Rectified Point Flow: Generic Point Cloud Pose Estimation

Tao Sun[*] Stanford University

Liyuan Zhu[*] Stanford University

Shengyu Huang NVIDIA Research

Shuran Song Stanford University

Iro Armeni Stanford University

Abstract

We present *Rectified Point Flow*, a unified parameterization that formulates pairwise point cloud registration and multi-part shape assembly as a single conditional generative problem. Given unposed point clouds, our method learns a continuous point-wise velocity field that transports noisy points toward their target positions, from which part poses are recovered. In contrast to prior work that regresses part-wise poses with ad-hoc symmetry handling, our method intrinsically learns assembly symmetries without symmetry labels. Together with a self-supervised encoder focused on overlapping points, Rectified Point Flow achieves a new state-of-the-art performance on six benchmarks spanning pairwise registration and shape assembly. Notably, our unified formulation enables effective joint training on diverse datasets, facilitating the learning of shared geometric priors and consequently boosting accuracy. Our code and models are available at *https://rectified-pointflow.github.io/*.

1 Introduction

Estimating the relative poses of rigid parts from 3D point clouds for alignment is a core task in computer vision and robotics, with applications spanning pairwise registration [1] and complex multi-part shape assembly [2]. In many settings, the input consists of an unordered set of part-level point clouds–without known correspondences, categories, or semantic labels–and the goal is to infer a globally consistent configuration of poses, essentially solving a multi-part (two or more) point cloud pose estimation problem. While conceptually simple, this problem is technically challenging due to the combinatorial space of valid assemblies and the prevalence of symmetry and part interchangeability in real-world shapes [3, 4, 5].

Despite sharing the goal of recovering 6-DoF transformations, different 3D reasoning tasks—such as object pose estimation, part registration, and shape assembly—have historically evolved in silos, treating each part independently and relying on task-specific assumptions and architectures. For instance, object pose estimators often assume known categories or textured markers [6, 7], while part assembly algorithms may require access to a canonical target shape or manual part correspondences [8]. This fragmentation has yielded solutions that perform well in narrow domains but fail to generalize across tasks, object categories, or real-world ambiguities.

Among these tasks, multi-part shape assembly presents especially unique challenges. The problem is inherently under constrained: parts are often symmetric [9], interchangeable [10], or geometrically ambiguous, leading to multiple plausible local configurations. As a result, conventional part-wise registration can produce flipped or misaligned configurations that are locally valid but globally inconsistent with the intended assembly. Overcoming such ambiguities requires a model that can

[*]Equal contribution.

Preprint. Under review.

reason jointly about part identity, relative placement, and overall shape coherence—without relying on strong supervision or hand-engineered heuristics.

In this work, we revisit 3D pose regression and propose a *conditional generative model* for generic point cloud pose estimation that casts the problem as learning a continuous point-wise flow over the input geometry, effectively capturing priors over assembled shapes. More specifically, our method, Rectified Point Flow, models the motion of points from random Gaussian noise in Euclidean space toward the point clouds of assembled objects, conditioned on the unposed part point clouds. This learned flow implicitly encodes part-level transformations, enabling both discriminative pose estimation and generative shape assembly within a single framework.

To further instill geometric awareness of inter-part relationships, we pretrain the encoder of the conditional point clouds on large-scale 3D shape datasets using a self-supervised task: predicting point-wise overlap across parts, formulated as a binary classification task. While GARF [11] also highlights the value of encoder pretraining for a flow model, it relies on mesh-based physical simulation [12] to generate fracture-based supervision signals. In contrast, we introduce a lightweight and scalable alternative that constructs pretraining data by computing geometric overlap between parts. Our data generation is agnostic to data sources tailored for different tasks—including part segmentation [13, 14, 15], shape assembly [12, 16, 17], and registration [18, 19]—without requiring watertight mesh or simulation, an important step towards scalable pretraining for pose estimation.

Our flow-based pose estimation departs from traditional pose-vector regression in three ways: (i) **Geometric grounding**: Rather than regressing poses directly in $\mathrm{SE}(3)$, we operate in Euclidean space over dense point clouds. This makes the model inherently robust to symmetries, part interchangeability, and spatial ambiguities that often challenge conventional methods; (ii) **Joint shape-pose reasoning**: By training to predict the final assembled point cloud, our model learns to reconstruct the shape of the object; and (iii) **Scalable shape prior**: We cast registration and assembly task as "reconstruct the completed assembly." This unified objective lets a single network learn from heterogeneous datasets, yielding scalable training and transferable geometric knowledge across standard pairwise registration, fracture reassembly, and complex furniture assembly tasks.

Our main contributions are summarized as follows:

- We propose Rectified Point Flow, a conditional generative model for generic point cloud pose estimation that addresses both pairwise registration and multi-part assembly tasks and achieves state-of-the-art performances on all the tasks.
- We propose a generalizable pretraining strategy for learning geometric awareness over inter-part relationships across several 3D shape datasets, and formulate it as point-wise overlap prediction across parts.
- We show that our parameterization supports joint training across different registration tasks, boosting the performance on each individual task.

2 Related Work

Parametrization for Pose Estimation. Euler angles and quaternions are the predominant parametrization of rotation in various pose regression tasks [20, 21, 22, 23, 24, 25, 11, 26] due to their simplicity and usability. As Euler angles and quaternions are discontinuous representations, Zhou *et al*. [27] proposed to represent 3D rotation with a continuous representation for neural networks using 5D and 6D vectors. In contrast to directly regressing pose vectors, other methods train networks to find sparse correspondences between image pairs or point cloud pairs and extract pose vectors using Singular Value Decomposition (SVD) [28, 29, 8, 30, 31, 32]. More recently, RayDiffusion [33] proposed to represent camera poses as ray bundles, naturally suited for coupling image features and transformer architectures. DUSt3R [34] directly regresses the pointmap of each camera in a global reference frame and then extracts the camera pose using RANSAC-PnP [35, 36]. Our proposed rectified point flow, extends the pointmap representation for learning generalizable pose estimation on point cloud registration and assembly.

Learning-based 3D Registration. 3D registration aims to align point cloud pairs in the same reference frame by solving the relative transformation from source to target. The first line of work focuses on correspondence-based methods [37, 1, 38, 39] that first extract correspondences between

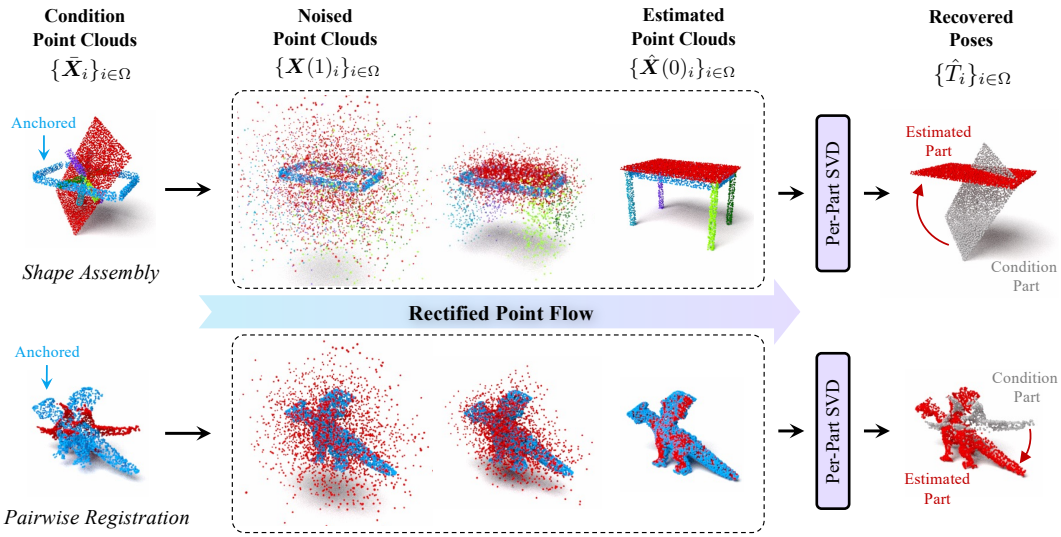

Figure 1: **Pose Estimation via Rectified Point Flow**. Our formulation supports *shape assembly* and *pairwise registration* tasks in a single framework. Given a set of unposed part point clouds $\{\bar{\boldsymbol{X}}_i\}_{i\in\Omega}$, Rectified Point Flow predicts each part's point cloud at the target assembled state $\{\hat{\boldsymbol{X}}_i(0)\}_{i\in\Omega}$. Subsequently, we solve Procrustes problem via SVD between the condition point cloud $\bar{\boldsymbol{X}}_i$ and the estimated point cloud $\hat{\boldsymbol{X}}_i(0)$ to recover the rigid transformation $\hat{T}_i$ for each non-anchored part.

point clouds, followed by robust estimators to recover the transformation. Subsequent works [29, 40, 41, 32] advance the performance by learning more powerful features with improved architecture and loss design. The second line of work comprises direct registration methods [31, 42, 30, 22] that directly compute a score matrix and apply differentiable weighted SVD to solve for the transformation. Correspondence-based methods can fail in extremely low-overlap scenarios in shape assembly and direct methods fall short in terms of pose accuracy. Our method, which directly regresses the coordinates of each point in the source point cloud, is agnostic and more generalizable to varying overlap ratios compared to direct methods.

Multi-Part Registration and Assembly. Multi-part registration and shape assembly generalize pairwise relative pose estimation to multiple parts, with applications in furniture assembly [17] and shape reassembly [12]. Methods [43, 44, 26, 45, 46] tackle the multi-part registration problem by estimating the transformation for each rigid part in the scene (multi-source and multi-target). Multi-part shape assembly differs as a task from registration because it has multi-source input and a canonical target, and each part has almost 'zero' overlap with each other. Chen *et al*. [47] adopt an adversarial learning scheme to examine the plausibility for different shape configurations. Wu *et al*. [48] leverage $\mathrm{SE}(3)$-equivariant representation to handle pose variations in shape assembly. DiffAssembly [49] and PuzzleFusion [24, 25] leverage diffusion models to predict the transformation for each part. GARF [11] combines fracture-aware pretraining with a flow matching model to predict per-part transformation. However, these approaches do not explicitly address part symmetry or interchangeability as effectively as our method. A concurrent work [50] tackles multi-part assembly via equivariant flow matching, relying on an $\mathrm{SE}(3)$-equivariant network and carefully designed flow trajectories to account for part symmetry; in contrast, we handle the part symmetry with a simpler straight-line flow formulation in Euclidean space. Moreover, Rectified Point Flow is the first class-agnostic solution for furniture assembly on the PartNet [15] and IKEA-Manual [17] datasets.

3 Pose Estimation via Rectified Point Flow

Rectified Point Flow addresses the multi-part point cloud pose estimation problem, defined in Sec. 3.1. The overall pipeline consists of two consecutive stages: self-supervised overlap-aware point encoding (Sec. 3.2) and conditional Rectified Point Flow (Sec. 3.3). Finally, we explain how our formulation inherently addresses the challenges posed by symmetric and interchangeable parts in Sec. E.

