Y}) = \frac{1}{n_x}\sum_{x\in\mathcal{X}}\min_{y\in\mathcal{Y}}\|x-y\| + \frac{1}{n_y}\sum_{y\in\mathcal{Y}}\min_{x\in\mathcal{X}}\|y-x\|, \tag{1}$$

where n_x and n_y denote the number of points in $\mathcal{X}$ and $\mathcal{Y}$, respectively. Following Eq. (1), for the primary branch, we compute the loss between the complete point cloud $\mathcal{C}$ and the ground truth $\mathcal{G}$ as $\mathcal{L}_{\text{pri}} = \mathcal{L}_{CD}(\mathcal{C},\mathcal{G})$.

Auxiliary Branch. As shown in Fig. 2, our network is designed with two auxiliary branches for self-supervised tasks:

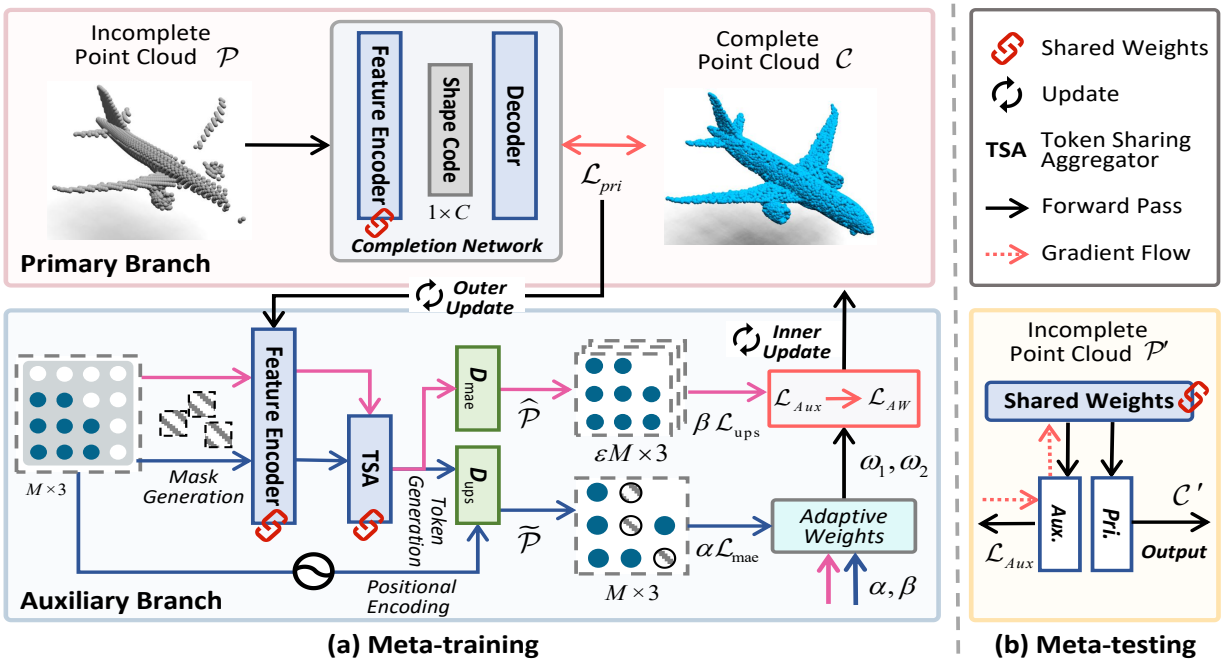

Figure 1: Overview of our test-time auto-weighted fast adaptation method. TAMECloud leverages point cloud completion as the primary task and incorporates reconstruction and upsamping as two self-supervised auxiliary tasks to enhance performance. During the Meta-training phase (a), meta-auxiliary learning integrates the primary and auxiliary tasks, enabling weight sharing and eliminating redundant structures. To handle domain shifts, test-time adaptation is performed in Meta-testing phase (b) by three steps: First, meta-learned weights generate initial point cloud completions. Then, the primary network is updated through self-supervised learning from the auxiliary tasks. Finally, the adapted model achieves more accurate and generalized point cloud completions compared to the pre-trained model in target domains.

Aux_{mae} (masked autoencoder reconstruction) and Aux_{ups} (upsamping), as explained below.

Reconstruction. To align with the primary completion task, we propose a self-supervised point cloud reconstruction branch as our first auxiliary task. Instead of directly exploiting the naive autoencoder models for the reconstruction branch as in [Chi *et al.*, 2021; Hatem *et al.*, 2023a], we propose to take advantage of masked autoencoder models for more effective feature learning. Basically, by adopting a random block masking strategy, our model is no longer confined to the fixed patterns present in the training data during the reconstruction process. Instead, it handles the missing information in various ways, thereby capturing richer features to strengthen the primary task's generalization on unseen data. Inspired by PointGPT [Chen *et al.*, 2024], the reconstruction branch predicts masked point clouds in an autoregressive manner, learning latent representations conditioned on previously reconstructed blocks.

Specifically, as shown in Fig. 2 (a), for the input of incomplete point cloud $\mathcal{P}$, we extract the shape code $\mathcal{V}$ using the shared-weight feature extractor $\mathcal{M}_E$ from the primary task, after which we perform a max-pooling operation along the dimension of the point (dim = 2) to aggregate these features into a compact global representation $\mathcal{F}$. Then, we introduce an additional Token Sharing Aggregator τ that transforms $\mathcal{F}$ into the group token $\mathcal{T}^{\mathcal{G}} \in \mathbb{R}^{N\times D}$, where N is the number of groups and D is the transformation dimension. Note that we will share the learnable weights $\psi_{share\text{-}a}$ in Aux_{ups}. This process can be formulated as:

$$\begin{aligned}\tau(\cdot) &= \text{Reshape}(\text{ReLU}(\text{BN}(\text{MLP}(\cdot)))),\\ \mathcal{T}^{\mathcal{G}} &= \tau(\text{Maxpooling}_{\text{d=2}}(\mathcal{M}_E(\mathcal{P})).\end{aligned} \tag{2}$$

where $\mathcal{T}^{\mathcal{G}}$ relies more on the weight sharing of the modules from the primary task.

