Neural Shape Mating: Self-Supervised Object Assembly with Adversarial Shape Priors

Yun-Chun Chen[1,2] Haoda Li[1,2] Dylan Turpin[1,2] Alec Jacobson[1,4] Animesh Garg[1,2,3]
[1]University of Toronto [2]Vector Institute [3]NVIDIA [4]Adobe Research, Toronto

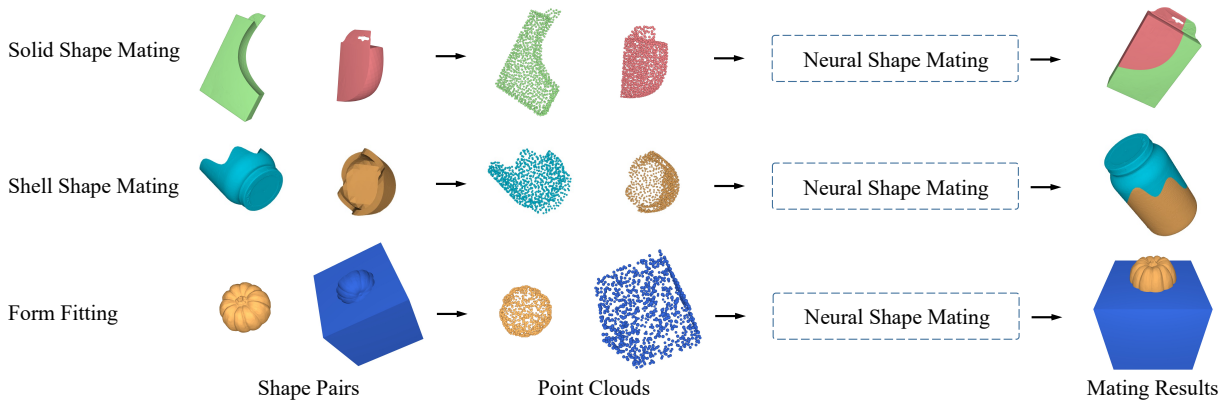

Figure 1. **Pairwise 3D geometric shape mating.** Neural Shape Mating (NSM) takes as input a pair of point clouds of the two shapes and predicts the mating configuration as output. NSM learns to mate shapes together with self-supervision and does not require semantic information or target shapes as guidance at test time. Our method can be applied to various shape mating settings, including solid shape mating (*top row*), shell shape mating (*middle row*), and form fitting (*bottom row*).

Abstract

Learning to autonomously assemble shapes is a crucial skill for many robotic applications. While the majority of existing part assembly methods focus on correctly posing semantic parts to recreate a whole object, we interpret assembly more literally: as mating geometric parts together to achieve a snug fit. By focusing on shape alignment rather than semantic cues, we can achieve across category generalization and scaling. In this paper, we introduce a novel task, pairwise 3D geometric shape mating, and propose Neural Shape Mating (NSM) to tackle this problem. Given point clouds of two object parts of an unknown category, NSM learns to reason about the fit of the two parts and predict a pair of 3D poses that tightly mate them together. In addition, we couple the training of NSM with an implicit shape reconstruction task, making NSM more robust to imperfect point cloud observations. To train NSM, we present a self-supervised data collection pipeline that generates pairwise shape mating data with ground truth by randomly cutting an object mesh into two parts, resulting in a dataset that consists of 200K shape mating pairs with numerous object meshes and diverse cut types. We train NSM on the collected dataset and compare it with several point cloud registration methods and one part assembly baseline approach. Extensive experimental results and ablation studies under various settings demonstrate the effectiveness of the proposed algorithm. Additional material is available at: neural-shape-mating.github.io.

1. Introduction

The human-built world is filled with objects that are shaped to fit, snap, connect, or mate together. Reassembling a broken object and inserting a plug into a socket are both instances of shape mating. This kind of geometric mating has many practical applications and appears in domains ranging from computer graphics [28], 3D design [6, 17], robotics [8, 46, 53, 54], and biology [41]. This paper introduces the problem of *pairwise* shape mating without known models, which is an integral subroutine in the broader problem of multi-part geometric assembly including applications in robotics [27] and AR/VR [42].

There have been many attempts that learn shape-to-shape matching algorithms in application-specific domains: furniture assembly [15, 27, 29], object assembly [1, 30], and

object packing [46]. Most of these assembly algorithms operate under the assumption that each shape corresponds to a recognizable semantic object part [15, 27, 29]. While these results are promising, they rely heavily on semantic information (e.g., part segmentation), target shapes [29] as guidance, and ground-truth part pose annotations [15, 29]. This reliance makes these methods application specific, hard to scale, and difficult to generalize.

In this paper, we consider shape mating from a geometric perspective, without relying on semantic information or prespecified target shapes as guidance. Specifically, we study the *pairwise* 3D geometric shape mating task, where shape mating is done based solely on geometric cues. To achieve this, we propose Neural Shape Mating (NSM). As shown in Figure 1, given a pair of shapes in the form of point clouds with random intial poses, NSM predicts a plausible mating configuration for them using geometric cues. The proposed task is challenging yet practical with mating being a subroutine in applications in robotics such as object kitting [8] and form fitting [53] and in biology such as protein binding [41] (where the binding between proteins requires reasoning about the geometric fit between two proteins).

We formulate the proposed task as a pose prediction problem and develop a Transfomer-based module [44] that takes as input the point clouds of the two shapes, reasons about the fit by attending to asymmetric correlations between local geometric cues, and predicts respective poses that bring them together. In addition, we adopt an adversarial learning scheme that learns shape priors for evaluating the plausibility of the generated shape mating configurations. Furthermore, to account for imperfect point cloud observations (e.g., noisy point clouds), we couple the training of NSM with an implicit shape reconstruction task [35, 40].

To train NSM, we present a self-supervised data collection pipeline that generates pairwise shape mating data with ground truth by randomly cutting an object mesh with different cut types into two parts. We collect object meshes from the Thingi10K [56], Google Scanned Objects [13], and ShapeNet [5] datasets and apply our data generation algorithm to each object mesh. The resulting geometric shape mating dataset covers a diverse set of cut types applied to numerous object instances of 11 categories, combining a total of 200K shape pairs suitable for evaluating the proposed task. We train NSM on the collected dataset in a self-supervised fashion and compare our method with several point cloud registration algorithms and one part assembly baseline approach. Extensive experimental results and ablation studies under various settings demonstrate the effectiveness of the proposed algorithm.