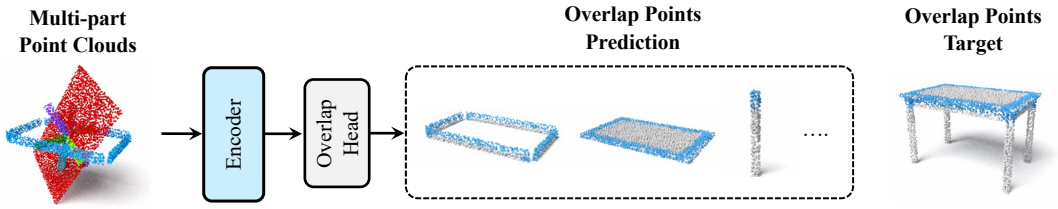

Figure 2: **Encoder pre-training via overlap points prediction.** Given unposed multi-part point clouds, our encoder with a point-wise overlap prediction head performs a binary classification to identify overlapping points. Predicted overlap points are shown in blue. For comparison, the ground-truth overlap points are visualized on the assembled object for clarity (target overlap).

3.1 Problem Definition

Consider a set of unposed point clouds of multiple object parts, $\{\boldsymbol{X}_i \in \mathbb{R}^{3\times N_i}\}_{i\in\Omega}$, where Ω is the part index set, $H := |\Omega|$ is the number of parts, and N_i is the number of points in part i. The goal is to solve for a set of rigid transformations $\{T_i \in \mathrm{SE}(3)\}_{i\in\Omega}$ that align each part in the unposed multi-part point cloud $\boldsymbol{X}$ to form a single, assembled object $\boldsymbol{Y}$ in a global coordinate frame, where

$$\boldsymbol{X} := \bigcup_{i\in\Omega} \boldsymbol{X}_i \in \mathbb{R}^{3\times N}, \quad \boldsymbol{Y} := \bigcup_{i\in\Omega} T_i\boldsymbol{X}_i \in \mathbb{R}^{3\times N}, \quad \text{and } N := \sum_{i\in\Omega} N_i. \tag{1}$$

To eliminate global translation and rotation ambiguity, we set the first part ($i = 0$) as the anchor and define its coordinate frame as the global frame. All other parts are registered to this anchor.

3.2 Overlap-aware Point Encoding

Pose estimation relies on geometric cues from mutually overlapping regions among connected parts [29, 32, 11]. In our work, we address this challenge through a self-supervised pretraining approach that develops a task-agnostic, overlap-aware encoder capable of producing pose-invariant point features. As illustrated in Fig. 2, we train an encoder F to identify overlapping points in different parts. Given a set of unposed parts $\{\boldsymbol{X}_i\}_{i\in\Omega}$, we first apply random rigid transforms $\tilde{T}_i \in \mathrm{SE}(3)$ and compose transformed point clouds $\tilde{\boldsymbol{X}}_i = \tilde{T}_i\boldsymbol{X}_i$ as input to the encoder. These data augmentations enable the encoder to learn more robust pose-invariant features. The encoder then computes per-point features $C_{i,j} \in \mathbb{R}^d$ for the j-th point on part i, after which an MLP overlap prediction head estimates the overlap probability $\hat{p}_{i,j}$. The binary ground-truth label $p_{i,j}$ is 1 if point $\tilde{\mathbf{x}}_{i,j}$ falls within radius ϵ of at least one point in other parts.

We train both the encoder and the overlap head using binary cross-entropy loss. For objects without predefined part segmentation, we employ off-the-shelf 3D part segmentation methods to generate the necessary labels. The features extracted by our trained encoder subsequently serve as conditioning input for our Rectified Point Flow model.

3.3 Generative Modeling for Pose Estimation

The overlap-aware encoder identifies potential overlap regions between parts but cannot determine their final alignment, particularly in symmetric objects that allow multiple valid assembly configurations. To address this limitation, we formulate the point cloud pose estimation as a *conditional generation task*. With this approach, Rectified Point Flow leverages the extracted point features to sample from the conditional distribution of all feasible assembled states across multi-part point clouds, generating estimates that maximize the likelihood of the conditional input point cloud. By recasting pose estimation as a generative problem, we naturally accommodate the inherent ambiguities arising from symmetry and part interchangeability in the data.

Preliminaries. Rectified Flow (RF) [51, 52] is a score-free generative modeling framework that learns to transform a sample $\boldsymbol{X}(0)$ from a source distribution, into $\boldsymbol{X}(1)$ from a target distribution. The forward process is defined as linear interpolation between them with a timestep t as

$$\boldsymbol{X}(t) = (1-t)\boldsymbol{X}(0) + t\boldsymbol{X}(1), \quad t \in [0,1]. \tag{2}$$

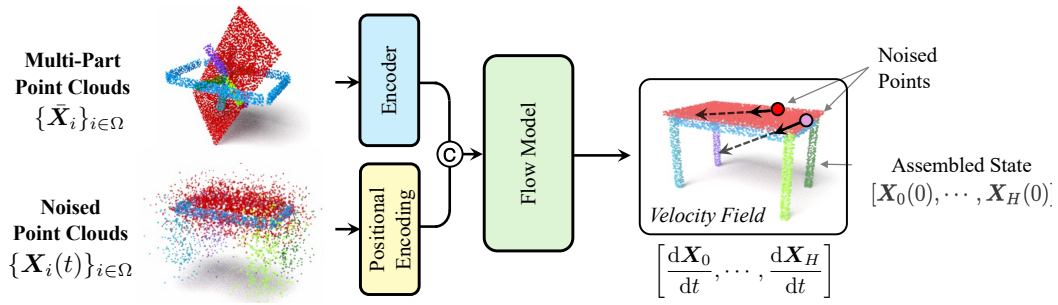

Figure 3: **Learning Rectified Point Flow.** The input to Rectified Point Flow are the condition point clouds $\{\tilde{\boldsymbol{X}}_i\}_{i\in\Omega}$ and noised point clouds $\{\boldsymbol{X}_i(t)\}_{i\in\Omega}$ at timestep t. They are first encoded by the pre-trained encoder and the positional encoding, respectively. The encoded features are concatenated and passed through the flow model, which predicts per-point velocity vectors $\{\mathrm{d}\boldsymbol{X}_i(t)/\mathrm{d}t\}_{i\in\Omega}$ and defines the flow used to predict the part point cloud in its assembled state.

The reverse process is modeled as a velocity field $\nabla_t \boldsymbol{X}(t)$, which is parameterized as a network $\boldsymbol{V}(t, \boldsymbol{X}(t) \mid \boldsymbol{X})$ conditioned on $\boldsymbol{X}$ and trained using conditional flow matching (CFM) loss [53],

$$\mathcal{L}_{\mathrm{CFM}}(\boldsymbol{V}) = \mathbb{E}_{t,\boldsymbol{X}}\left[\left\|\boldsymbol{V}(t, \boldsymbol{X}(t) \mid \boldsymbol{X}) - \nabla_t \boldsymbol{X}(t)\right\|^2\right]. \tag{3}$$

Rectified Point Flow. In our method, we directly apply RF to the 3D Euclidean coordinates of the multi-part point clouds. Let $\boldsymbol{X}_i(t) \in \mathbb{R}^{3\times M_i}$ denote the time-dependent point cloud for part i, where M_i is number of sampled points. At $t = 0$, $\{\boldsymbol{X}_i(0)\}_{i\in\Omega}$ is uniformly sampled from the assembled object $\boldsymbol{Y}$, while at $t = 1$, points on each part are independently sampled from a Gaussian, *i.e.*, $\boldsymbol{X}_i(1) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. Then, we define the continuous flow for each part as straight-line interpolation in 3D Euclidean space between the points in noised and assembled states. Specifically, for each part i,

$$\boldsymbol{X}_i(t) = (1-t)\boldsymbol{X}_i(0) + t\boldsymbol{X}_i(1), \quad t \in [0, 1]. \tag{4}$$

The velocity field of Rectified Point Flow is therefore,

$$\frac{\mathrm{d}\boldsymbol{X}_i(t)}{\mathrm{d}t} = \boldsymbol{X}_i(1) - \boldsymbol{X}_i(0). \tag{5}$$

We fix the anchored part ($i = 0$) by setting $\boldsymbol{X}_0(t) = \boldsymbol{X}_0(0)$ for all $t \in [0, 1]$, implemented via a mask that zeros out the velocity for its points. Once the model predicts the assembled point cloud of each part $\hat{\boldsymbol{X}}_i(0)$, we recover its pose T_i in a Procrustes problem,

$$\hat{T}_i = \underset{\hat{T}_i\in\mathrm{SE}(3)}{\arg\min} \|\hat{T}_i\boldsymbol{X}_i - \hat{\boldsymbol{X}}_i(0)\|_F. \tag{6}$$

Solving poses $\hat{T}_i$ for all non-anchored parts via SVD completes the pose estimation task in Eq. 1.

Learning Objective. We train a flow model $\boldsymbol{V}$ to recover the velocity field in Eq. 5, taking the noised point clouds $\{\boldsymbol{X}_i(t)\}_{i\in\Omega}$ and conditioning on unposed multi-part point cloud $\boldsymbol{X}$, as shown in Fig. 3. First, we encode $\boldsymbol{X}$ using the pre-trained encoder F. For each noised point cloud, we apply a positional encoding to its 3D coordinates and part index, concatenate these embeddings with the point features, and feed the result into the flow model. We denote its predicted velocity field by the flow model for all points by $\boldsymbol{V}(t, \{\boldsymbol{X}_i(t)\}_{i\in\Omega}; \boldsymbol{X}) \in \mathbb{R}^{3\times M}$. We optimize the flow model $\boldsymbol{V}$ by minimizing the conditional flow matching loss in Eq. 3.

3.4 Invariance Under Rotational Symmetry and Interchangeability

In our method, the straight-line point flow and point-cloud sampling, while simple, guarantee that every flow realization and its loss in Eq. (3) remain invariant under an assembly symmetry group $\mathcal{G}$:

Theorem 1 ($\mathcal{G}$-invariance of the learning objective)**.** *For every element $g \in \mathcal{G}$, we have the learning objective in Eq. 3 following $\mathcal{L}_{\mathrm{CFM}}(\boldsymbol{V}) = \mathcal{L}_{\mathrm{CFM}}(g(\boldsymbol{V}(t, \{\boldsymbol{X}_i(t)\}_{i\in\Omega}; g(\boldsymbol{X}))))$.*

The formal definition of $\mathcal{G}$ and the proof of Theorem 1 appear in the supplementary material. As a result, the flow model learns all the symmetries in $\mathcal{G}$ during training, without the need for additional hand-made data augmentation or heuristics on symmetry and interchangeability.