Meanwhile, given the input point cloud, we apply Farthest Point Sampling (FPS) to obtain the center coordinates for N group $\mathbf{q}_i^n \in \mathcal{Q} = \{\mathbf{q}_i^1, \mathbf{q}_i^2, \ldots, \mathbf{q}_i^N\}$ and feed them into a learnable positional embedding module $PE(\cdot)$, producing the position embedding $\mathbf{z}_i^n \in \mathcal{Z} = \{\mathbf{z}_i^1, \mathbf{z}_i^2, \ldots, \mathbf{z}_i^N\}$. Next, we adopt the dual-masking strategy [Chen *et al.*, 2024], which is beneficial for reconstruction by enforcing a step-by-step generation process. The mathematical formulation of the masked-attention is:

$$\text{AttnMask}(x) = \text{Softmax}\Big(\frac{QK^T}{\sqrt{D}} - (1-M^d)\cdot\infty\Big)V, \tag{3}$$

where Q and K are the query and key projections, D is the projection dimension, and M^d denotes the binary attention mask. The $\text{AttnMask}(\cdot)$ is applied to token sequence $\mathcal{T}^{\mathcal{G}}$ to enforce a stepwise process, ultimately yielding context-rich features, $\mathcal{F}_M = (\mathcal{F}_{M_1}, \mathcal{F}_{M_2}, \ldots, \mathcal{F}_{M_N})$ for reconstruction.

Finally, instead of directly reshaping these features ($\mathcal{F}_M$) into a point cloud as in [Chen *et al.*, 2024], we first apply a max-pooling operation along the feature dimension before several linear layers, reducing the input dimensionality and

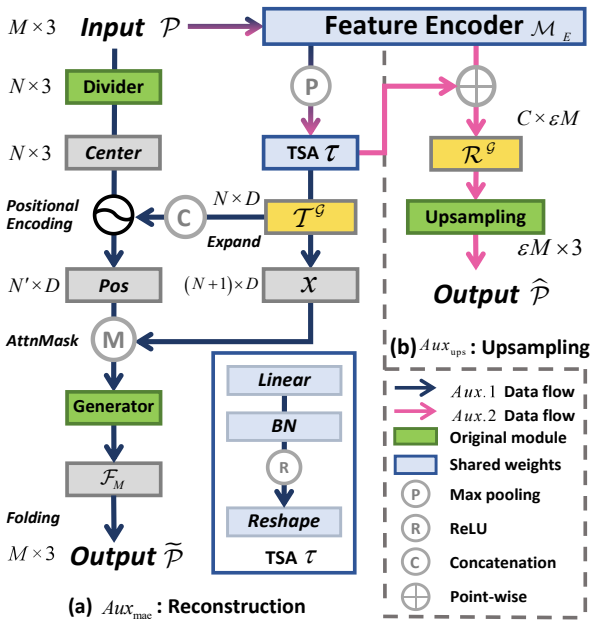

Figure 2: Overall architecture of the auxiliary branch with two self-supervised tasks: (a) The Aux_{mae} is a reconstruction task, utilizing the shared encoder $\mathcal{M}_E$ in $Pri.$ to extract the shape code $\mathcal{V}$. It then employs a dual-masking strategy to reconstruct the complete point $\tilde{\mathcal{P}}$; (b) TheAux_{ups} is an upsampling task, using the shared encoder $\mathcal{M}_E$ alongside the Token Sharing Aggregator τ to generate the upsampled point $\hat{\mathcal{P}}$.

the number of parameters of linear layers. Afterward, we exploit a FoldingNet-based decoder [Yang *et al.*, 2018] to reconstruct the point cloud $\tilde{\mathcal{P}}$. We compute the self-supervised reconstruction loss between the input point cloud $\mathcal{P}$ and the reconstructed point cloud $\tilde{\mathcal{P}}$ using Chamfer Distance and $\mathcal{L}_{\text{mae}} = \mathcal{L}_{CD}(\mathcal{P}, \tilde{\mathcal{P}})$.

Upsampling. In addition to the reconstruction branch, we propose to incorporate another auxiliary branch Aux_{ups} that conducts upsampling task. Specifically, this auxiliary task is to learn a mapping function $\Upsilon_\varepsilon : \mathbb{R}^{M\times 3} \rightarrow \mathbb{R}^{\varepsilon M\times 3}$ (where $\varepsilon = 4$), such that $\hat{\mathcal{P}} = \Upsilon_\varepsilon(\mathcal{P})$. As shown in Fig. 2 (b), first we utilize shared encoders $\mathcal{M}_E$ of the primary branch to extract global features, which is followed by a shared Token Sharing Aggregator τ from Aux_{mae} to gather refined features within specified point cloud neighborhoods. In this way, we can share the group token extraction with Aux_{mae}, reducing redundancy and avoiding re-learning similar functionality. By sharing certain parameters with the primary task ($Pri.$) and Aux_{mae}, the model can more effectively transfer knowledge across tasks, thereby strengthening inter-task relationships and synergistic effects. Next, we use a point-wise operator to combine the group tokens and local features into a refined sequence $\mathcal{R}^{\mathcal{G}} \in \mathbb{R}^{C\times\varepsilon M}$. Finally, we retain the SpatialRefiner module from Dis-PU [Li *et al.*, 2021] to generate the upsampled point cloud $\hat{\mathcal{P}}$. Similarly, we determine loss by comparing the ground truth $\mathcal{G}$ with the output $\hat{\mathcal{P}}$ as follows: $\mathcal{L}_{\text{ups}} = \mathcal{L}_{CD}(\hat{\mathcal{P}}, \mathcal{G})$.