Summary of contributions:

1. We introduce a novel task of pairwise geometric shape mating and propose Neural Shape Mating that predicts mating configurations using geometric cues.
2. We collect a large-scale geometric shape mating dataset for evaluation.
3. We compare NSM with several point cloud registration methods and one part assembly baseline approach.
4. Experimental results and analysis support our design choices and demonstrate the robustness of NSM when presented with realistically noisy observations.

2. Related Work

3D shape assembly. A distinct, but related, line of work investigates generative models that learn to represent objects as concatenations of simple 3D shapes. [43] trains per-class models that generate objects by assembling volumetric primitives (cuboids). [23] trains a single model that can generate cuboid primitives across all classes. [21] models objects with ShapeAssembly programs, learned by a variational autoencoder (VAE) [25], which can be executed to generate 3D shapes as concatenations of cuboids. These methods provide powerful abstractions and reveal correspondences between objects by abstracting away details of local geometry. In contrast, we consider the problem of discovering plausible fits between shapes with complex geometry that do not correspond to any semantic part or natural object decomposition. The validity of a fit relies on the alignment of detailed local geometric features, which provide cues for shape mating.

The task that comes closest to our own is part assembly [14, 15, 20, 29, 50, 51], which aims at making a complete object from a set of parts. [29] learns to predict translations and rotations for point cloud parts to assemble a target object specified by an image. [14, 15] frames the part assembly problem as graph learning and learns to assemble parts into complete objects by iterative message passing. These methods use the PartNet dataset [34], and thus the parts to assemble are always a reasonable semantic decomposition of the target object. Shape is important in part assembly, but cues can also be taken from part semantics directly, bypassing the geometric cues. In contrast, we consider the problem of learning to fit together pieces with no particular semantics and without a provided target.

Pose estimation. Existing pose estimation methods predict poses for known objects by aligning a provided model with an observation [3, 55]. Other learning-based approaches predict poses for novel objects as bounding box corners [26] or semantic keypoints [45, 52] or mappings to a normalized coordinate space [47]. Rather than independently estimating the current pose of a single observed object, our method leverages cross-shape information to predict a new pose for each observed shape that assembles them into a whole object.

Learning shape priors. Our model includes an adversarial prior implemented by a discriminator that learns to distinguish between ground-truth assembled shape pairs and the predicted mating results. Conditional generative adversarial networks [12, 33] have achieved impressive results

on image generation tasks even when paired ground truth is unavailable, as in unpaired image-to-image translation [58], or when ground truth is available but multiple plausible outputs exist, as in MUNIT [16]. Even when the ground truth is available and a unimodal correct output exists, adversarial losses lead to enhanced detail and more realistic outputs, e.g., for super-resolution [31]. In our problem, we learn shape priors with an adversarial loss that encourages our model to generate plausible shape mating configurations.

Implicit shape reconstruction. A core problem in computer vision is reconstructing 3D shapes from 2D observations. Rather than representing the reconstructed shapes as finite sets of points, voxels, or meshes, a recent line of work aims to represent them as implicit functions parameterized by neural networks. These encode shapes by their signed distance functions (SDFs) [35, 40] or the indicator functions [32], which are continuous, differentiable and can be queried at arbitrary resolution. DeepSDF [35] learns SDFs for many shape classes with a feedforward neural network. Further work [9, 10] adds an additional structure to improve reconstruction accuracy and memory efficiency. We follow a similar approach to [19, 22, 59], which take inspiration from implicit reconstruction to improve performance on a pose prediction task. Specifically, as in [19, 59], we include implicit shape reconstruction as an auxiliary task and show, through ablation, that this improves performance on our main shape mating task, suggesting significant synergies between shape mating and shape reconstruction.

Point cloud registration. If we had access to additional information, our problem would reduce to point cloud registration [3, 4, 57]. Specifically, if we had a segmentation of the interface of each piece (the subset of its surface that contacts the other piece in the assembled pose), computing the assembled pose would reduce to aligning the paired interfaces. If we were given correspondences between these interfaces, alignment would become a well-characterized optimization problem solvable with Procrustes. Without correspondences, we would be left with a registration problem. Feature-free methods such as Iterative Closest Point (ICP) [3] approximate correspondences simply as pairs of closest points. Sparse ICP [4] improves robustness to noise by distinguishing between inliers and outliers. Learning-based methods seek to approximate correspondences in a learned feature space [7, 11, 48]. Unlike registration methods which aim to align two point clouds with (partial) overlap, our method is designed to predict paired poses that bring two *disjoint* shapes together to form a single whole object.

3. Problem Statement: Pairwise 3D Geometric Shape Mating

We formulate the pairwise 3D geometric shape mating task as a pose prediction problem. In this task, we assume we are given the point clouds P_A and P_B of the two shapes S_A and S_B, where $P_A = \{p_i^A\}_{i=1}^N$, $p_i^A \in \mathbb{R}^3$ is a point in P_A, $P_B = \{p_j^B\}_{j=1}^M$, $p_j^B \in \mathbb{R}^3$ is a point in P_B, and N and M denote the number of points in point clouds P_A and P_B, respectively. Shape S_A and shape S_B are the two parts of a whole object S. We aim to learn a model that takes as input the two point clouds P_A and P_B and predicts a canonical SE(3) pose $\{(R_k, T_k) \mid R_k \in \mathbb{R}^{3\times3}, T_k \in \mathbb{R}^3\}$ for each point cloud P_k, where R_k denotes the rotation matrix, T_k is the translation vector, and $k \in \{A, B\}$.[1] The predicted SE(3) poses will then be applied to transform each respective input point cloud. The union of the two transformed point clouds $\bigcup_{k\in\{A,B\}} R_k P_k + T_k$ forms the shape mating result.

4. Method: Neural Shape Mating

We describe Neural Shape Mating and the loss functions used to train our model in this section.