4 Experiments

Implementation Details. We use PointTransformerV3 (PTv3) [54] as the backbone for point cloud encoder, and use Diffusion Transformer (DiT) [55] as our flow model. Each DiT layer applies two self-attention stages: *(i)* part-wise attention to consolidate part-awareness, and *(ii)* global attention over all part tokens to fuse information. We stabilize the attention computation by applying RMS Normalization [56, 57] to the query and key vectors per head before attention operations. We sample the time steps from a U-shaped distribution following [58]. We pre-train the PTv3 encoder on all datasets with an additional subset of Objaverse [14] meshes, where we apply PartField [13] to obtain annotations. After pretraining, we freeze the weights of the encoder. We train our flow model on 8 NVIDIA A100 80GB GPUs for 400k iterations with an effective batch size of 256. We use the AdamW [59] optimizer with an initial learning rate 5×10^{-4} which is halved every 25k iterations after the first 275k iterations.

Table 1: **Dataset Statistics.** We train our model on seven datasets with varying sizes and complexity.

Dataset	Task	Train & Val		Test	
		# Samples	# Parts	# Samples	# Parts
IKEA-Manual [17]	Assembly	84	[2, 19]	18	[2, 19]
TwoByTwo [16]	Assembly	308	[2, 2]	144	[2, 2]
PartNet-Assembly	Assembly	23755	[2, 64]	261	[2, 64]
BreakingBad [12]	Assembly	35114	[2, 49]	265	[2, 49]
TUD-L [18]	Registration	19138	[2, 2]	300	[2, 2]
ModelNet 40 [19]	Registration	19680	[2, 2]	260	[2, 2]
Objaverse [14]	*Pre-training Only*	63199	[3, 12]	6794	[3, 12]

4.1 Experimental Setting

Datasets. For the multi-part shape assembly task, we experiment on the BreakingBad [12], TwoByTwo [16], PartNet [15], and IKEA-Manual [17] datasets. The PartNet dataset has been processed for the shape assembly task following the same procedure as [17] but includes all object categories; we refer to this version as PartNet-Assembly. Evaluation of the pairwise registration is performed on the TUDL [18] and ModelNet 40 [19] datasets. We split all datasets into train/val/test sets following existing literature for fair comparisons. The statistics of all datasets are in Tab. 1.

Evaluation Metrics. We evaluate the pose accuracy following the convention of each benchmark, with Rotation Error (RE), Translation Error (TE), Rotation Recall at 5° (Recall @ 5°), and Translation Recall at 1 cm (Recall @ 1 cm). For the shape assembly task, we measure Part Accuracy (Part Acc) by computing per object the fraction of parts with Chamfer Distance under 1 cm, and then averaging those per-object scores across the dataset, following [25, 11, 30, 17].

Baseline Methods. We evaluate our method against state-of-the-art methods for pairwise registration and shape assembly. For pairwise registration, we compare against DCPNet [31], RPMNet [30], GeoTransformer [32], and Diff-RPMNet [22]. For shape assembly, we compare against MSN [47], SE(3)-Assembly [48], Jigsaw [60], PuzzleFussion++ [25], and GARF [11]. We did not include the results of [50] because it is evaluated only on samples up to 8 parts (in BreakingBad-Everyday) and their models and codes have not yet been released. We report our performance in two training configurations: dataset-specific training in which models are trained independently for each dataset (*Ours (Single)*), and joint training in which a single model is trained on all datasets (*Ours (Joint)*).

4.2 Evaluation

We report pose accuracy for shape assembly and pairwise registration in Tab. 2 [2] and Tab. 3, respectively. Our model outperforms all existing approaches by a substantial margin. For **multi-part**

[2]We found that the BreakingBad benchmark [12, 11, 25] originally computed rotation error (RE) using the RMSE of Euler angles, which is not a proper metric on $\mathrm{SO}(3)$. To ensure consistency, we re-evaluate GARF using the geodesic distance between rotation matrices via the Rodrigues formula [61, 31, 29, 32].

Figure 4: **Qualitative Results on PartNet-Assembly.** Columns show objects with increasing number of parts (left to right). Rows display (1) colored input point clouds of each part, (2) GARF outputs (dashed boxes indicate samples limited to 20 by GARF's design, selecting the top 20 parts by volume), (3) Rectified Point Flow outputs, and (4) ground-truth assemblies. Compared to GARF, our method produces more accurate pose estimation on most parts, especially as the number of parts increases.

Table 2: **Multi-part Assembly Results.** Rectified Point Flow (Ours) achieves the best performance across all metrics on BreakingBad-Everyday, TwoByTwo, and PartNet-Assembly datasets.

Methods	**BreakingBad-Everyday** [12]			**TwoByTwo** [16]		**PartNet-Assembly**		
	RE ↓ [deg]	TE ↓ [cm]	Part Acc ↑ [%]	RE ↓ [deg]	TE ↓ [cm]	RE ↓ [deg]	TE ↓ [cm]	Part Acc ↑ [%]
MSN [47]	85.6	15.7	16.0	70.3	28.4	–	–	–
SE(3)-Assembly [48]	73.3	14.8	27.5	52.3	23.3	–	–	–
Jigsaw [60]	42.3	10.7	68.9	53.3	36.0	–	–	–
PuzzleFusion++ [25]	38.1	8.0	76.2	58.2	34.2	–	–	–
GARF [11]	9.9	2.0	93.0	22.1	7.1	66.9	21.9	25.7
***Ours** (Single)*	9.6	**1.8**	**93.5**	18.7	4.1	24.8	15.4	50.2
***Ours** (Joint)*	**7.4**	2.0	91.1	**13.2**	**3.0**	**21.8**	**14.8**	**53.9**

assembly, the closest competitor is GARF [11], which formulates per-part pose estimation as 6-DoF pose regression; see Figs. 4 and 5. We attribute our superior results to two key advantages of Rectified Point Flow: (i) in contrast to our closest competitor GARF [11] which performs 6-DoF pose regression, our dense shape-and-pose parametrization helps the model learn better global shape prior and fine-grained geometric details more effectively; and (ii) our generative formulation natively handles part symmetries and interchangeability. For **pairwise registration**, GARF–despite being retrained on target datasets–fails to generalize beyond the original task. In contrast, our method achieves a new state-of-the-art performance on registration benchmarks, outperforming methods designed solely for registration (*e.g.*, GeoTransformer [32] and Diff-RPMNet [22]) and demonstrating strong generalization across different datasets (Fig 5). We also achieve the strongest shape assembly performance on IKEA-Manual [17]; for more details on evaluation and visualizations, see supplementary.

Joint Training. By recasting pairwise registration as a two-part assembly task, our unified formulation enables joint training on all six datasets—including very small sets like TwoByTwo (308 samples) and IKEA-Manual (84 samples). *Ours (Joint)* consistently matches or outperforms the

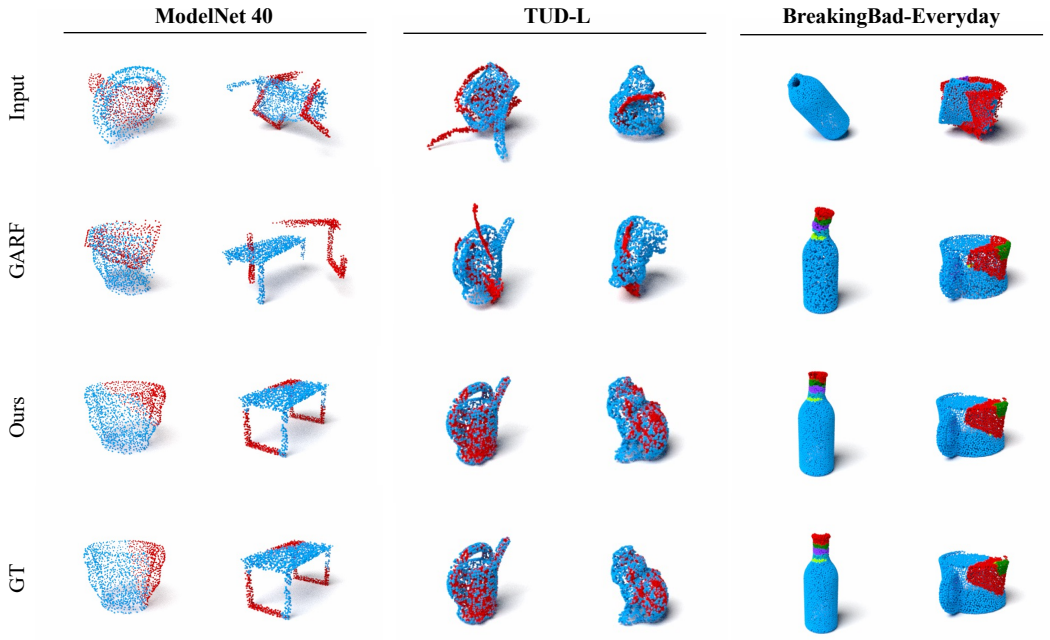

Figure 5: **Qualitative Results Across Registration and Assembly Tasks.** From left to right: pairwise registration on ModelNet 40 and TUD-L, shape assembly on BreakingBad-Everyday. From top to bottom: Colored input point clouds, GARF results, ours, and ground truth (GT). Our single model performs the best across registration and assembly tasks.

Table 3: **Pairwise Registration Results.** Rectified Point Flow (Ours) outperforms all baselines on both TUD-L and ModelNet 40, achieving the highest accuracy and lowest errors across all metrics.

Methods	**TUD-L** [18]		**ModelNet 40** [19]	
	Recall @5° ↑ [%]	Recall @1cm ↑ [%]	RE ↓ [deg]	TE ↓ [unit]
DCPNet [31]	23.0	4.0	11.98	0.171
RPMNet [30]	73.0	89.0	1.71	0.018
GeoTransformer [32]	88.0	97.5	1.58	0.018
GARF [11]	53.1	52.5	42.5	0.063
Diff-RPMNet [22]	90.0	98.0	–	–
***Ours** (Single)*	<u>97.0</u>	<u>98.7</u>	<u>1.37</u>	<u>0.003</u>
***Ours** (Joint)*	**97.7**	**99.0**	**0.93**	**0.002**

dataset-specific (*Ours (Single)*) models. For example, on TwoByTwo the rotation error drops from 18.7° to 13.2° (≈30%), and on BreakingBad from 9.6° to 7.4° (≈23%), while on ModelNet40, the rotation error is reduced from 1.37° to 0.93°. These results demonstrate that joint training enables the model to learn shared geometric priors—such as part symmetries, common pose distributions, and cross-dataset correlations—which substantially boosts performance particularly on datasets with limited training samples.

Symmetry Handling. We demonstrate our model's ability to handle symmetry (Sec. E) on IKEA-Manual [17], a dataset with symmetric assembly configurations. As shown in Fig. 6, while being only trained on a single configuration (left), Rectified Point Flow samples various reasonable assembly configurations (right), conditioned on the same input unposed point clouds. Note how our model permutes the 12 repetitive vertical columns and swaps the two middle baskets, yet always retains the non-interchangeable top and footed bottom baskets in their unique positions.