3.2 Meta-Auxiliary Learning

Joint Training. We first jointly train the primary and auxiliary tasks on the source domain, with the loss function formulated as a weighted sum of the primary task loss and the auxiliary loss:

$$\begin{aligned}\mathcal{L} = &\mathcal{L}_{\text{pri}}(\mathcal{C}, \mathcal{P}; \phi_{shar\text{-}p}, \phi_{pri}) \\ &+ \mu \Big[\mathcal{L}_{\text{mae}}(\tilde{\mathcal{P}}, \mathcal{P}; \phi_{shar\text{-}p}, \phi_{shar\text{-}a}, \phi_{Aux_{\text{mae}}}) \\ &+\mathcal{L}_{\text{ups}}(\hat{\mathcal{P}}, \mathcal{P}; \phi_{shar\text{-}p}, \phi_{shar\text{-}a}, \phi_{Aux_{\text{ups}}})\Big],\end{aligned} \tag{4}$$

where the hyper-parameter $\mu \in (0, 1]$ is the weight coefficient used to balance the primary and auxiliary tasks.

Meta-training. We then meta-learn from this pre-trained model, refining its parameters to optimize performance on the primary task. This two-stage strategy alleviates the difficulty of meta-learning from scratch and leverages the auxiliary tasks to boost final completion quality.

Specifically, we decompose the model parameters into shared weights $\{\phi_{shar\text{-}p}, \phi_{shar\text{-}a}\}$ between primary and auxiliary branches, and task-specific weights $\{\phi_{pri}, \phi_{Aux_{\text{mae}}}, \phi_{Aux_{\text{ups}}}\}$ for each branch. For each auxiliary task, we perform an inner-loop optimization to minimize the auxiliary loss:

$$\begin{aligned}\phi_{Aux_{\text{mae}}} &\leftarrow \phi - \alpha\nabla_\phi \mathcal{L}_{\text{mae}}(\tilde{\mathcal{P}}, \mathcal{P}; \phi), \\ \phi_{Aux_{\text{ups}}} &\leftarrow \phi - \beta\nabla_\phi \mathcal{L}_{\text{ups}}(\hat{\mathcal{P}}, \mathcal{P}; \phi),\end{aligned} \tag{5}$$

where $\tilde{\mathcal{P}}$ and $\hat{\mathcal{P}}$ represent the predicted outputs for each task, and α and β are the learning rates for the auxiliary tasks. These updates ensure that the auxiliary tasks are aligned with the objectives of the primary task.

After updating the auxiliary task parameters, we optimize the primary task loss:

$$\mathcal{L}_{\text{pri}} = \frac{1}{T}\sum_{i=1}^{T} \mathcal{L}_{pri}(\mathcal{C}, \mathcal{P}; \phi_{pri}), \tag{6}$$

where T is the batch size, $\mathcal{L}_{pri}$ is the primary task loss function, and ϕ_{pri} is the parameter set specific to the primary task. The optimization is performed using gradient descent:

$$\phi \leftarrow \phi - \gamma\nabla_\phi \mathcal{L}_{\text{pri}}. \tag{7}$$

where γ is the learning rate for the primary task.

Meta-testing. As shown in Fig. 1(b), at inference stage we perform test-time adaptation by applying small gradient updates to model parameters using auxiliary losses. This improves performance on unseen samples and ensures TAME-Cloud generalizes effectively to new instances.

3.3 Adaptive Multi-Task Loss

Through extensive experiments, we observed that selecting the appropriate weights for the losses in multi-task settings is a non-trivial problem. Careless selection of these hyper-parameters can lead to accumulation of errors in the primary task over time. However, existing works [Hatem *et al.*, 2023b; Chi *et al.*, 2022; Liu *et al.*, 2023] often overlook this issue. Inspired by previous studies [Sener and Koltun, 2018; Liebel

Algorithm 1: Meta-training with an Adaptive Multi-task Loss

Input: parameter
$\phi = \{\phi_{shar\text{-}p}, \phi_{pri}, \phi_{shar\text{-}a}, \phi_{Aux_{\text{mae}}}, \phi_{Aux_{\text{ups}}}\}$
γ: primary model learning rate
Output: ϕ: meta-auxiliary learned parameters
Initialize model parameters ϕ_{Aux} and loss weights α, β Set model learning rate η_ϕ and weight learning rate η_w
while *not done* **do**
  sample a training batch from the $\{\mathcal{C}^{(t)}, \mathcal{P}^{(t)}\}_{t=1}^{T}$;
  foreach *t* **do**
    evaluate the auxiliary losses $\mathcal{L}_{\text{mae}}, \mathcal{L}_{\text{ups}}$;
    normalize weights and compute weighted loss:
    $\alpha \leftarrow \log(1+\alpha^2), \quad \beta \leftarrow \log(1+\beta^2)$
    $w_1 \leftarrow \frac{\exp(\alpha)}{\exp(\alpha)+\exp(\beta)}, \quad w_2 \leftarrow \frac{\exp(\beta)}{\exp(\alpha)+\exp(\beta)}$
    $\mathcal{L}_{AW} \leftarrow w_1 \mathcal{L}_{\text{mae}} + w_2 \mathcal{L}_{\text{ups}}$
    update auxiliary branch parameters:
    $\nabla_{\phi_{Aux}} \mathcal{L}_{AW} \leftarrow \text{backprop}(\mathcal{L}_{AW}, \phi_{Aux})$
    $\phi_{Aux} \leftarrow \phi_{Aux} - \eta_\phi \nabla_{\phi_{Aux}} \mathcal{L}_{AW}$
    update loss weights:
    $w_1 \leftarrow w_1 - \eta_w \nabla_{w_1} \mathcal{L}_W$
    $w_2 \leftarrow w_2 - \eta_w \nabla_{w_2} \mathcal{L}_W$
  evaluate the primary task and update:
  $\phi \leftarrow \phi - \gamma \sum_{t=1}^{T} \nabla_\phi \mathcal{L}_{\text{pri}}(\mathcal{C}, \mathcal{P}; \phi_{shar\text{-}p}, \phi_{pri})$