4.1. Algorithmic overview

Given two point clouds P_A and P_B, our goal is to learn a model that predicts an SE(3) pose for each input point cloud. We propose Neural Shape Mating, which comprises four components: 1) a point cloud encoder, 2) a pose estimation network, 3) an adversarial shape prior module, and 4) an implicit shape reconstruction network.

As shown in Figure 2, we first apply the point cloud encoder E to point clouds P_A and P_B to extract point features $f_A = E(P_A) = \{f_i^A\}_{i=1}^N$ and $f_B = E(P_B) = \{f_j^B\}_{j=1}^M$, respectively, and $f_i^A \in \mathbb{R}^d$ and $f_j^B \in \mathbb{R}^d$. Next, the point features f_A and f_B are passed to the pose estimation network to reason about the fit between the two point clouds P_A and P_B and predict SE(3) poses $\{R_k, T_k\}$ for $k \in \{A, B\}$ for them. The point features f_A and f_B are also passed to the SDF network F for learning implicit shape reconstruction. The predicted SE(3) poses are then applied to the respective input point clouds, and the union of the two transformed point clouds forms the mating result P_{pred}. To learn plausible shape mating configurations, we have a discriminator that takes as input the predicted mating configuration P_{pred} and the ground truth P_{GT} and distinguishes whether the mating configurations look visually realistic or not.

Point cloud encoder. There are several point cloud models such as PointNet [38], PointNet++ [39], and DGCNN [49] that are applicable for learning point features. In this work, we follow DCP [48] and adopt DGCNN as our point cloud encoder E, and the dimension d of the point features f_i^A and f_j^B is $1,024$ (i.e., $f_i^A \in \mathbb{R}^{1024}$ and $f_j^B \in \mathbb{R}^{1024}$). We refer the reader to [49] for more details of DGCNN.

Rotation representation. We follow prior work [29] and use quaternion to represent rotations.

[1] We perform canonical pose estimation to ensure that the model is symmetric. Once the canonical mating is estimated, a shape registration with a rigid transformation can be applied to align either shape with the other in the observed scene (see Figure 4). The canonical pose prediction formulation is also adopted by recent methods [15, 29].

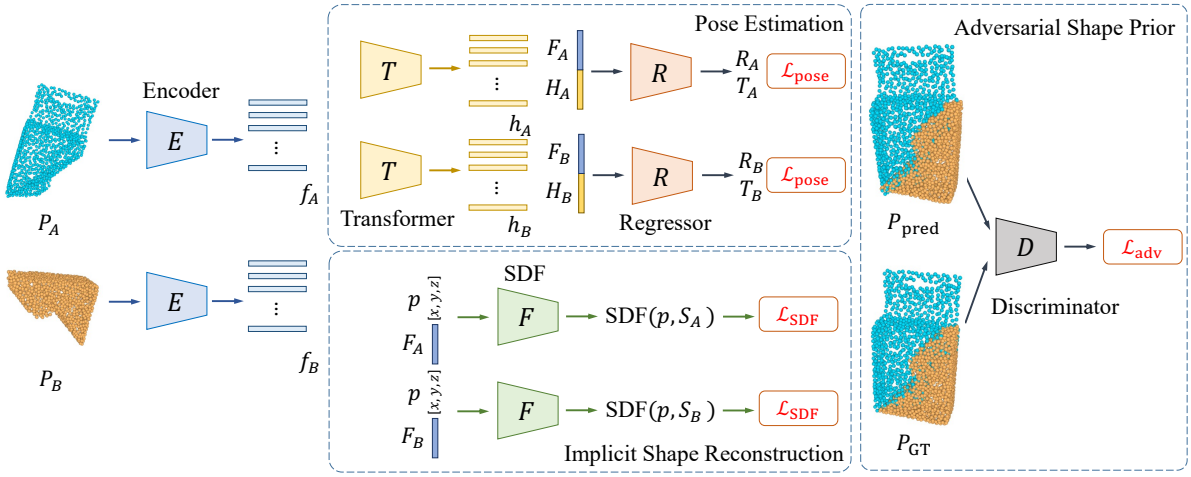

Figure 2. **Overview of Neural Shape Mating.** Neural Shape Mating is composed of four main components: a point cloud encoder E, a pose estimation module that consists of a Transformer network T and a regressor network R, an implicit shape reconstruction module that learns signed distance functions (SDFs), and a discriminator D for learning shape priors.

4.2. Pose estimation for shape assembly

To achieve shape mating, we predict an SE(3) pose for each input point cloud. Unlike existing object pose estimation methods [45, 52] that independently predict a pose for each object, our task requires reasoning about the fit between the two input point clouds for pose prediction. To achieve this, we have a feature correlation module T that captures cross-shape information for providing geometric cues and a regressor R for predicting poses.

We adopt a Transformer network [44] as our feature correlation module T, as it allows the model to learn asymmetric cross-shape information. Given the point features $f_A = \{f_i^A\}_{i=1}^N$ and $f_B = \{f_j^B\}_{j=1}^M$ as input, the feature correlation module T computes pairwise feature correlation between each point feature $f_i^A \in f_A$ and each point feature $f_j^B \in f_B$ to obtain feature $h_A = \{h_i^A\}_{i=1}^N$ for point cloud P_A and feature $h_B = \{h_j^B\}_{j=1}^M$ for point cloud P_B, where $h_i^A \in \mathbb{R}^d$ and $h_j^B \in \mathbb{R}^d$. The details of Transformer are provided in Appendix B.

To predict SE(3) poses, we aggregate all features h_i^A in h_A to obtain the feature $H_A \in \mathbb{R}^d$ and all features h_j^B in h_B to obtain the feature $H_B \in \mathbb{R}^d$. Similarly, we aggregate all point features f_i^A in f_A to obtain the feature $F_A \in \mathbb{R}^d$ and all point features f_j^B in f_B to obtain the feature $F_B \in \mathbb{R}^d$. We use max pooling for feature aggregation as in PointNet [38]. The features F_A and H_A are concatenated (resulting in a feature of dimension $2d$) and passed to the regressor R to predict a unit quaternion q_A (which can be converted to a rotation matrix R_A) and a translation vector T_A. The prediction of q_B (or R_B) and T_B can be similarly derived.[2]

To guide the learning of the pose estimation network, we have a pose prediction loss $\mathcal{L}_{\text{pose}}$, which is defined as

$$\mathcal{L}_{\text{pose}} = \sum_{k\in\{A,B\}} \|R_k^\top R_k^{\text{GT}} - I\| + \|T_k - T_k^{\text{GT}}\|, \quad (1)$$

where R_k^{GT} and T_k^{GT} denote the ground-truth rotation and translation, respectively, and I is the identity matrix.