Ablation on Self-supervised Pretraining. Tab. 4 compares four pretraining strategies for our flow-based assembly model on BreakingBad–Everyday [12]. The first two encoders (MLP and

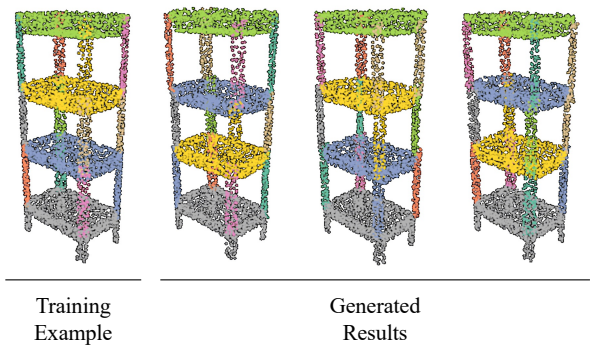

Figure 6: **Learning from a symmetric assembly.** Left to right: (1) a single training sample from IKEA-Manual [17], and (2–4) three independent samples generated, conditioned on the same inputs. Parts are color-coded consistently across plots. (*Best viewed in color.*)

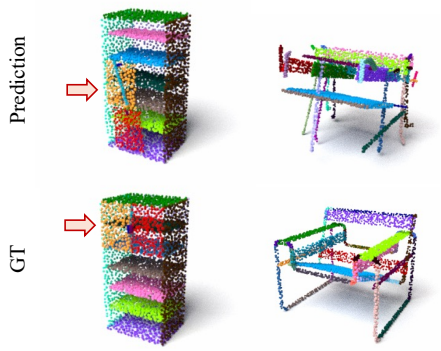

Figure 7: **Two common failure types.** First column: Assemblies that are geometrically plausible but mechanically non-functional. Second column: Objects with high geometric complexity.

Table 4: **Ablation on Encoder Pre-training.** We ablate the impact of different pre-training tasks on the shape assembly performance. Our overlap detection pre-training yields the best results.

Encoder	**Pre-training Task**	**BreakingBad-Everyday** [12]		
		RE ↓ [deg]	TE ↓ [cm]	Part Acc ↑ [%]
MLP	*No Pre-training*	41.7	12.3	68.3
PTv3	*No Pre-training*	18.5	4.9	79.5
PTv3	Instance Segmentation	16.7	4.4	80.9
PTv3	Overlap Detection (*ours*)	**9.6**	**1.8**	**93.5**

PTv3 without pre-training) are trained jointly with the flow model. The last two encoders are PTv3 pretrained on instance segmentation and our overlap-aware prediction tasks, respectively. Their pretrained weights are frozen during flow model training. We find that PTv3 is a more powerful feature encoder compared to the MLP, and pretraining on instance segmentation can already extract useful features for pose estimation, while our proposed overlap-aware pretraining leads to the best accuracy. We hypothesize that, although the segmentation backbone provides strong semantic features, only our overlap prediction task explicitly encourages the encoder to learn fine-grained part interactions and pre-assembly cues—critical for precise assembly and registration.

5 Conclusion

We introduce Rectified Point Flow, a unified flow-based framework for point cloud pose estimation across registration and assembly tasks. By modeling part poses as velocity fields, it captures fine geometry, handles symmetries and part interchangeability, and scales to varied part counts via joint training on 100K shapes. Our two-stage pipeline—overlap-aware encoding and rectified flow training—achieves state-of-the-art results on six benchmarks. Our work opens up new directions for robotic manipulation and assembly by enabling precise, symmetry-aware motion planning.

Limitations and Future Work. While our experiments focus on object-centric point clouds, real-world scenarios often involve cluttered scenes and partial observations. Moreover, while our model can generate multiple plausible assemblies, some of these may not be mechanically functional; see Fig. 7 (first column). Also, our model cannot handle objects that exceed a certain geometric complexity; see Fig. 7 (second column). Another limitation arises from the number of points our model can handle, which may restrict its usage on large-scale objects. Future work will extend Rectified Point Flow to robustly handle occlusion, support scene-level and multi-body registration, incorporate object-function reasoning, and scale to objects with larger point clouds.

Broader Impact. Rectified Point Flow makes it easier to build reliable 3D alignment and assembly systems directly from raw scans—benefiting robotics, digital manufacturing, AR, and heritage

reconstruction. However, the model can still produce incorrect, hallucinated, or nonfunctional assemblies. For safety, further work on assembly verification and error detection will be essential.

Acknowledgments

Tao Sun is supported by the Stanford Graduate Fellowship (SGF). Liyuan Zhu is partially supported by SPIRE Stanford Student Impact Fund Grant. Shuran Song is supported by the NSF Award #2037101. The authors also thank Stanford Marlowe Cluster [62] for providing GPU resources.

Supplementary Material

In this supplementary material, we provide the following:

- **Model Details** (Sec. A): Description of the DiT architecture and positional encoding scheme.
- **Additional Evaluation** (Sec. B): Comparison against category-specific assembly models on PartNet and IKEA-Manual, evaluation on the preservation of rigidity at the part level, and analysis of different generative formulations.
- **Randomness in Assembly Generation** (Sec. C): Investigation of the assemblies generated through noise sampling and linear interpolation in the noise space.
- **Generalization Ability** (Sec. D): Qualitative results on unseen assemblies with same- or cross-category parts to test the model's generalization ability.
- **Proof of Theorem 1** (Sec. E): Formal definition of the assembly symmetry group $\mathcal{G}$ and complete proof of the flow invariance under the group $\mathcal{G}$.
- **Generalization Bounds** (Sec. F): Derivation of the generalization risk guarantees and comparison with that of existing 6-DoF methods.

A Model Details

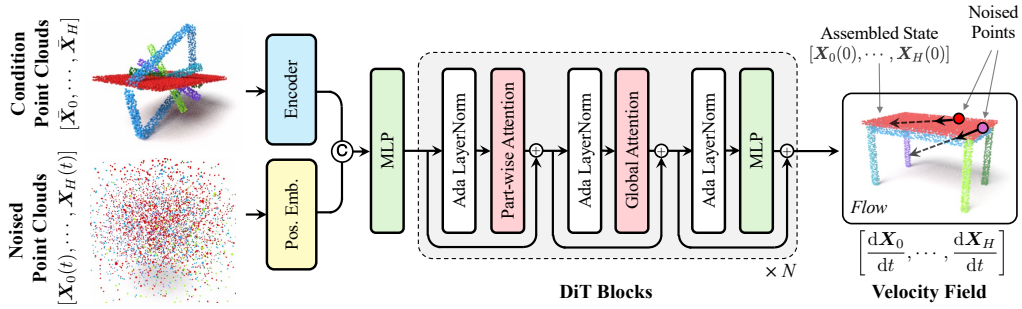

Figure 8: **Details of the DiT Block.** Our flow model consists of an Encoder and a position embedding (Pos. Emb.), and sequential DiT blocks ($N = 6$). Each block comprises Part-wise Attention, Global Attention, MLP, and AdaLayerNorm layers.

DiT Architecture. Our flow model consists of 6 sequential DiT [55] blocks, each with a hidden dimension of 512. For the multi-head self-attention in the DiT block, we set the number of attention heads to 8, resulting in a head dimension of 64. As illustrated in Figure 8, inspired by [63], we apply separated *Part-wise Attention* and *Global Attention* operations in each DiT block to capture both intra-part and inter-part context:

- **Part-wise Attention:** Points within each part independently undergo a self-attention operation, improving the model's ability to capture local geometric structures.
- **Global Attention:** Subsequently, global self-attention operation is applied to all points across parts, facilitating inter-part information exchange.

As discussed in Sec. 4, we apply RMS normalization individually to the query and key features in each attention head before both attention operations to enhance numerical stability during training. Additionally, every DiT block employs AdaLayerNorm, a layer normalization whose scaling and shifting parameters are modulated by the time step t, following [55].

Positional Encoding. We adopt a multi-frequency Fourier feature mapping [64], to encode spatial information in both the condition and noised point clouds. For the j-th point in the i-th part in the condition point cloud, $\mathbf{x}_{i,j} \in \boldsymbol{X}$, we construct a 10-dimensional vector, which comprises:

- The 3D absolute coordinates of $\mathbf{x}_{i,j}$.
- The 3D surface normal $\boldsymbol{n}_{i,j}$ at that point $\mathbf{x}_{i,j}$ in the condition point cloud.
- The 3D absolute coordinates of the noised point cloud $\boldsymbol{X}(t)$ at the index (i, j).
- The scalar part index i.

Each of these vectors is mapped through sinusoidal embeddings at multiple frequencies and then concatenated with the point-wise feature output of encoder F.

Inference. At inference time, we recover the assembled point cloud of each part by numerically integrating the predicted velocity fields $\boldsymbol{V}(t, \{\boldsymbol{X}_i(t)\}_{i\in\Omega} \mid \boldsymbol{X})$ from $t = 1$ to $t = 0$. In practice, we perform K uniform Euler steps as,

$$\hat{\boldsymbol{X}}(t - \Delta t) = \hat{\boldsymbol{X}}(t) - \boldsymbol{V}(t, \{\boldsymbol{X}_i(t)\}_{i\in\Omega} \mid \boldsymbol{X})\Delta t, \quad \text{where } \Delta t = 1/K.$$

After K iterations, the resulting $\hat{\boldsymbol{X}}(0)$ approximates the point clouds of all parts in the assembled state. For all evaluations, we set $K = 20$.

B Additional Evaluation

Comparison with Category-specific Models. We compare against category-specific point-cloud assembly methods in Table 5. All baselines are trained separately for each category, and the category labels are assumed to be known at inference time. RGL-Net [65] additionally assumes a top-to-bottom ordering of the input parts. In contrast, Rectified Point Flow is *class-agnostic* and performs inference without any class label or part ordering. We evaluated both shape Chamfer Distance (CD) and Part Accuracy (PA) in PartNet-Assembly and IKEA-Manual, following the protocol of Huang *et al*. [46].

Without category or ordering assumptions like the baseline methods, our joint model still achieves the lowest CD and matches or exceeds the PA of category-specific baselines optimized for each category (chair, table, lamp). In particular, we observe a relative improvement of 110.2% on Lamps PA over the strongest baseline. In IKEA-Manual, we observe that all category-specific models collapse to PA $\leq 6.9\%$ for the Chair category. We hypothesize that the baselines' architecture and hyperparameter are largely tailored to PartNet. In contrast, our joint model achieves 29.9% PA for the Chair category and 33.2% PA for all categories, over 4 times higher than any baselines. Those observations confirm that our category-agnostic cross-dataset training improves the learning of shared geometric priors far beyond any single category or dataset.