and Körner, 2018; Kendall *et al.*, 2018], we propose an automatic weighting mechanism to adapt the loss weights for the auxiliary branches integrated with our meta-training process. The overall procedure of meta-training is detailed in Alg. 1.

Specifically, in each training iteration, we first compute the normalized weights w_1 and w_2 using the initial parameters α and β, constructing the weighted loss $\mathcal{L}_{AW}$. Then, using gradient descent with learning rate η_w, we adjust w_1 and w_2 to minimize this weighted loss, and subsequently refine the primary network parameters using the loss from the primary task. The loss weights are dynamically adjusted during this process, allowing for an adaptive balance between the contributions of each task.

4 Experiments

In this section, we evaluate our method on both synthetic (ShapeNet [Chang *et al.*, 2015], PCN [Yuan *et al.*, 2018]) and real-world (KITTI [Geiger *et al.*, 2013]) datasets. ShapeNet and PCN provide dense, uniformly sampled synthetic point clouds with paired ground truth for training and evaluation. KITTI offers real LiDAR scans with sparse, unevenly distributed points without paired ground truth. Therefore, it is used only for evaluation purposes.

4.1 Results on PCN Dataset

Dataset Briefs and Evaluation Metrics. PCN dataset [Yuan *et al.*, 2018] includes eight categories with a total of 30,974 CAD models. We compare with [Yang *et al.*, 2018; Groueix *et al.*, 2018; Yuan *et al.*, 2018; Tchapmi *et al.*, 2019; Xie *et al.*, 2020; Wen *et al.*, 2021; Xiang *et al.*, 2021; Yu *et al.*, 2021; Wen *et al.*, 2022; Wang *et al.*, 2024] based on their open-source code and use the optimal hyperparameters specified in their papers for a fair comparison. To assess the performance of each method, we follow the previous setting on this dataset and evaluate the results using the ℓ_1-norm variant of the Chamfer distance.

CD-ℓ_1($\times$1000)	Plane	Cabinet	Car	Chair	Lamp	Couch	Table	Boat	Avg
FoldingNet [Yang *et al.*, 2018]	9.49	15.80	12.61	15.55	16.41	15.97	13.65	14.99	14.31
AtlasNet [Groueix *et al.*, 2018]	6.37	11.94	10.10	12.06	12.37	12.99	10.33	10.61	10.85
PCN [Yuan *et al.*, 2018]	5.50	22.70	10.63	8.70	11.00	11.34	11.68	8.59	9.64
TopNet [Tchapmi *et al.*, 2019]	7.61	13.31	10.90	13.82	14.44	14.78	11.22	11.12	12.15
GRNet [Xie *et al.*, 2020]	6.45	10.37	9.45	9.41	7.96	10.51	8.44	8.04	8.83
CRN [Wang *et al.*, 2020]	4.79	9.97	8.31	9.49	8.94	10.69	7.81	8.05	8.51
NSFA [Zhang *et al.*, 2020b]	4.76	10.18	8.63	8.53	7.03	10.53	7.35	7.48	8.06
PMP-Net [Wen *et al.*, 2021]	5.65	11.24	9.64	9.51	6.95	10.83	8.72	7.25	8.73
SnowflakeNet [Xiang *et al.*, 2021]	4.29	9.16	8.08	7.89	6.07	9.23	6.55	6.40	7.21
PoinTr [Yu *et al.*, 2021]	4.75	10.47	8.68	9.39	7.75	10.93	7.78	7.29	8.38
PMP-Net++ [Wen *et al.*, 2022]	4.39	9.96	8.53	8.09	6.06	9.82	7.17	6.52	7.56
PointAttN [Wang *et al.*, 2024]	3.87	9.00	7.63	7.43	**5.90**	8.68	6.32	**6.09**	6.86
Ous	**3.69**	**8.82**	**7.44**	**7.27**	6.07	**8.59**	**6.17**	6.49	**6.81**

Table 1: Quantitative comparison on the PCN dataset (per-point CD-$\ell_1 \times 1000$). Both the output and ground truth point clouds consist of 16,384 points. (Lower CD is better)

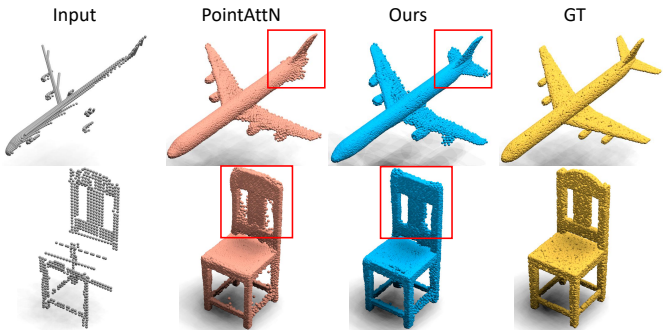

Figure 3: Visualization results on the PCN dataset, including input, GT, and predictions of PointAttN and our model. It can be see that our proposed method can generate more complete and compact surfaces with detailed structures.