4.3. Adversarial learning of shape priors

To encourage NSM to predict plausible mating results, we propose to learn global shape priors to further constrain the prediction space. We exploit the idea that when the two point clouds are mated together using the predicted poses, the mating configuration should look visually realistic like an object. We cast this as an adversarial learning problem and introduce a discriminator D that takes as input the predicted mating result $P_{\text{pred}} = \bigcup_{k\in\{A,B\}} R_k P_k + T_k$ and the ground-truth mating configuration $P_{\text{GT}} = \bigcup_{k\in\{A,B\}} R_k^{\text{GT}} P_k + T_k^{\text{GT}}$ and distinguishes whether the input mating configurations look visually realistic like an object or not.

To achieve this, we have a loss $\mathcal{L}_{\text{G}}$ for training the generator (i.e., the point cloud encoder and the pose prediction network), which is defined as

$$\mathcal{L}_G = \mathbb{E}[\|D(P_{\text{pred}}) - 1\|], \quad (2)$$

and an adversarial loss $\mathcal{L}_{\text{adv}}$ for training the discriminator D, which is defined as

$$\mathcal{L}_{\text{adv}} = \mathbb{E}[\|D(P_{\text{pred}})\|] + \mathbb{E}[\|D(P_{\text{GT}}) - 1\|]. \quad (3)$$

Having this adversarial training allows NSM to predict poses that result in plausible mating results. The details of adversarial learning are provided in Appendix C.

4.4. Implicit shape reconstruction

Since the same object can be described by different point clouds and the point cloud capturing can be imperfect, to

[2] We normalize the predicted quaternion q_k by its length so that $\|q_k\|_2 = 1$ for $k \in \{A, B\}$.

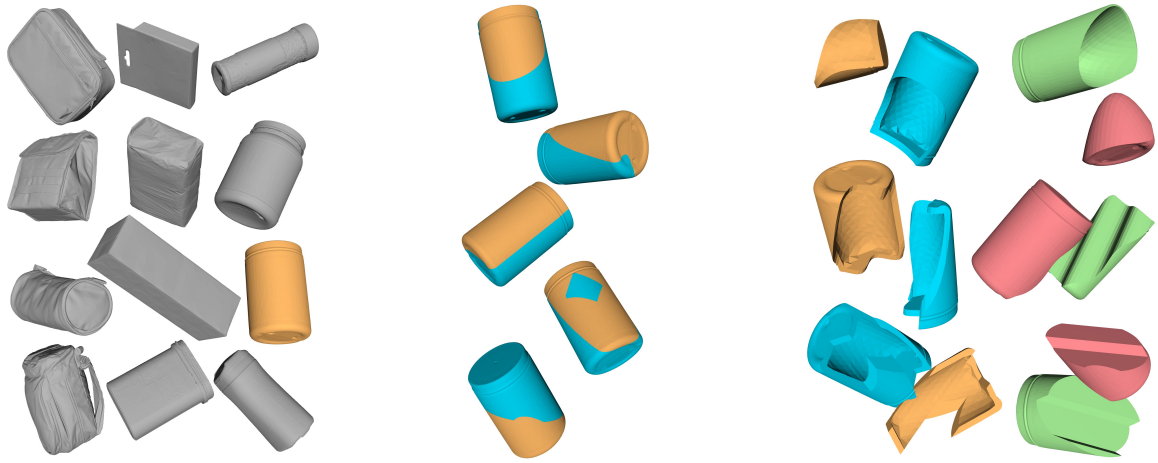

Figure 3. **Dataset overview.** (*Left*) Our dataset is composed of object meshes from 11 categories. (*Middle*) We define five different types of cut functions. Each object mesh can then be cut with many different ways using varying parametric cut functions. (*Right*) Finally, each pair of parts can be randomized with an initial SE(3) pose. In our dataset, we also generate solid and shell variations of each shape, when cutting a mesh to create different mating interfaces for the same problem instance.

account for the noise in point cloud sampling, we couple the learning of NSM with an implicit shape reconstruction task. This is motivated by recent advances in implicit shape modeling [32, 36], where learning SDFs allows the model to learn more robust shape representations. Specifically, we have an SDF network F that takes as input the aggregated features F_A and F_B, respectively, and a point $p \in \mathbb{R}^3$ in the 3D space, and predicts the signed distances between point p and shape S_A and between point p and shape S_B.

To train the SDF network, we have an SDF regression loss, which is defined as

$$\mathcal{L}_{\text{SDF}} = \sum_{k \in \{A,B\}} \|\text{SDF}(p, S_k) - \text{SDF}_{\text{GT}}(p, S_k)\|, \quad (4)$$

where $\text{SDF}(p, S_k)$ and $\text{SDF}_{\text{GT}}(p, S_k)$ denote the predicted and the ground-truth signed distances between the point p and the shape S_k, respectively.

5. The Geometric Shape Mating Dataset

To train NSM, we present a self-supervised method that generates pairwise geometric shape mating data with ground truth by randomly cutting an object mesh into two parts.

Mesh cutting. We normalize each object mesh by the longest bounding box length such that the object mesh has a maximum bounding box length of 1 and the aspect ratio remains the same. To cut the object, we use the mesh boolean functions provided by `libigl` [18]. We construct a height field that will be used to intersect the object mesh for mesh cutting. The height field can be parameterized by different functions. In our work, we define five different types of functions, including a planar function, a sine function, a parabolic function, a square function, and a pulse function. Each of these functions will result in a type of cut. We generate two types of shapes when performing cutting: the solid shape and the shell shape. To generate solid shape data, we use the height field to intersect with each object mesh. To generate shell shape data, we first construct an offset surface at the -0.05 level set of an object. We then use the height field to intersect with the difference between the original object mesh and the generated offset surface. We set the cut-off ratio to be no less than 25% (each object part mesh has a volume at least 25% of the uncut mesh). Figure 3 presents example data generated by applying different types of cuts. More visual examples are provided in Appendix E.