Part-level Rigidity Preservation. As a dense point map prediction framework, Rectified Point Flow is not explicitly trained to preserve the rigidity of each part. To quantify how well it preserves the rigidity of the parts, we first align each predicted part $\hat{\boldsymbol{X}}_i(0)$ with the part in assembled state $\boldsymbol{X}_i(0)$ using the Kabsch algorithm. We then measure two rigidity preservation errors using (1) the Root Mean Square Error (RMSE) over all points and (2) the Overlap Ratio (OR) over all points at varying thresholds $\tau \in \{0.1, 0.2, 0.5, 1, 2\}$ cm. Specifically, for part i with M_i points, we compute

$$\text{RMSE} = \sqrt{\frac{1}{M_i}\sum_{j=1}^{M_i}\left\|T_i'\hat{\mathbf{x}}_{i,j}(0) - \mathbf{x}_{i,j}(0)\right\|^2} \quad \text{and}$$

$$\text{OR}\,(\tau) = \frac{1}{M_i}\left|\{j \mid \|T_i'\hat{\mathbf{x}}_{i,j}(0) - \mathbf{x}_{i,j}(0)\| < \tau\}\right|.$$

Table 5: **Comparison with Category-specific Models.** We report Shape Chamfer Distance (CD) and Part Accuracy (PA) on the PartNet-Assembly and IKEA-Manual. All baselines are trained per category, whereas Rectified Point Flow is trained over all categories. (*RGL-Net additionally requires a top-to-bottom part ordering.)

Method	Known Category	PartNet-Assembly								IKEA–Manual [17]			
		Chair		*Table*		*Lamp*		**All**		*Chair*		**All**	
		CD ↓ [cm]	PA ↑ [%]	CD ↓ [cm]	PA ↑ [%]	CD ↓ [cm]	PA ↑ [%]	CD ↓ [cm]	PA ↑ [%]	CD ↓ [cm]	PA ↑ [%]	CD ↓ [cm]	PA ↑ [%]
B-LSTM [66]	✓	1.31	21.8	1.25	28.6	0.77	20.8	–	–	1.81	3.5	–	–
B-Global [66]	✓	1.46	15.7	1.12	15.4	0.79	22.6	–	–	1.95	0.9	–	–
RGL-Net* [65]	✓	0.87	**49.2**	0.48	**54.2**	0.72	37.6	–	–	5.08	4.0	–	–
Huang *et al.* [66]	✓	0.91	39.0	0.50	49.5	0.93	33.3	–	–	1.51	6.9	–	–
Ours (*Joint*)	×	**0.71**	44.1	**0.36**	49.4	**0.49**	**70.0**	**0.48**	**53.9**	**1.49**	**29.9**	**0.48**	**33.2**

Table 6: **Part-level Rigidity Preservation Evaluation.** Rectified Point Flow demonstrates low shape discrepancy between condition and predicted part point clouds, measured by the Root Mean Square Error (RMSE), Relative RMSE, and Overlap Ratios (ORs) across datasets. D represents the average object scale of each dataset. (*Abbr*: BreakingBad-E = BreakingBad-Everyday; PartNet-A = PartNet-Assembly; IKEA-M = IKEA-Manual.)

Metric		Shape Assembly				Pairwise Registration	
		BreakingBad-E	TwoByTwo	PartNet-A	IKEA-M	TUD-L	ModelNet-40
Object Scale D	[cm] –	52.1	107.7	89.0	61.4	40.8	70.0
RMSE	[cm] ↓	0.76	2.46	1.04	0.66	0.16	0.30
Relative RMSE	[%] ↓	1.5	2.3	1.2	1.1	0.4	0.4
OR ($\tau = 0.1$ cm)	[%] ↑	52.3	63.8	33.1	46.7	96.9	95.0
OR ($\tau = 0.2$ cm)	[%] ↑	61.7	70.8	48.6	57.7	97.1	96.0
OR ($\tau = 0.5$ cm)	[%] ↑	74.9	76.8	66.8	69.8	97.4	96.3
OR ($\tau = 1$ cm)	[%] ↑	81.4	78.7	77.9	81.0	97.7	96.6
OR ($\tau = 2$ cm)	[%] ↑	89.5	81.9	87.4	92.0	98.2	97.1

Here, $T_i' \in \mathrm{SE}(3)$ denotes the optimal rigid transform returned by Kabsch; $\mathbf{x}_{i,j}$ and $\hat{\mathbf{x}}_{i,j}$ denote the j-th point on $\boldsymbol{X}_i(0)$ and on $\hat{\boldsymbol{X}}_i(0)$, respectively. Because each part is first rigidly aligned to the ground-truth assembled state, these metrics intentionally ignore pose errors, and only measure the shape difference between the predicted and ground truth point parts. To factor in the variations in object size across datasets, we compute the average scale of an object, denoted by D, as twice the average distance from the object's center of gravity to all its points. Then, we define the Relative RMSE as RMSE / D, *i.e.*, the RMSE normalized by the average object scale. We report these metrics averaged for all parts in each dataset in Table 6.

For the **pairwise registration** task, Rectified Point Flow demonstrates strong rigidity preservation. On TUD-L, we obtain a Relative RMSE of 0.4% and ORs above 96.9% even at the strictest $\tau = 0.1$ cm threshold; on ModelNet-40, we achieve the same Relative RMSE of 0.4% with similar high ORs above 95.0%. Specifically, on TUD-L we record ORs of 96.9% ($\tau = 0.1$ cm), 97.1% ($\tau = 0.2$ cm), 97.4% ($\tau = 0.5$ cm), 97.7% ($\tau = 1$ cm) and 98.2% ($\tau = 2$ cm); on ModelNet-40 the corresponding ORs are 95.0%, 96.0%, 96.3%, 96.6% and 97.1%, demonstrating consistently strong rigidity preservation.

In the more challenging **shape assembly** task, rigidity errors remain low. Across the four datasets, the Relative RMSE ranges from 1.1% to 2.3%. At a strict threshold of $\tau = 0.1$ cm, overlap ratios (ORs) span 33.1 % (PartNet-Assembly) up to 63.8 % (TwoByTwo); By $\tau = 1$ cm, the ORs exceed 77.9% in the four datasets (77.9%-81.4%), increasing further to 81.9%-92.0% in the more relaxed $\tau = 2$ cm. The highest Relative RMSE and lower averaged ORs are observed in TwoByTwo, probably due to its limited training samples and lower shape similarity to other datasets, and the fact that TwoByTwo has the largest overall object scale of 107.7 cm among all datasets. In contrast, IKEA-Manual, despite having fewer training samples, benefits from shared priors in furniture objects in joint training,

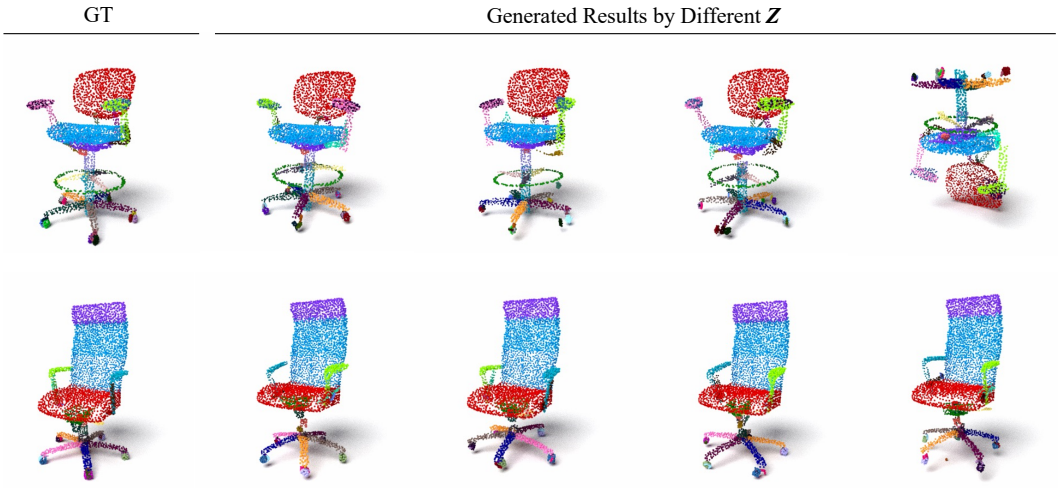

Figure 9: **Sampling in Noise Space.** For each fixed condition input point clouds, we sample four independent Gaussian noise vectors to generate distinct assembly outputs (shown in columns 2–5). While all samples preserve the object's structure, they show meaningful variation in part placement, orientation, and overall geometry, particularly for symmetric parts (*e.g.*, armrests and chair bases). For comparison, the first column shows the ground-truth assemblies.

delivering the lowest RMSE and high ORs at all thresholds. These results demonstrate robust rigidity preservation of Rectified Point Flow even in complex shape assembly scenarios.

Note that the subsequent pose recovery stage in Rectified Point Flow further refines part poses via an SVD-based global optimization, which fits optimal poses under noises. In Sec. F, we demonstrate that this SVD-based optimization is crucial to reducing the risk limits of generalization to match the rates achieved by the existing 6-DoF methods. Overall, we empirically confirm that Rectified Point Flow generates point clouds that reliably respect the rigid structure of the conditioning parts.

Ablation on Generative Formulation. As an alternative to the generative formulation of Rectified Flow (RF) in our method, we also evaluate a Denoising Diffusion Probabilistic Model (DDPM) [67] using an identical DiT architecture and the pre-trained encoder. In this setup, the forward noising process employs constant variances (β) that increase linearly from 10^{-4} to 0.02 over $T = 1000$ timesteps. As shown in Table 7, the RF-based model consistently outperforms the DDPM variant on both shape assembly and pairwise registration tasks, with 35.3% lower rotation error (RE) and 11.63% lower translation error (TE). This result is in line with the findings of GARF [11]. We hypothesize that the straight-line flow in RF reduces the learning difficulty in our tasks. DDPM's frequency-based generation—which works well for images—may not be as effective as RF for 3D point cloud synthesis in Euclidean space.

Table 7: **Generative Formulation Comparison.** We compare Rectified Flow (RF) with Denoising Diffusion Probabilistic Model (DDPM) in our method, with both using the same DiT architecture and pretrained encoder. RF achieves superior performance on Rotation Error (RE) and Translation Error (TE) across all datasets. (*Abbr*: BreakingBad-E = BreakingBad-Everyday; PartNet-A = PartNet-Assembly; IKEA-M = IKEA-Manual.)