Quantitative and Visual Comparison. In Table 1, TAMECloud achieves SOTA performance across most categories. Furthermore, as shown in Fig. 3, TAMECloud outperforms PointAttN [Wang *et al.*, 2024] in recovering fine-grained geometry and structure, particularly around the challenging areas such as airplane's tail (first row) and the backrest of the chair (second row). These results highlight TAMECloud's ability to excel in diverse and complex scenarios.

4.2 Results on ShapeNet-55 Dataset

Dataset Briefs and Evaluation Metrics. We also conduct experiments across all 55 object categories in ShapeNet [Chang *et al.*, 2015] for comprehensive and diverse evaluation. The dataset is split by 8:2, with 80% of objects for training (41,952 models) and 20% for testing (10,518 models). Following previous settings on this dataset, we adopt the ℓ_2-norm Chamfer distance for evaluation and also report results using the F-Score@1% metric [Tatarchenko *et al.*, 2019]. For a fair comparison, we train and test several prior methods [Yang *et al.*, 2018; Yuan *et al.*, 2018; Tchapmi *et al.*, 2019; Zhang *et al.*, 2020a; Xie *et al.*, 2020; Yu *et al.*, 2021; Xiang *et al.*, 2021; Li *et al.*, 2023] on this dataset.

Quantitative and Visual Comparison. As shown in Table 2,

CD-ℓ_2($\times$1000)	Table	Chair	Plane	Car	Sofa	Bird House	Bag	Remote	Key board	Rocket	CD-S	CD-M	CD-H	CD-Avg	F1
FoldingNet [Yang *et al.*, 2018]	2.53	2.81	1.43	1.98	2.48	4.71	2.79	1.44	1.24	1.48	2.67	2.66	4.05	3.12	0.082
PCN [Yuan *et al.*, 2018]	2.13	2.29	1.02	1.85	2.06	4.50	2.86	1.33	0.89	1.32	1.94	1.96	4.08	2.66	0.133
TopNet [Tchapmi *et al.*, 2019]	2.21	2.53	1.14	2.18	2.36	4.83	2.93	1.49	0.95	1.32	2.26	2.16	4.30	2.91	0.126
PFNet [Zhang *et al.*, 2020a]	3.95	4.24	1.81	2.53	3.34	6.21	4.96	2.91	1.29	2.36	3.83	3.87	7.97	5.22	0.339
GRNet [Xie *et al.*, 2020]	1.63	1.88	1.02	1.64	1.72	2.97	2.06	1.09	0.89	1.03	1.35	1.71	2.85	1.97	0.238
PoinTr [Yu *et al.*, 2021]	0.81	0.95	0.44	0.91	0.79	1.86	0.93	0.53	0.38	0.57	0.58	0.88	1.79	1.09	0.464
SnowflakeNet [Xiang *et al.*, 2021]	0.75	0.84	0.42	0.88	0.72	1.74	0.81	0.48	0.36	0.51	0.52	0.80	1.62	0.98	0.477
ProxyFormer [Li *et al.*, 2023]	0.70	0.83	**0.34**	0.78	0.69	-	-	-	-	-	0.49	0.75	1.55	0.93	0.483
Ous	**0.65**	**0.78**	**0.34**	**0.71**	**0.62**	**1.38**	**0.70**	**0.39**	**0.29**	**0.45**	**0.44**	**0.69**	**1.43**	**0.85**	**0.516**

Table 2: Quantitative comparison on ShapeNet-55. We provide detailed results for each method across 10 categories and present overall results for all 55 categories under three difficulty levels. The CD-ℓ_2 results for small, medium, and hard cases are denoted as CD-S, CD-M, and CD-H, respectively. F1 is the average result across all categories. (Lower CD and higher F1 are better)