Point cloud sampling. We uniformly sample 1,024 points on each object part mesh ($N = 1,024$ and $M = 1,024$).

Signed distance ground truth. We use the Fast Winding Numbers method [2] for computing ground-truth signed distances. For each part mesh, we sample 40,000 points that are close to the mesh surface for learning the SDF network.

Pose transformation. Each point cloud is mean centered. During training, we sample two rotation matrices on the fly and apply them to transform the poses of the two input point clouds, respectively.

Statistics. We use 11 shape categories: `bag`, `bowl`, `box`, `hat`, `jar`, `mug`, `plate`, `shoe`, `sofa`, `toy`, and `table` in initial dataset version due to computational reasons. We note that the procedural data generation can be extended naively to other shape categories. The object meshes are collected from the Thingi10K [56], Google Scanned Objects [13], and ShapeNet [5] datasets. The dataset statistics are provided in Appendix D.

Table 1. **Experimental results of geometric shape mating.** R and T denote Rotation and Translation, respectively. Lower is better on all metrics. It is worth noting that many methods can get reasonably close in position, but be completely off in orientation as demonstrated by the RMSE error in rotation. NSM outperforms the best baseline in predicting the correct orientation by up to $4\times$ in MAE.

Method	Shape Matching Type	MSE (R)	RMSE (R)	MAE (R)	MSE (T)	RMSE (T)	MAE (T)
Solid Shape Mating					($\times 10^{-3}$)		
ICP (point-to-point) [3]	Local	9108.79	95.44	83.31	211.76	460.18	381.03
ICP (point-to-plane) [3]	Local	6748.62	82.15	74.31	82.15	286.61	203.17
Sparse ICP (point-to-point) [4]	Local	4751.34	68.93	65.44	33.89	184.09	152.63
Sparse ICP (point-to-plane) [4]	Local	3267.27	57.16	53.27	39.30	198.23	178.35
DCP [48]	Global	3400.06	58.31	56.17	55.26	235.08	227.41
GNN Assembly [29]	Global	1087.68	32.98	30.42	19.23	138.67	125.08
Neural Shape Mating (NSM)	Global	94.67	9.73	7.61	15.48	124.40	110.44
Shell Shape Mating							
ICP (point-to-point) [3]	Local	8725.43	93.41	89.07	608.81	780.26	747.32
ICP (point-to-plane) [3]	Local	6696.15	81.83	79.63	463.88	681.09	658.13
Sparse ICP (point-to-point) [4]	Local	5099.39	71.41	69.93	395.74	629.08	601.44
Sparse ICP (point-to-plane) [4]	Local	3517.68	59.31	56.62	327.91	572.63	556.74
DCP [48]	Global	3861.38	62.14	59.03	336.53	580.11	653.02
GNN Assembly [29]	Global	1662.19	40.77	38.49	113.69	337.18	320.01
Neural Shape Mating (NSM)	Global	290.02	17.03	14.52	107.60	328.03	301.19

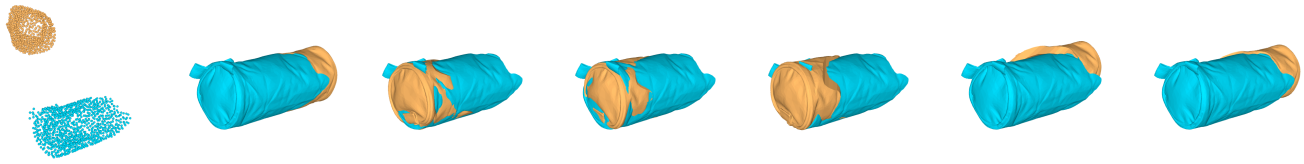

Figure 4. **Qualitative pairwise shape mating.** NSM predicts poses to accurately mate the two shapes together to make a `bag`. ICP, Sparse ICP and DCP methods estimate a pose for the yellow point cloud that aligns with the blue one. Both GNN Assembly and NSM (our method) predict poses for both the yellow and the blue point clouds. For visualization purposes, we convert the predicted canonical poses for both point clouds to the pose that aligns the yellow point cloud to the blue one for both GNN Assembly and NSM.

6. Experiments

We perform evaluations and analysis of NSM to answer the following questions:

1. How well does NSM perform when compared to point cloud registration methods and the graph network-based assembly baseline approaches?
2. Can NSM generalize to unseen object categories and unseen cut types?
3. How does NSM perform when presented with more realistic, noisy point clouds?
4. How much do the adversarial, reconstruction and pose losses contribute to final performance?

6.1. Experimental setup

Evaluation metrics. We follow the evaluation scheme from DCP [48]. We compute the mean squared error (MSE), root mean squared error (RMSE), and mean absolute error (MAE) between the predicted rotation and translation values and the ground truth values. The unit of rotation is degree.

Baselines. We compare our model with several point cloud registration methods: ICP [3], Sparse ICP [4] and DCP [48] as well as a graph-based part assembly approach adapted from [29], denoted as GNN Assembly. The three registration methods are all correspondence-based. That is, they approximate correspondences between point clouds and then find poses that minimize an energy based on those correspondences. ICP estimates correspondences as closest points and proceeds to iterate between updating poses (from the latest correspondences) and updating correspondences (from the latest poses). Since ICP weighs all correspondences equally, it can be thrown off by a few bad points. Sparse ICP improves robustness to noise by downweighting outliers. We include two variants of ICP and Sparse ICP, one computing distances point-to-point and the other point-to-plane (using ground-truth normals). DCP is a learning-based method, which learns to compute correspondences from which a final pose is generated with SVD. GNN Assembly is another learning-based method, but predicts rotations and translations with a message passing algorithm without correspondences (see Section 2 for more details). In each experiment, DCP, GNN Assembly, and NSM (our method) are all trained on the same training data.

Implementation details. We implement NSM using PyTorch [37]. We use the ADAM [24] optimizer for training. The learning rate is set to 1×10^{-3} with a learning rate decay of 1×10^{-6}. We train NSM for 5,000 epochs using four NVIDIA P100 GPUs with 12GB memory each. The network

Table 2. **Experimental results of model generalization.** (a) Results of unseen categories geometric shape mating. The test set contains shape pairs from the box and bag category. The training set contains shape pairs from the remaining 9 categories. (b) Results of unseen cut types geometric shape mating. The training set contains the planar, sine, square and pulse cut types. The test set contains the parabolic cut type. Results reported are in the solid shape mating setting.