Metric	**Generative Formulation**	**Shape Assembly**				**Pairwise Registration**	
		BreakingBad-E	TwoByTwo	PartNet-A	IKEA-M	TUD-L	ModelNet-40
RE [deg] ↓	DDPM	13.0	17.2	29.5	21.4	2.6	3.4
	RF	**7.4**	**13.2**	**21.8**	**10.8**	**1.4**	**0.9**
TE [cm] ↓	DDPM	3.5	10.1	21.3	19.2	0.5	0.7
	RF	**2.0**	**3.0**	**14.8**	**17.2**	**0.3**	**0.2**

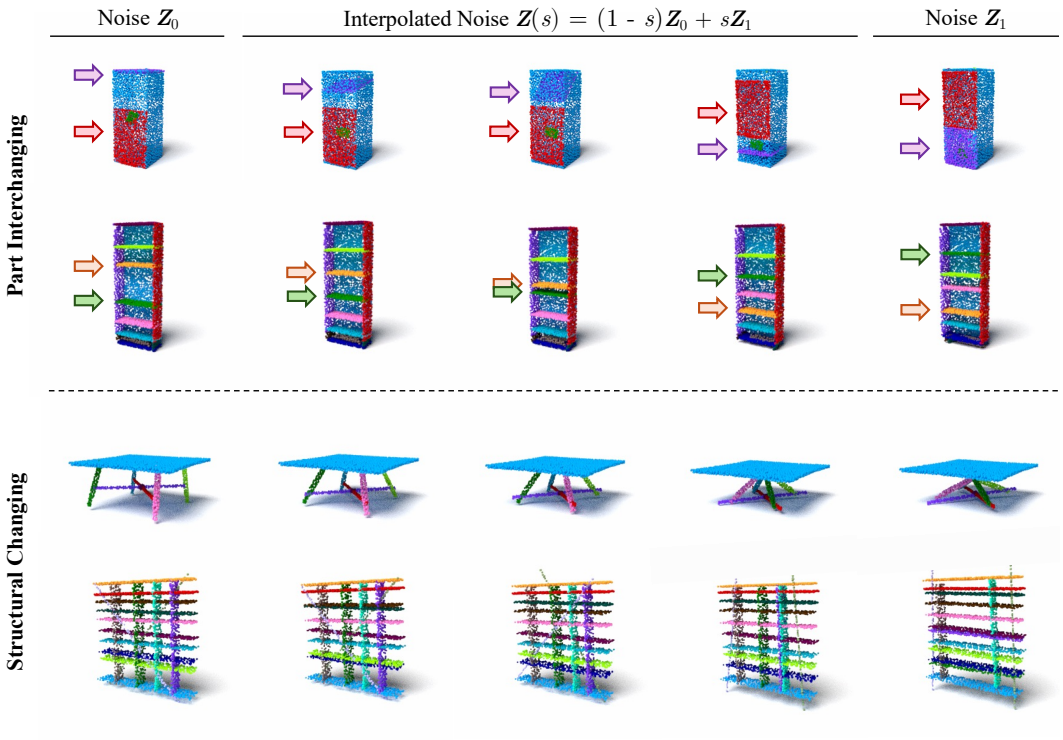

Figure 10: **Linear Interpolation in Noise Space.** For different objects in each row, we fix the same conditional input and decode two independently sampled Gaussian noise vectors, $\boldsymbol{Z}_0$ (leftmost) and $\boldsymbol{Z}_1$ (rightmost), into plausible part configurations. The three center columns show outputs from the linearly interpolated noises between $\boldsymbol{Z}_0$ and $\boldsymbol{Z}_1$. We observe a continuous, semantically meaningful mapping from Gaussian noise to valid assemblies.

C Randomness in Assembly Generation

Diversity via Noise Sampling. To evaluate the diversity of assembly configurations generated by Rectified Point Flow, we sample the Gaussian noise vector $\boldsymbol{Z}$ multiple times for the same conditional (unposed) point cloud inputs. At inference time, we set $\boldsymbol{X}(1) = \boldsymbol{Z}$ and run the model to obtain prediction $\hat{\boldsymbol{X}}(0)$. In Figure 9, each row corresponds to a single final assembly: the first column shows the ground-truth assembly, and the next four columns display outputs produced by four different Gaussian noises. All generated assemblies preserve the part structure, yet exhibit meaningful variations in the parts' placement and orientation, and overall geometry of the object. As expected, the model produces diverse configurations for symmetric or interchangeable parts, such as the armrests and the chair base. This shows that Rectified Point Flow effectively captures a diverse conditional distribution of valid assemblies.

Linear Interpolation in Noise Space. We illustrate Rectified Point Flow's learned mapping from random Gaussian noise to plausible assembly configurations. In Figure 10, each row uses the same condition (unposed) point cloud, with the left and right columns showing the outputs of two randomly sampled noise vectors $\mathbf{Z}_0$ and $\mathbf{Z}_1$, respectively. The three columns in between display results generated by $\mathbf{Z}(s)$ which linearly interpolates between $\mathbf{Z}_0$ and $\mathbf{Z}_1$ in noise space, *i.e.*,

$$\boldsymbol{Z}(s) := (1 - s)\boldsymbol{Z}_0 + s\boldsymbol{Z}_1, \quad \text{where } s \in \{0.25, 0.5, 0.75\}. \tag{7}$$

At each interpolation step s, we run inference with $\boldsymbol{X}(1) = \boldsymbol{Z}(s)$. As s increases, the predicted shapes smoothly morph from the configuration induced by $\boldsymbol{Z}_0$ toward that of $\boldsymbol{Z}_1$. As shown in the first 2 rows in Figure 10, we observe smooth transitions among interchangeable parts in both examples. The 2 bottom rows in Figure 10 visualize the transitions in the overall structure of objects. In the table example, we observe a gradual reduction in overall height, a lowering of the horizontal beams, and a more centralized positioning where the four legs meet. In the shelf example, the transformation

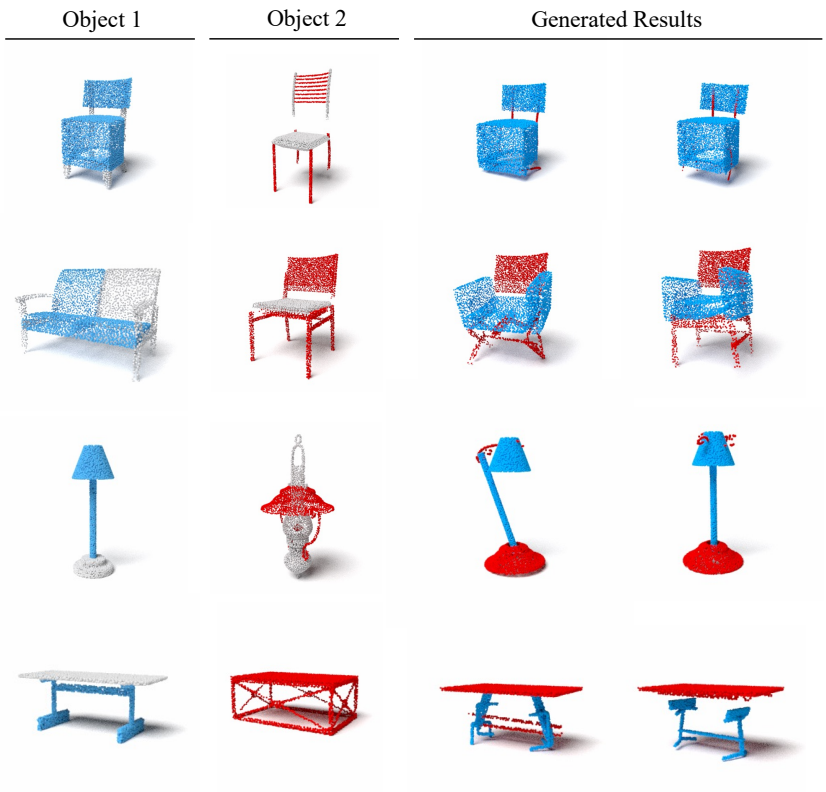

Figure 11: **Generalization to Unseen Assemblies Within the Same Category**: We select parts from two objects of the same category in the PartNet-Assembly test set. Parts from Object 1 are shown in blue, and parts from Object 2 in red; unselected parts are shown in gray. The results demonstrate that the model comprehends the underlying geometric structure of the category and can re-target parts to construct the final shape of the same category.

is more drastic: two vertical boards become horizontal and two diagonal cables are rearranged to a new vertical configuration. The above transitions across various assemblies confirm that Rectified Point Flow learns a continuous mapping from Gaussian noise to a semantically meaningful geometry space. Note that most of the interpolated configurations are physically plausible assemblies, creating functional objects that can stand in real-world.

D Generalization Ability

We test the generalization ability of our model for novel assemblies under two different settings: between objects from the same (in-category) and different (cross-category) categories. Given two objects in PartNet-Assembly, we select certain parts from each of them as the input to Rectified Point Flow to test if the model can generate novel and plausible assemblies.

In-category Test. As shown in Figure 11, parts selected from Object 1 are rendered in blue and those from Object 2 in red. Our model then synthesizes novel assemblies that blend and reconfigure these parts in a coherent and category-consistent manner. For example, in the chair category (first two rows), the model successfully retains a functional and plausible seat-back-leg structure while creatively mixing parts. In the lamp category (third row), even though the base and shade style differ significantly between objects, generated results exhibit sensible combinations that maintain structural integrity. Similarly, in the table category (last row), our method combines parts from a flat-top table and lattice-style base to produce hybrid yet coherent table designs.

Cross-category Test. Figure 12 highlights Rectified Point Flow's ability to generalize to unseen part combinations across categories. This is a particularly challenging test, since such part combination

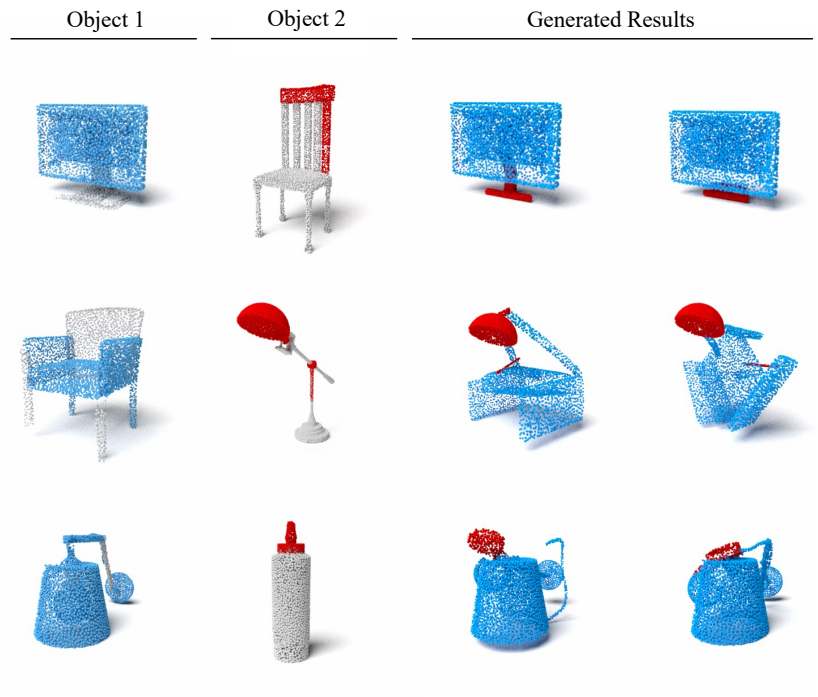

Figure 12: **Generalization to Unseen Assemblies Across Categories**: We select parts from two objects of different categories in the PartNet-Assembly test set. Parts from Object 1 are shown in blue, and parts from Object 2 in red; unselected parts are shown in gray. The results demonstrate that the model can reason about part compositionality and re-target parts to construct a plausible final shape even if some of them originate in completely different objects.

may not even be possible to be assembled into a meaningful object. Nevertheless, our method still demonstrates a certain degree of generalization. We show two input objects from different categories, for example, a monitor and a chair, a chair and a lamp, or a wall sconce and a spray bottle. The results on the right demonstrate that our model can reconfigure these parts into plausible new assemblies, preserving geometric coherence. This suggests that the model has learned a strong understanding of part relationship, allowing it to reason about compositionality even across category boundaries.