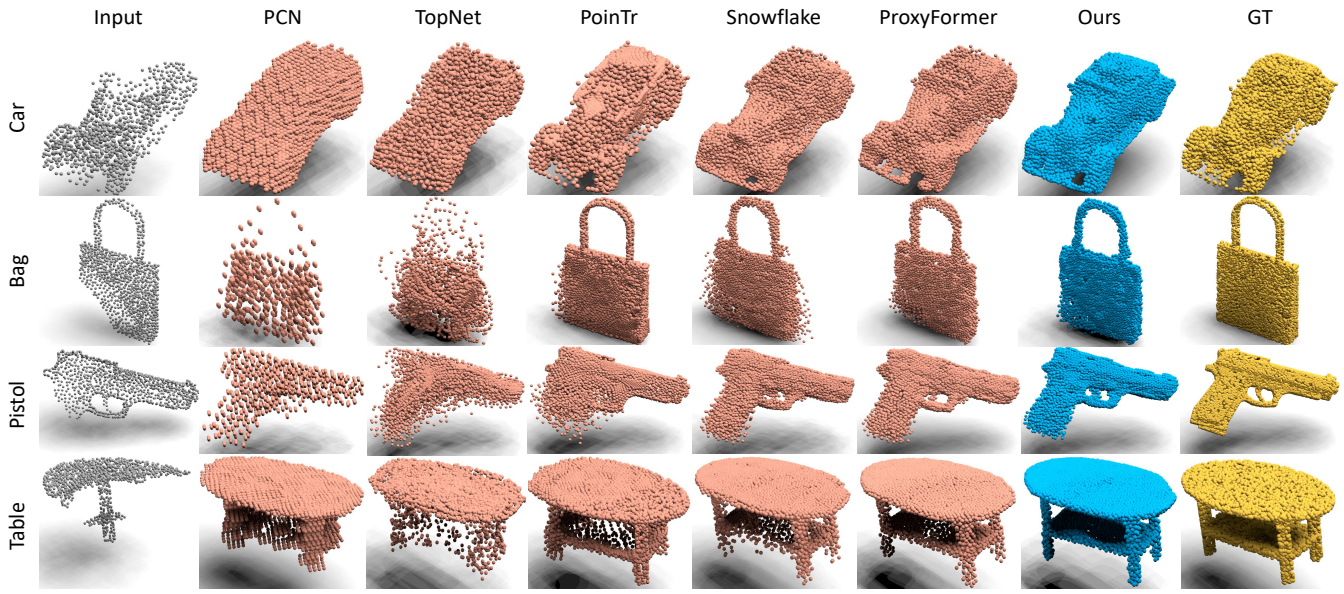

Figure 4: Visualization results on ShapeNet-55 dataset. Our methods can produce complete point clouds with higher fidelity, which is less noisy and can preserve fine details such as the trigger of the Pistol on the third row.

TAMECloud achieves SOTA performance in nearly all categories, indicating its effectiveness in diverse scenarios. Fig. 4 visualizes the completion results. Notably, whether only a few points are missing (the first two rows) or the point cloud is extensively incomplete (the last two rows), TAMECloud consistently outperforms previous methods, underscoring its robustness in handling varying degrees of missing data. In the Car category, our method preserves complex structures and details, while others show blurriness and missing elements. Similar observations hold for the Bag and Pistol categories. Even with highly incomplete inputs in the Table category, TAMECloud generates shapes closely resembling the ground truth, whereas other methods yield noticeably noisy results.

4.3 Results on KITTI Dataset

Dataset Briefs and Evaluation Metrics. We evaluate our method on the KITTI the dataset [Geiger *et al.*, 2013], which contains incomplete point clouds of cars from LiDAR scans in real-world environments with no ground truth. Since there is no paired ground truth, Fidelity and Minimal Matching Distance (MMD) are used as evaluation metrics. We compare with [Groueix *et al.*, 2018; Yuan *et al.*, 2018; Yang *et al.*, 2018; Tchapmi *et al.*, 2019; Liu *et al.*, 2020; Zhang *et al.*, 2020b; Wang *et al.*, 2020; Xie *et al.*, 2020; Yu *et al.*, 2021; Zhou *et al.*, 2022; Li *et al.*, 2023].

Quantitative and Visual Comparison. Table 3 shows the quantitative experiments, while Fig. 5 presents the visualization results compared to PoinTr and ProxyFormer. Even when dealing with highly sparse LiDAR scan point clouds, our approach reconstructs the shape of cars with higher quality at a fine-grained level. This indicates that our proposed meta-auxiliary training and test-time adaptation framework possesses more substantial generalization capabilities for handling unseen data.

4.4 Ablation Studies

Impact of Auxiliary Branches. We conduct the ablation experiments across five categories in ShapeNet-55 dataset. As shown in Table 4, each auxiliary branch individually provides a modest performance gain, while using both yields a significant boost, highlighting the combined benefits of these tasks.

Impact of Token Sharing Aggregator and Auto-weighted

	AtlasNet	PCN	FoldingNet	TopNet	MSN	NSFA	CRN	GRNet	PoinTr	SeedFormer	ProxyFormer	Ours
Fidelity ↓	1.759	2.235	7.467	5.354	0.434	1.281	1.023	0.816	**0.000**	0.151	**0.000**	0.124
MMD ↓	2.108	1.366	0.537	0.636	2.259	0.891	0.872	0.568	0.526	0.516	0.508	**0.470**

Table 3: Quantitative comparison on the KITTI dataset. we use the Fidelity Distance and Minimal Matching Distance (MMD) for evaluation metrics. (Lower Fidelity and MMD are better)

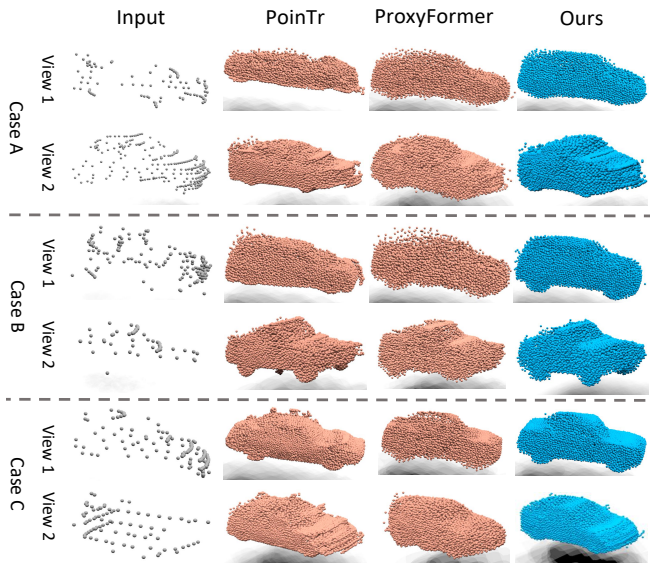

Figure 5: Visual comparison on the KITTI dataset.