(a) Unseen categories geometric shape mating.

Method	MSE (R)	RMSE (R)	MAE (R)	MSE (T)	RMSE (T)	MAE (T)
Solid Shape Mating				($\times 10^{-3}$)		
DCP [48]	6567.48	81.04	78.92	108.67	329.65	305.44
GNN Assembly [29]	2413.76	49.13	46.52	55.30	235.16	214.96
Neural Shape Mating (NSM)	266.34	16.32	13.74	55.22	234.98	212.57

(b) Unseen cut types geometric shape mating.

Method	MSE (R)	RMSE (R)	MAE (R)	MSE (T)	RMSE (T)	MAE (T)
Solid Shape Mating				($\times 10^{-3}$)		
DCP [48]	5905.92	76.85	74.02	88.90	298.16	274.32
GNN Assembly [29]	2143.69	46.30	43.91	58.33	241.51	220.49
Neural Shape Mating (NSM)	251.54	15.86	13.46	53.34	230.96	207.58

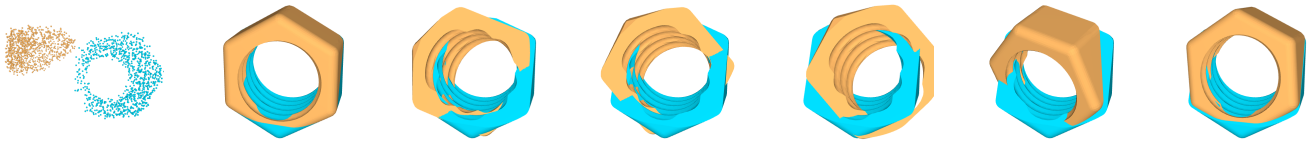

Figure 5. **Visual results of noisy point cloud pairwise geometric shape mating.** The two part meshes are obtained by applying a planar cut to a `nut` mesh. Gaussian noise with mean 0.0 and standard deviation 0.05 is added to each point. When noisy point clouds are presented, our method is still able to predict plausible mating configurations.

details are provided in Appendix A. We use the Open3D implementation for ICP. The implementations of Sparse ICP and DCP are from their official GitHub repository. We use the codebase from [29] for GNN Assembly and remove the part segmentation network branch. We evaluate all methods on both solid and shell shape mating settings.

6.2. Performance evaluation and comparisons

6.2.1 Comparison to existing approaches

We compare the performance of our method with existing approaches on pairwise 3D geometric shape mating. In this evaluation, we use 80% of the shape pairs for training, 10% for validation and 10% for testing (metrics are reported on this holdout set). Table 1 presents results for both solid and shell shape mating settings. Figure 4 presents a visual comparison between methods. More results are provided in Appendix F.

Quantiatively, results in both settings follow a similar pattern. NSM achieves the best rotation MSE by an order of magnitude. For translation prediction, NSM and GNN Assembly both achieve strong results.

Point cloud registration methods. NSM outperforms registration methods by a large margin in rotation prediction. This may be surprising as shape mating and point cloud registration are similar problems. In fact, shape mating reduces to point cloud alignment given an interface segmentation. Despite this, these results suggest that existing point cloud registration methods are insufficient for the shape mating task. In our qualitative results, we can see registration methods often attempt to overlay pieces rather than mating them together and this matches our hypothesis that the inferior performance of registration methods is due to their correspondence assumptions. In point cloud registration, it is assumed that the inputs correspond usually to a rigid transformation and some observation noise. Even with outlier handling, they are unable to leave the non-interface portion of the surface out of correspondence in order to precisely align the interface portions. More surprisingly, this may be true even for learning-based methods like DCP, where the interpolation of correspondences may force consideration of non-interface points. These results highlight that shape mating is a distinct problem from registration, requiring more specialized method design.

Part assembly. NSM outperforms GNN Assembly on rotation prediction and performs similarly on translation prediction. The GNN Assembly architecture is designed for the part assembly task where semantic cues are available and fine-grained geometric details are not as important for alignment. We hypothesize that our adversarial loss for learning shape priors and the Transformer architecture for capturing cross-shape information are better suited to the *geometric* shape mating task which relies on these details.

These results support our conviction that the proposed

Table 3. **Ablation study on Neural Shape Mating model design choices.** Testing performance with each loss removed. The training and test sets remain the same as in the main experiment (as presented in Table 1).

Method	MSE (R)	RMSE (R)	MAE (R)	MSE (T)	RMSE (T)	MAE (T)
Solid Shape Mating				($\times 10^{-3}$)		
NSM	94.67	9.73	7.61	15.48	124.40	110.44
NSM w/o $\mathcal{L}_{\text{adv}}$ and $\mathcal{L}_{\text{G}}$	324.36	18.01	15.04	22.76	150.87	135.41
NSM w/o $\mathcal{L}_{\text{SDF}}$	185.78	13.63	11.42	20.18	142.06	128.90
NSM w/o $\mathcal{L}_{\text{pose}}$	7826.94	88.47	85.09	710.75	843.06	776.43
Shell Shape Mating						
NSM	290.02	17.03	14.52	107.60	328.03	301.19
NSM w/o $\mathcal{L}_{\text{adv}}$ and $\mathcal{L}_{\text{G}}$	446.90	21.14	18.63	133.83	365.83	341.07
NSM w/o $\mathcal{L}_{\text{SDF}}$	418.61	20.46	17.44	111.74	334.27	311.07
NSM w/o $\mathcal{L}_{\text{pose}}$	9374.11	96.82	93.02	593.90	770.65	689.44

task is distinct from point cloud registration and part assembly, and that progress will require further investigation into the *geometric* shape mating problem specifically.

6.2.2 Generalization to unseen categories and cut types

Unseen categories. To test the generalization across categories, we test on the `box` and `bag` categories and train on the remaining 9 categories. Table 2a presents results on the solid shape mating setting and the results on the shell shape mating setting are provided in Appendix G. Notably, NSM is category agnostic, and relies mainly on aligning surface geometry details than class-specific semantic cues, we expected strong generalization. Compared to in-category testing, while the performance degrades slightly, NSM still performs favorably against existing methods.