E Proof of Theorem 1

A key advantage of Rectified Point Flow is that it learns both rotational symmetries of individual parts and the interchangeability of a set of identical parts, without any labels of symmetry parts. Below, we first formally define an *assembly symmetry group* $\mathcal{G}$ that characterizes the symmetry and interchangeability of the parts in the multi-part point cloud.

Definition 1 (Assembly symmetry group)**.** *For each part $i \in \Omega$, let $G_i \subseteq \mathrm{SO}(3)$ be the (finite) stabilizer of its assembled shape,* i.e.*, $R\boldsymbol{X}_i(0) = \boldsymbol{X}_i(0)$ for all $R \in G_i$. Let $S \subseteq \mathfrak{S}_{|\Omega|}$ be the set of permutations that only permute indices of identical parts. We define the assembly symmetry group as the semidirect product*

$$\mathcal{G} = \big(G_1 \times \cdots \times G_{|\Omega|}\big) \rtimes S. \tag{8}$$

A group element $g = (R_1, \ldots, R_{|\Omega|}, \sigma) \in \mathcal{G}$ acts on every realization of the Rectified Point Flow by $g(\boldsymbol{X}_i(t)) := R_i\, \boldsymbol{X}_{\sigma^{-1}(i)}(t)$, and on network outputs of the i-th part (denoted as $\boldsymbol{V}_i$) by $g(\boldsymbol{V}_i(t, g(\boldsymbol{X}))) := R_i\, \boldsymbol{V}_{\sigma^{-1}(i)}(t, g(\boldsymbol{X}))$.

Now, we show the following result that a single point's flow distribution is invariant under any $g \in \mathcal{G}$.

Lemma 1 ($\mathcal{G}$-invariance of the flow distribution)**.** *For every element $g \in \mathcal{G}$ and a given multi-part point cloud $\boldsymbol{X}$, we sample a flow realization:*

$$\mathbf{x}(t) = t\mathbf{x}(1) + (1-t)\mathbf{x}(0), \quad \textit{where } \mathbf{x}(1) \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}), \mathbf{x}(0) \sim \boldsymbol{X}.$$

then, we have

$$p\big(\{g \cdot \mathbf{x}(t)\}_{t\in[0,1]}\big) = p(\{\mathbf{x}(t)\}_{t\in[0,1]}).$$

Proof. Recall that, in Rectified Point Flow, a flow of a single point is $\mathbf{x}(t) := (1-t)\mathbf{x}(0) + t\mathbf{x}(1)$, where $\mathbf{x}(0) \sim \boldsymbol{X}$ is drawn uniformly from the assembled shape and $\mathbf{x}(1) \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Because the end–points of the linear interpolation are sampled independently, the PDF of the path distribution factorizes as

$$p\big(\{\mathbf{x}(t)\}_{t\in[0,1]}\big) = p\big(\mathbf{x}(1)\big)p\big(\mathbf{x}(0)\big), \tag{9}$$

which indicates the randomness resides by the states $t = 0$ and $t = 1$ only. Because the perturbation $\mathbf{x}(1) \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is isotropic, $p(\mathbf{x}(1))$ is invariant under *every* rotation $R \in \mathrm{SO}(3)$. For $p(\mathbf{x}(0))$ we distinguish two cases:

- *Rotational symmetry:* If $R \in G_i$, then $R\boldsymbol{X}_i(0) = \boldsymbol{X}_i(0)$ point-wise, so $p(\mathbf{x}(0)) = p(R\mathbf{x}(0))$.
- *Interchangeability.* If parts i and j are identical, sampling first a part index with probability $p(i) = N_i/N$ and then a point uniformly inside it implies $p(\mathbf{x}(0) \in \boldsymbol{X}_i(0)) = p(\mathbf{x}(0) \in \boldsymbol{X}_j(0))$. Therefore exchanging the indices ($\sigma(i) = j, \sigma(j) = i$) leaves $p(\mathbf{x}(0))$ unchanged.

By composing the above two properties for all parts, we complete the proof. □

Lemma 1 can directly lift from single points to the full multi–part flow $\{\boldsymbol{X}_i(t)\}_{i\in\Omega}$. This leads us to the Theorem 1: For every element $g \in \mathcal{G}$, we have the learning objective in Eq. 3 following $\mathcal{L}_{\mathrm{CFM}}(\boldsymbol{V}) = \mathcal{L}_{\mathrm{CFM}}(g(\boldsymbol{V}(t, \{\boldsymbol{X}_i(t)\}_{i\in\Omega}; g(\boldsymbol{X}))))$.

F Generalization Bounds

While the Rectified Point Flow predicts a much higher-dimensional space ($3M_i$ coordinates per part), we find that its Rademacher complexity scales exactly the same rate as the 6-DoF methods, $O(1/\sqrt{m})$, where m is the number of samples in the training set.

Below, we compute their Rademacher complexities and empirical risks, respectively. Without loss of generality, we use the reconstruction error for the evaluation of poses, *i.e.*, $\ell(\hat{R}, \hat{\boldsymbol{t}}; R^\star, \boldsymbol{t}^\star) = \big\|(\hat{R} - R^\star)\boldsymbol{X}^\star + \hat{\boldsymbol{t}} - \boldsymbol{t}^\star\big\|_F$. First, we define hypothesis classes for both methods:

- Our Rectified Point Flow:

$$\mathcal{F}_i = \{C_i \mapsto \hat{\boldsymbol{X}}_i(0;\theta) \mid \theta \in \Theta\}, \quad \text{where } \hat{\boldsymbol{X}}_i(0;\theta) := \boldsymbol{X}_i(1) - \int_0^1 \boldsymbol{V}_i(t; C, \theta)\mathrm{d}t.$$

- Pose vector-based flow:

$$\mathcal{G}_i = \{C_i \mapsto (\hat{R}_i, \hat{\boldsymbol{t}}_i)_\phi \mid \phi \in \Phi\}.$$

Rademacher Complexity of Rectified Point Flow. With m *i.i.d.* training objects $D = \{(C^{(k)}, R^{\star(k)}, \boldsymbol{t}^{\star(k)})\}_{k=1}^m$, we write the population risk $\mathcal{R}(h) = \mathbb{E}\left[\ell\big(h(C), R^\star, \boldsymbol{t}^\star\big)\right]$ and empricial risk

$$\hat{\mathcal{R}}_D(h) = \frac{1}{m}\sum_{k=1}^m \ell\big(h(C^{(k)}), R^{\star(k)}, \boldsymbol{t}^{\star(k)}\big).$$

Since our Rectified Point Flow method estimates the part pose by the Procrustes operator, *i.e.*, $(\hat{R}, \hat{\boldsymbol{t}}) = \mathrm{Pr}\big(\hat{\boldsymbol{X}}(0;\theta)\big)$, where $\mathrm{Pr} : \mathbb{R}^{3N} \to \mathrm{SE}(3)$ is the Procrustes operator, we have following Lipschitz contracting property.

Property 1 (Lipschitz Contracting)**.** *Let $\boldsymbol{X}^\star \in \mathbb{R}^{3N}$ be the centralized ground-truth point set of a single part, and denote $\sigma_{\min} = \sigma_{\min}((\boldsymbol{X}^\star)^\top \boldsymbol{X}^\star)$. If $\|\hat{\boldsymbol{X}}(0) - \boldsymbol{X}^\star\|_F \le \varepsilon$, the optimal Procrustes solution $(\hat{R}, \hat{\boldsymbol{t}}) = P(\hat{\boldsymbol{X}}(0))$ satisfies*

$$\|(\hat{R} - R^\star, \hat{\boldsymbol{t}} - \boldsymbol{t}^\star)\| \;\le\; \frac{\varepsilon}{\sqrt{\sigma_{\min}}}. \tag{10}$$

This property directly follows from Davis–Kahan perturbation bounds [68] for the Top-3 singular vectors. Crucially, $\sigma_{\min} = \Omega(N)$[3] for well-spread point clouds, so Pr is a $\frac{1}{\sqrt{N}}$-*Lipschitz* map.

Let $\mathfrak{R}_m(\mathcal{H})$ denote the empirical Rademacher complexity on S. Because composition with a L-Lipschitz map contracts Rademacher complexity

$$\mathfrak{R}_m(\Pr \circ \mathcal{F}) \;\le\; \frac{1}{\sqrt{N}}\,\mathfrak{R}_m(\mathcal{F}) \;\le\; \frac{L_\Theta\sqrt{3N}}{\sqrt{N}}\,\frac{1}{\sqrt{m}} \;=\; O\Big(\frac{L_\Theta}{\sqrt{m}}\Big). \tag{11}$$

Rademacher Complexity of 6DoF-based Methods. For the baseline we need only regress $d = 6$ numbers, hence

$$\mathfrak{R}_m(\mathcal{G}) \;\le\; \frac{L_\Phi\sqrt{d}}{\sqrt{m}} = O\Big(\frac{L_\Phi}{\sqrt{m}}\Big). \tag{12}$$

Comparison of Generalization Bounds. Applying Bartlett Theorem and using (10), we obtain, with probability at least $1-\delta$ over the samples from D,

$$\mathcal{R}\big(P\circ\hat{f}\big) \;\le\; \hat{\mathcal{R}}_D\big(P\circ\hat{f}\big) + 2\,\mathfrak{R}_m(P\circ\mathcal{F}) + 3\sqrt{\frac{\log(2/\delta)}{2m}}, \tag{FLOW}$$

$$\mathcal{R}(\hat{g}) \;\le\; \hat{\mathcal{R}}_D(\hat{g}) + 2\,\mathfrak{R}_m(\mathcal{G}) + 3\sqrt{\frac{\log(2/\delta)}{2m}}, \tag{6DOF}$$

where $\hat{f}\in\mathcal{F}$ and $\hat{g}\in\mathcal{G}$ are the empirical-risk minimizers on S.

In conclusion, while Rectified Point Flow predicts a much higher-dimensional space, the contraction of the SVD stage cancels this apparent over-parameterization, producing a complexity term that scales at the same rate of $O(1/\sqrt{m})$ as the 6-DoF baseline; (FLOW)–(6DOF).

As a result, our method enjoys at least same generalization risk guarantees despite operating in an over-parameterized prediction space, while retaining the $\mathcal{G}$-invariance benefits proven in Sec. E.

References

[1] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser, "3dmatch: Learning local geometric descriptors from rgb-d reconstructions," in *CVPR*, 2017.

[2] Y. Li, A. Zeng, and S. Song, "Rearrangement planning for general part assembly," in *Conference on Robot Learning*, 2023.

[3] Y. Li, L. Jiang, Y. Liu, Y. Nie, H. Zhu, and D. Lin, "Gapartnet: Graph-structured assembly from part segments," in *ECCV*, 2022.

[4] J. Zhang, M. Wu, and H. Dong, "Generative category-level object pose estimation via diffusion models," *NeurIPS*, vol. 36, 2024.