Aux_{mae}	Aux_{ups}	Table	Chair	Plane	Car	Sofa	CD-Avg (ℓ_2)
✗	✗	0.747	0.844	0.418	0.883	0.722	0.723
✓	✗	0.675	0.797	0.359	0.736	0.655	0.644
✗	✓	0.709	0.824	0.394	0.813	0.676	0.683
✓	✓	**0.654**	**0.783**	**0.344**	**0.709**	**0.623**	**0.623**

Table 4: Ablation studies on Aux_{mae} (reconstruction) and Aux_{ups} (upsampling) across five categories in ShapeNet-55 dataset. (Lower CD is better)

Loss. Table 5 demonstrates the effectiveness of two modules: (A) Token Sharing Aggregator (TSA) and (B) Auto-weighted Loss (AW). We conduct the evaluation on both PCN and ShapeNet-55 datasets. The results show that sharing weights among auxiliary tasks enhances their strong correlation, facilitating the transfer of useful information. Additionally, the Auto-weighted Loss (AW) module further improves performance and eliminates the need for manual parameter tuning.

Impact of Framework Components. As shown in Table 6, we first investigate the impact of the Auxiliary Learning ($Aux.$) component. This component yields a modest improvement for the primary task. Exploiting the model trained under the multi-task setting with both primary and auxiliary branches, we assess Test-Time Adaptation (TTA) and observe that it does not significantly improve performance. This suggests that TTA alone may not be highly effective without the support of additional components. Next, we examine the model after the meta-auxiliary training, which produces a noticeable enhancement. Finally, given the model after the meta-auxiliary training, our complete framework further improves point cloud completion capability by fine-tuning the model at test time. To better illustrate the effectiveness of the components, we provide a visualization as shown in Fig. 6.

Model			**PCN**	**ShapeNet**	
			CD-Avg (ℓ_1)	CD-Avg (ℓ_2)	F1
(A)	Token Sharing Aggrator	w/o TSA	7.15	0.98	0.479
(B)	Auto-weight Loss	w/o AW	6.84	0.92	0.486
(C)	Full Model	w/ (TSA + AW)	**6.81**	**0.85**	**0.516**

Table 5: Effects of the proposed Token Sharing Aggregator (TSA) and Auto-weighted Loss (AW) evaluated on the PCN and ShapeNet datasets. (Lower CD and F1 are better)

Model			**ShapeNet**	
$Aux.$	TTA	Meta	CD-Avg (ℓ_2)	F1
			0.98	0.477
✓			0.92	0.490
✓	✓		0.97	0.477
✓		✓	0.89	0.505
✓	✓	✓	**0.85**	**0.516**

Table 6: Ablation studies on framework components evaluated on the ShapeNet: Auxiliary Learning ($Aux.$), Meta-Learning (Meta), and Test-time Adaptation (TTA). (Lower CD and F1 are better)

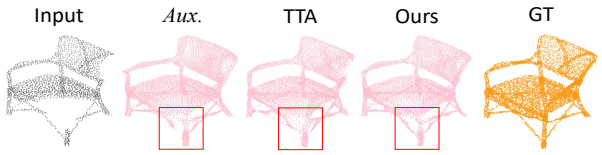

Figure 6: Visualization of the ablation study on different components of our framework.

5 Conclusion

In this paper, we introduce TAMECloud, a novel framework for point cloud completion that integrates test-time adaptation with meta-auxiliary learning. By sharing weights across tasks and dynamically balancing losses, TAMECloud streamlines the weight updates and effectively adapts to diverse and unseen scenarios. Experimental results on multiple datasets demonstrate that TAMECloud achieves SOTA performance, offering a robust solution for 3D point cloud completion.

Limitation and Future Work. The incorporation of meta-learning and auxiliary branches increases the overall complexity of the network. Training such a network would require more computational resources and time compared to training the base architecture with just one primary branch. How to reducing the training cost for our framework will be worth to investigate in the future. Furthermore, it will also be interesting to explore whether integrating more auxiliary tasks can further improve the performance.

References

Seyed Mehdi Ayyoubzadeh, Wentao Liu, Irina Kezele, Yuanhao Yu, Xiaolin Wu, Yang Wang, and Tang Jin. Test-time adaptation for optical flow estimation using motion vectors. *IEEE Transactions on Image Processing*, 2023.

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

Guangyan Chen, Meiling Wang, Yi Yang, Kai Yu, Li Yuan, and Yufeng Yue. Pointgpt: Auto-regressively generative pre-training from point clouds. *Advances in Neural Information Processing Systems*, 36, 2024.

Zhixiang Chi, Yang Wang, Yuanhao Yu, and Jin Tang. Test-time fast adaptation for dynamic scene deblurring via meta-auxiliary learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9137–9146, June 2021.

Zhixiang Chi, Li Gu, Huan Liu, Yang Wang, Yuanhao Yu, and Jin Tang. Metafscil: A meta-learning approach for few-shot class incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14166–14175, 2022.

Qiongjie Cui and Huaijiang Sun. Towards accurate 3d human motion prediction from incomplete observations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4801–4810, 2021.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

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

Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan Russell, and Mathieu Aubry. AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Liang-Yan Gui, Yu-Xiong Wang, Deva Ramanan, and José MF Moura. Few-shot human motion prediction via meta-learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 432–450, 2018.

Ahmed Hatem, Yiming Qian, and Yang Wang. Point-tta: Test-time adaptation for point cloud registration using multitask meta-auxiliary learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16494–16504, October 2023.

Ahmed Hatem, Yiming Qian, and Yang Wang. Test-time adaptation for point cloud upsampling using meta-learning. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1284–1291. IEEE, 2023.