Unseen cut types. To test the generalization across different cut types, we test on the parabolic cuts and train on the remaining 4 cut types. Table 2b presents results on the solid shape mating setting and the results on the shell shape mating setting are provided in Appendix H. As with unseen cut types, the performance degrades for all methods, while NSM still achieves the best results.

6.2.3 Evaluation on noisy point clouds

Real-world point cloud data, e.g., captured by depth cameras, contains measurement error that the point clouds in our training set do not. For our framework to be applicable to real-world problems, it must be robust to noise in the point cloud observations. To test robustness to noise, we train and test each model on a noise-augmented version of our dataset. Gaussian noise with mean 0.0 and standard deviation 0.05 is added to each point. As can be seen in Figure 5, while the performance of all methods, including ours, does decline, NSM is still able to predict reasonable mating poses. Quantitative results are provided in Appendix I. The performance of correspondence-based methods (ICP, Sparse ICP, and even learning-based DCP) all show large drops in performance.

6.2.4 Ablation study: Contribution of loss functions

To evaluate our design choices, we conduct an ablation study by removing a loss function at a time. Table 3 presents the results for both solid and shell shape mating settings. Visual results are provided in Appendix J. The training and test sets remain the same as in the main experiment (as presented in Table 1). Performance declines significantly without adversarial learning (i.e., without $\mathcal{L}_{\text{adv}}$ and $\mathcal{L}_{\text{G}}$), confirming our hypothesis that adversarial learning can serve as a pose refinement or regularizer and improve predictions even when ground truth is available. Performance also declines without learning implicit shape reconstruction (i.e., without $\mathcal{L}_{\text{SDF}}$), suggesting that there are useful synergies between shape mating and geometry reconstruction. Without the pose loss $\mathcal{L}_{\text{pose}}$, the model does not learn shape mating at all, which suggests adversarial training with implicit shape reconstruction alone is not sufficient.

7. Conclusions

This paper introduces a new problem with broad applications, an insightful procedural data generation method, and an algorithmic architecture designed for the proposed task. The self-supervised data generation pipeline allows NSM to learn shape mating without ground truth. Since NSM learns to align geometric features rather than semantic ones, it is able to generalize across categories and across surface cut types. Experimental results show that NSM predicts plausible mating configurations and outperforms all competing methods. An ablation study suggests that the novel adversarial training scheme significantly improves performance (even though ground truth is available) and the performance benefits of an auxiliary implicit shape reconstruction task suggest synergies between shape reconstruction and shape mating. We hope that this paper can convincingly establish geometric shape mating as a meaningful task, distinct from semantic part assembly. Pairwise geometric shape mating is a core task to solve multi-step reasoning required for assembling parts to form an object. Natural extensions of NSM would go beyond pairwise shape mating to consider the problem of mating N-parts.

Acknowledgments. We thank Ziyi Wu for providing feedback to early draft. Alec Jacobson is supported by Canada Research Chairs Program and gifts by Adobe Systems. Animesh Garg is supported by CIFAR AI Chair, NSERC Discovery Award, University of Toronto XSeed award, and gifts from LG.

References

[1] Maneesh Agrawala, Doantam Phan, Julie Heiser, John Haymaker, Jeff Klingner, Pat Hanrahan, and Barbara Tversky. Designing effective step-by-step assembly instructions. *ACM TOG*, 2003. 1

[2] Gavin Barill, Neil G Dickson, Ryan Schmidt, David IW Levin, and Alec Jacobson. Fast winding numbers for soups and clouds. *ACM TOG*, 2018. 5

[3] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. *TPAMI*, 1992. 2, 3, 6

[4] Sofien Bouaziz, Andrea Tagliasacchi, and Mark Pauly. Sparse iterative closest point. *Computer Graphics Forum*, 2013. 3, 6

[5] 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*, 2015. 2, 5

[6] Xuelin Chen, Hao Zhang, Jinjie Lin, Ruizhen Hu, Lin Lu, Qi-Xing Huang, Bedrich Benes, Daniel Cohen-Or, and Baoquan Chen. Dapper: decompose-and-pack for 3d printing. *ACM TOG*, 2015. 1

[7] Haowen Deng, Tolga Birdal, and Slobodan Ilic. Ppfnet: Global context aware local features for robust 3d point matching. In *CVPR*, 2018. 3

[8] Shivin Devgon, Jeffrey Ichnowski, Michael Danielczuk, Daniel S Brown, Ashwin Balakrishna, Shirin Joshi, Eduardo MC Rocha, Eugen Solowjow, and Ken Goldberg. Kitnet: Self-supervised learning to kit novel 3d objects into novel 3d cavities. In *CASE*, 2021. 1, 2

[9] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. Local deep implicit functions for 3d shape. In *CVPR*, 2020. 3

[10] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. Learning shape templates with structured implicit functions. In *ICCV*, 2019. 3

[11] Zan Gojcic, Caifa Zhou, Jan D Wegner, and Andreas Wieser. The perfect match: 3d point cloud matching with smoothed densities. In *CVPR*, 2019. 3

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 2

[13] GoogleResearch. Google scanned objects, 2021. 2, 5

[14] Abhinav Narayan Harish, Rajendra Nagar, and Shanmuganathan Raman. Rgl-net: A recurrent graph learning framework for progressive part assembly. In *WACV*, 2022. 2

[15] Jialei Huang, Guanqi Zhan, Qingnan Fan, Kaichun Mo, Lin Shao, Baoquan Chen, Leonidas Guibas, and Hao Dong. Generative 3d part assembly via dynamic graph learning. In *NeurIPS*, 2020. 1, 2, 3

[16] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *ECCV*, 2018. 3

[17] Alec Jacobson. Generalized matryoshka: Computational design of nesting objects. *Computer Graphics Forum*, 2017. 1

[18] Alec Jacobson, Daniele Panozzo, et al. libigl: A simple C++ geometry processing library, 2018. 5

[19] Zhenyu Jiang, Yifeng Zhu, Maxwell Svetlik, Kuan Fang, and Yuke Zhu. Synergies between affordance and geometry: 6-dof grasp detection via implicit representations. In *RSS*, 2021. 3