[5] H. Zhao, S. Wei, D. Shi, W. Tan, Z. Li, Y. Ren, X. Wei, Y. Yang, and S. Pu, "Learning symmetry-aware geometry correspondences for 6d object pose estimation," in *ICCV*, 2023.

[6] T. Hodan *et al.*, "Bop challenge 2020 on 6d object localization," in *ECCV Workshops*, 2020.

[7] B. Tekin, S. Sinha, and P. Fua, "Real-time seamless single shot 6d object pose prediction," in *CVPR*, 2018.

[8] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song, and L. J. Guibas, "Normalized object coordinate space for category-level 6d object pose and size estimation," in *CVPR*, 2019.

[9] K. A. Murphy, C. Esteves, V. Jampani, S. Ramalingam, and A. Makadia, "Implicit-pdf: Non-parametric representation of probability distributions on the rotation manifold," in *ICML*, 2021.

[10] Y. Li, K. Mo, Y. Duan, H. Wang, J. Zhang, and L. Shao, "Category-level multi-part multi-joint 3d shape assembly," in *CVPR*, pp. 3281–3291, 2024.

[11] S. Li, Z. Jiang, G. Chen, C. Xu, S. Tan, X. Wang, I. Fang, K. Zyskowski, S. P. McPherron, R. Iovita, C. Feng, and J. Zhang, "Garf: Learning generalizable 3d reassembly for real-world fractures," *arXiv preprint arXiv:2504.05400*, 2025.

[12] S. Sellán, Y.-C. Chen, Z. Wu, A. Garg, and A. Jacobson, "Breaking bad: A dataset for geometric fracture and reassembly," *NeurIPS*, 2022.

[3] Here, $\Omega(\cdot)$ denote the asymptotic rate, instead of part index set.

[13] M. Liu, M. A. Uy, D. Xiang, H. Su, S. Fidler, N. Sharp, and J. Gao, "Partfield: Learning 3d feature fields for part segmentation and beyond," in *arxiv*, 2025.

[14] M. Deitke, D. Schwenk, J. Salvador, L. Weihs, O. Michel, E. VanderBilt, L. Schmidt, K. Ehsani, A. Kembhavi, and A. Farhadi, "Objaverse: A universe of annotated 3d objects," *arXiv preprint arXiv:2212.08051*, 2022.

[15] K. Mo, S. Zhu, A. X. Chang, L. Yi, S. Tripathi, L. J. Guibas, and H. Su, "Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding," in *CVPR*, 2019.

[16] Y. Qi, Y. Ju, T. Wei, C. Chu, L. L. Wong, and H. Xu, "Two by two: Learning multi-task pairwise objects assembly for generalizable robot manipulation," *CVPR*, 2025.

[17] R. Wang, Y. Zhang, J. Mao, R. Zhang, C.-Y. Cheng, and J. Wu, "Ikea-manual: Seeing shape assembly step by step," in *NeurIPS Datasets and Benchmarks Track*, 2022.

[18] T. Hodan, F. Michel, E. Brachmann, W. Kehl, A. GlentBuch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, *et al.*, "Bop: Benchmark for 6d object pose estimation," in *ECCV*, 2018.

[19] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *CVPR*, 2015.

[20] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *CVPR*, 2016.

[21] Z. Zhu, S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. R. Oswald, and M. Pollefeys, "Nice-slam: Neural implicit scalable encoding for slam," in *CVPR*, 2022.

[22] H. Jiang, M. Salzmann, Z. Dang, J. Xie, and J. Yang, "Se (3) diffusion model-based point cloud registration for robust 6d object pose estimation," in *NeurIPS*, 2023.

[23] L. Zhu, Y. Li, E. Sandström, S. Huang, K. Schindler, and I. Armeni, "Loopsplat: Loop closure by registering 3d gaussian splats," in *3DV*, 2025.

[24] S. Hosseini, M. A. Shabani, S. Irandoust, and Y. Furukawa, "Puzzlefusion: Unleashing the power of diffusion models for spatial puzzle solving," in *NeurIPS*, 2023.

[25] Z. Wang, J. Chen, and Y. Furukawa, "Puzzlefusion++: Auto-agglomerative 3d fracture assembly by denoise and verify," in *ICLR*, 2025.

[26] L. Zhu, S. Huang, and I. A. Konrad Schindler, "Living scenes: Multi-object relocalization and reconstruction in changing 3d environments," in *CVPR*, 2024.

[27] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the continuity of rotation representations in neural networks," in *CVPR*, 2019.

[28] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superglue: Learning feature matching with graph neural networks," in *CVPR*, 2020.

[29] S. Huang, Z. Gojcic, M. Usvyatsov, and K. S. Andreas Wieser, "Predator: Registration of 3d point clouds with low overlap," in *CVPR*, 2021.

[30] Z. J. Yew and G. H. Lee, "Rpm-net: Robust point matching using learned features," in *CVPR*, 2020.

[31] Y. Wang and J. M. Solomon, "Deep closest point: Learning representations for point cloud registration," in *ICCV*, 2019.

[32] Z. Qin, H. Yu, C. Wang, Y. Guo, Y. Peng, and K. Xu, "Geometric transformer for fast and robust point cloud registration," in *CVPR*, 2022.

[33] J. Y. Zhang, A. Lin, M. Kumar, T.-H. Yang, D. Ramanan, and S. Tulsiani, "Cameras as rays: Pose estimation via ray diffusion," in *ICLR*, 2024.

[34] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud, "Dust3r: Geometric 3d vision made easy," in *CVPR*, 2024.

[35] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, 1981.

[36] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Ep n p: An accurate o (n) solution to the p n p problem," *IJCV*, 2009.

[37] C. Choy, J. Park, and V. Koltun, "Fully convolutional geometric features," in *ICCV*, pp. 8958–8966, 2019.

[38] H. Deng, T. Birdal, and S. Ilic, "Ppfnet: Global context aware local features for robust 3d point matching," in *CVPR*, 2018.

[39] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser, "The perfect match: 3d point cloud matching with smoothed densities," in *CVPR*, pp. 5545–5554, 2019.

[40] X. Bai, Z. Luo, L. Zhou, H. Fu, L. Quan, and C.-L. Tai, "D3feat: Joint learning of dense detection and description of 3d local features," *arXiv:2003.03164 [cs.CV]*, 2020.

[41] H. Wang, Y. Liu, Z. Dong, and W. Wang, "You only hypothesize once: Point cloud registration with rotation-equivariant descriptors," in *ACM International Conference on Multimedia*, 2022.

[42] Y. Wang and J. M. Solomon, "Prnet: Self-supervised learning for partial-to-partial registration," *NeurIPS*, 2019.

[43] J. Huang, H. Wang, T. Birdal, M. Sung, F. Arrigoni, S. Hu, and L. J. Guibas, "Multibodysync: Multi-body segmentation and motion estimation via 3d scan synchronization," in *CVPR*, 2021.

[44] C. Deng, J. Lei, B. Shen, K. Daniilidis, and L. Guibas, "Banana: banach fixed-point network for pointcloud segmentation with inter-part equivariance," in *NeurIPS*, 2023.

[45] M. Atzmon, J. Huang, F. Williams, and O. Litany, "Approximately piecewise e(3) equivariant point networks," in *ICLR*, 2024.

[46] S. Huang, Z. Gojcic, J. Huang, A. Wieser, and K. Schindler, "Dynamic 3d scene analysis by point cloud accumulation," in *ECCV*, 2022.

[47] Y.-C. Chen, H. Li, D. Turpin, A. Jacobson, and A. Garg, "Neural shape mating: Self-supervised object assembly with adversarial shape priors," in *CVPR*, 2022.

[48] R. Wu, C. Tie, Y. Du, Y. Zhao, and H. Dong, "Leveraging se (3) equivariance for learning 3d geometric shape assembly," in *ICCV*, 2023.

[49] G. Scarpellini, S. Fiorini, F. Giuliari, P. Morerio, and A. Del Bue, "Diffassemble: A unified graph-diffusion model for 2d and 3d reassembly," *arXiv preprint arXiv:2402.19302*, 2024.

[50] Z. Wang, N. Xue, and R. Jörnsten, "Equivariant flow matching for point cloud assembly," *arXiv preprint arXiv:2505.21539*, 2025.

[51] X. Liu, C. Gong, and Q. Liu, "Flow straight and fast: Learning to generate and transfer data with rectified flow," *arXiv preprint arXiv:2209.03003*, 2022.

[52] Q. Liu, "Rectified flow: A marginal preserving approach to optimal transport, 2022," *URL https://arxiv.org/abs/2209.14577*.

[53] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," *arXiv preprint arXiv:2210.02747*, 2022.

[54] X. Wu, L. Jiang, P.-S. Wang, Z. Liu, X. Liu, Y. Qiao, W. Ouyang, T. He, and H. Zhao, "Point transformer v3: Simpler, faster, stronger," in *CVPR*, 2024.

[55] W. Peebles and S. Xie, "Scalable diffusion models with transformers," *arXiv preprint arXiv:2212.09748*, 2022.

[56] B. Zhang and R. Sennrich, "Root mean square layer normalization," *NeurIPS*, vol. 32, 2019.

[57] P. Esser, S. Kulal, A. Blattmann, R. Entezari, J. Müller, H. Saini, Y. Levi, D. Lorenz, A. Sauer, F. Boesel, *et al.*, "Scaling rectified flow transformers for high-resolution image synthesis," in *ICML*, 2024.

[58] S. Lee, Z. Lin, and G. Fanti, "Improving the training of rectified flows," vol. 37, pp. 63082–63109, 2024.

[59] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[60] J. Lu, Y. Sun, and Q. Huang, "Jigsaw: Learning to assemble multiple fractured objects," *NeurIPS*, 2023.

[61] Wikipedia contributors, "Rodrigues' rotation formula—Wikipedia, The Free Encyclopedia." `https://en.wikipedia.org/wiki/Rodrigues%27_rotation_formula`, 2025. [Online; accessed 11 May 2025].

[62] C. Kapfer, K. Stine, B. Narasimhan, C. Mentzel, and E. Candes, "Marlowe: Stanford's gpu-based computational instrument," Jan. 2025.

[63] J. Wang, M. Chen, N. Karaev, A. Vedaldi, C. Rupprecht, and D. Novotny, "Vggt: Visual geometry grounded transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[64] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[65] A. Narayan, R. Nagar, and S. Raman, "Rgl-net: A recurrent graph learning framework for progressive part assembly," in *WACV*, 2022.

[66] G. Zhan, Q. Fan, K. Mo, L. Shao, B. Chen, L. J. Guibas, H. Dong, *et al.*, "Generative 3d part assembly via dynamic graph learning," *NeurIPS*, 2020.

[67] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *NeurIPS*, 2020.

[68] C. Davis and W. M. Kahan, "The rotation of eigenvectors by a perturbation. iii," *SIAM Journal on Numerical Analysis*, vol. 7, no. 1, pp. 1–46, 1970.