Jin-Ting He, Fu-Jen Tsai, Jia-Hao Wu, Yan-Tsung Peng, Chung-Chi Tsai, Chia-Wen Lin, and Yen-Yu Lin. Domain-adaptive video deblurring via test-time blurring. In *European Conference on Computer Vision*, pages 125–142. Springer, 2025.

Shengxiang Hu, Huaijiang Sun, Bin Li, Dong Wei, Weiqing Li, and Jianfeng Lu. Fast adaptation for human pose estimation via meta-optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1792–1801, 2024.

Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7482–7491, 2018.

Ruihui Li, Xianzhi Li, Pheng-Ann Heng, and Chi-Wing Fu. Point cloud upsampling via disentangled refinement. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Shanshan Li, Pan Gao, Xiaoyang Tan, and Mingqiang Wei. Proxyformer: Proxy alignment assisted point cloud completion with missing part sensitive transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9466–9475, 2023.

Lukas Liebel and Marco Körner. Auxiliary tasks in multi-task learning. *arXiv preprint arXiv:1805.06334*, 2018.

Minghua Liu, Lu Sheng, Sheng Yang, Jing Shao, and Shi-Min Hu. Morphing and sampling network for dense point cloud completion. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 11596–11603, 2020.

Huan Liu, Zhixiang Chi, Yuanhao Yu, Yang Wang, Jun Chen, and Jin Tang. Meta-auxiliary learning for future depth prediction in videos. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 5756–5765, 2023.

Yu Liu, Shuting Wang, Yuanlong Xie, Tifan Xiong, and Mingyuan Wu. A review of sensing technologies for indoor autonomous mobile robots. *Sensors*, 24(4):1222, 2024.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 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, 2017.

Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. *Advances in neural information processing systems*, 31, 2018.

Maxim Tatarchenko, Stephan R Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What do

single-view 3d reconstruction networks learn? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3405–3414, 2019.

Lyne P Tchapmi, Vineet Kosaraju, Hamid Rezatofighi, Ian Reid, and Silvio Savarese. Topnet: Structural point cloud decoder. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 383–392, 2019.

Partoo Vafaeikia, Khashayar Namdar, and Farzad Khalvati. A brief review of deep multi-task learning and auxiliary task learning. *arXiv preprint arXiv:2007.01126*, 2020.

Xiaogang Wang, Marcelo H Ang Jr, and Gim Hee Lee. Cascaded refinement network for point cloud completion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 790–799, 2020.

Zeyu Wang, Cuong Nguyen, Paul Asente, and Julie Dorsey. Pointshopar: Supporting environmental design prototyping using point cloud in augmented reality. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–15, 2023.

Jun Wang, Ying Cui, Dongyan Guo, Junxia Li, Qingshan Liu, and Chunhua Shen. Pointattn: You only need attention for point cloud completion. In *Proceedings of the AAAI Conference on artificial intelligence*, volume 38, pages 5472–5480, 2024.

Xin Wen, Peng Xiang, Zhizhong Han, Yan-Pei Cao, Pengfei Wan, Wen Zheng, and Yu-Shen Liu. Pmp-net: Point cloud completion by learning multi-step point moving paths. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7443–7452, 2021.

Xin Wen, Peng Xiang, Zhizhong Han, Yan-Pei Cao, Pengfei Wan, Wen Zheng, and Yu-Shen Liu. Pmp-net++: Point cloud completion by transformer-enhanced multi-step point moving paths. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):852–867, 2022.

Peng Xiang, Xin Wen, Yu-Shen Liu, Yan-Pei Cao, Pengfei Wan, Wen Zheng, and Zhizhong Han. 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, 2021.

Haozhe Xie, Hongxun Yao, Shangchen Zhou, Jiageng Mao, Shengping Zhang, and Wenxiu Sun. Grnet: Gridding residual network for dense point cloud completion. In *European conference on computer vision*, pages 365–381. Springer, 2020.

Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 206–215, 2018.

Zetong Yang, Li Chen, Yanan Sun, and Hongyang Li. Visual point cloud forecasting enables scalable autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14673–14684, 2024.

Xumin Yu, Yongming Rao, Ziyi Wang, Zuyan Liu, Jiwen Lu, and Jie Zhou. Pointr: Diverse point cloud completion with geometry-aware transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12498–12507, 2021.

Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In *2018 international conference on 3D vision (3DV)*, pages 728–737. IEEE, 2018.

Junchao Zhang, Jianbo Shao, Jianlai Chen, Degui Yang, Buge Liang, and Rongguang Liang. Pfnet: an unsupervised deep network for polarization image fusion. *Optics letters*, 45(6):1507–1510, 2020.

Wenxiao Zhang, Qingan Yan, and Chunxia Xiao. Detail preserved point cloud completion via separated feature aggregation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXV 16*, pages 512–528. Springer, 2020.

Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16259–16268, 2021.

Haoran Zhou, Yun Cao, Wenqing Chu, Junwei Zhu, Tong Lu, Ying Tai, and Chengjie Wang. Seedformer: Patch seeds based point cloud completion with upsample transformer. In *European conference on computer vision*, pages 416–432. Springer, 2022.

Tianpei Zou, Sanqing Qu, Zhijun Li, Alois Knoll, Lianghua He, Guang Chen, and Changjun Jiang. Hgl: Hierarchical geometry learning for test-time adaptation in 3d point cloud segmentation. In *European Conference on Computer Vision*, pages 19–36. Springer, 2025.

A Implementation Details

The model is trained for 300 epochs with a batch size of 32. For the joint training stage, the learning rates α and β are set to 2.5×10^{-5}. Throughout both meta-training and meta-testing, we perform ten gradient updates to fine-tune the model parameters using the auxiliary loss. All experiments are performed on two NVIDIA V100 GPU.