[20] Benjamin Jones, Dalton Hildreth, Duowen Chen, Ilya Baran, Vladimir G Kim, and Adriana Schulz. Automate: A dataset and learning approach for automatic mating of cad assemblies. *ACM TOG*, 2021. 2

[21] R Kenny Jones, Theresa Barton, Xianghao Xu, Kai Wang, Ellen Jiang, Paul Guerrero, Niloy J Mitra, and Daniel Ritchie. Shapeassembly: Learning to generate programs for 3d shape structure synthesis. *ACM TOG*, 2020. 2

[22] Korrawe Karunratanakul, Jinlong Yang, Yan Zhang, Michael Black, Krikamol Muandet, and Siyu Tang. Grasping field: Learning implicit representations for human grasps. In *3DV*, 2020. 3

[23] Salman H Khan, Yulan Guo, Munawar Hayat, and Nick Barnes. Unsupervised primitive discovery for improved 3d generative modeling. In *CVPR*, 2019. 2

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2014. 6

[25] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014. 2

[26] Hei Law, Yun Teng, Olga Russakovsky, and Jia Deng. Cornernet-lite: Efficient keypoint based object detection. In *BMVC*, 2020. 2

[27] Youngwoon Lee, Edward S Hu, Zhengyu Yang, Alex Yin, and Joseph J Lim. Ikea furniture assembly environment for long-horizon complex manipulation tasks. In *ICRA*, 2021. 1, 2

[28] Honghua Li, Ibraheem Alhashim, Hao Zhang, Ariel Shamir, and Daniel Cohen-Or. Stackabilization. *ACM TOG*, 2012. 1

[29] Yichen Li, Kaichun Mo, Lin Shao, Minhyuk Sung, and Leonidas Guibas. Learning 3d part assembly from a single image. In *ECCV*, 2020. 1, 2, 3, 6, 7

[30] Or Litany, Emanuele Rodolà, Alexander M Bronstein, Michael M Bronstein, and Daniel Cremers. Non-rigid puzzles. *Computer Graphics Forum*, 2016. 1

[31] Alice Lucas, Santiago Lopez-Tapia, Rafael Molina, and Aggelos K Katsaggelos. Generative adversarial networks and perceptual losses for video super-resolution. *TIP*, 2019. 3

[32] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*, 2019. 3, 5

[33] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv*, 2014. 2

[34] Kaichun Mo, Shilin Zhu, Angel X. Chang, Li Yi, Subarna Tripathi, Leonidas J. Guibas, and Hao Su. PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding. In *CVPR*, 2019. 2

[35] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. 2, 3

[36] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. 5

[37] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming

Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 6

[38] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 3, 4

[39] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017. 3

[40] Vincent Sitzmann, Eric R Chan, Richard Tucker, Noah Snavely, and Gordon Wetzstein. Metasdf: Meta-learning signed distance functions. In *NeurIPS*, 2020. 2, 3

[41] Freyr Sverrisson, Jean Feydy, Bruno E Correia, and Michael M Bronstein. Fast end-to-end learning on protein surfaces. In *CVPR*, 2021. 1, 2

[42] Anna Syberfeldt, Oscar Danielsson, Magnus Holm, and Lihui Wang. Visual assembling guidance using augmented reality. *Procedia Manufacturing*, 2015. 1

[43] Shubham Tulsiani, Hao Su, Leonidas J. Guibas, Alexei A. Efros, and Jitendra Malik. Learning shape abstractions by assembling volumetric primitives. In *CVPR*, 2017. 2

[44] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2, 4

[45] Chen Wang, Roberto Martín-Martín, Danfei Xu, Jun Lv, Cewu Lu, Li Fei-Fei, Silvio Savarese, and Yuke Zhu. 6-pack: Category-level 6d pose tracker with anchor-based keypoints. In *ICRA*, 2020. 2, 4

[46] Fan Wang and Kris Hauser. Stable bin packing of non-convex 3d objects with a robot manipulator. In *ICRA*, 2019. 1, 2

[47] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *CVPR*, 2019. 2

[48] Yue Wang and Justin M Solomon. Deep closest point: Learning representations for point cloud registration. In *ICCV*, 2019. 3, 6, 7

[49] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM TOG*, 2019. 3

[50] Karl DD Willis, Pradeep Kumar Jayaraman, Hang Chu, Yunsheng Tian, Yifei Li, Daniele Grandi, Aditya Sanghi, Linh Tran, Joseph G Lambourne, Armando Solar-Lezama, et al. Joinable: Learning bottom-up assembly of parametric cad joints. In *CVPR*, 2022. 2

[51] Kangxue Yin, Zhiqin Chen, Siddhartha Chaudhuri, Matthew Fisher, Vladimir G Kim, and Hao Zhang. Coalesce: Component assembly by learning to synthesize connections. In *3DV*, 2020. 2

[52] Yang You, Yujing Lou, Chengkun Li, Zhoujun Cheng, Liangwei Li, Lizhuang Ma, Cewu Lu, and Weiming Wang. Keypointnet: A large-scale 3d keypoint dataset aggregated from numerous human annotations. In *CVPR*, 2020. 2, 4

[53] Kevin Zakka, Andy Zeng, Johnny Lee, and Shuran Song. Form2fit: Learning shape priors for generalizable assembly from disassembly. In *ICRA*, 2020. 1, 2

[54] Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, et al. Transporter networks: Rearranging the visual world for robotic manipulation. In *CoRL*, 2020. 1

[55] Andy Zeng, Kuan-Ting Yu, Shuran Song, Daniel Suo, Ed Walker, Alberto Rodriguez, and Jianxiong Xiao. Multi-view self-supervised deep learning for 6d pose estimation in the amazon picking challenge. In *ICRA*, 2017. 2

[56] Qingnan Zhou and Alec Jacobson. Thingi10k: A dataset of 10,000 3d-printing models. In *SGP*, 2021. 2, 5

[57] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Fast global registration. In *ECCV*, 2016. 3

[58] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 3

[59] Minghan Zhu, Maani Ghaffari, and Huei Peng. Correspondence-free point cloud registration with so (3)-equivariant implicit shape representations. In *CoRL*, 2022. 3