Learning to Synthesizes 3D Shapes with Kinematic Joints

Anonymous ECCV submission

Paper ID 7120

Abstract. A man-made object consists of parts that are stitched together using various joints, including rigid joints, revolute joints, and prismatic joints. While parts describe the overall geometric shape, joints encode the relative motions among different parts. They largely dictate the semantics and functions of the underlying object. This paper introduces a first deep generative model that takes a latent code as input and outputs a 3D part-based model with predicted joints between adjacent parts. Our approach introduces a novel graph-based representation for encoding shape details and joint-part configurations. We show how to train a shape generator by combining losses on vertex attributes and induced edge attributes. We then show how to enhance the quality of the generator by developing self-supervision losses that promote consistency among predicted attributes and the underlying kinematic structure. Experimental results show that our approach generates high-quality results across various shape categories. We demonstrate the usefulness of our approaches in applications of joint-based shape parsing and constrained shape synthesis.

Keywords: numerical optimization, shape analysis, planar hexagonal mesh

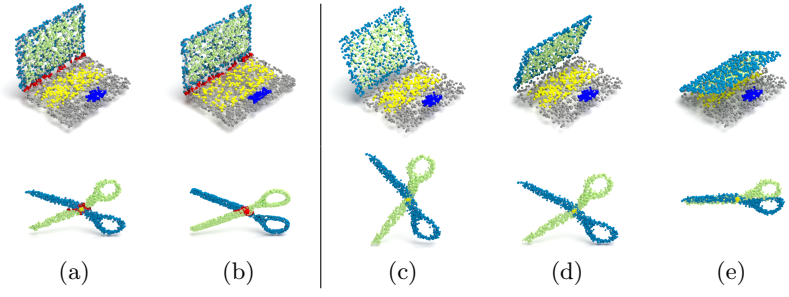

Fig. 1: (Left). Generated functional 3D shapes (a) with part-joint configurations. Red edges represent joints. Each shape is accompanied with the closest shape (b) in the training dataset. (Right) The learned generative models enable novel applications in shape interpolation (c-e).

1 Introduction

Part-based models form a popular geometric representation for 3D shapes [29, 45]. Shape parts convey rich semantic information and provide a concise way to compose and synthesize new shapes. Most man-made objects, ranging from furniture models to electronics to architectural models, naturally decompose into geometrically and semantically meaningful parts. In the deep learning era, there are great successes in developing deep neural networks to synthesize part-based 3D models [5].

However, existing 3D shape synthesis approaches have predominantly focused on static attributes of part-based representations, such as part shapes [9, 43, 49], part relations [23, 52, 51, 56], and symmetric structures among parts [24, 30]. They do not consider kinematic joints between adjacent parts that convey critical semantic information, physical properties, and dynamics, which offer many applications. For example, in 3D fabrication several works studied converting 3D models into articulated objects that can interact with physical environments [2, 6, 10, 22, 26, 27, 37, 38, 55]. Joint-based representations offer flexible means to synthesize and edit 3D shapes (c.f. [47]). They are also important for applications in human-object interactions and robot-object interactions. Therefore, it is critical to develop synthesis approaches that output both 3D shapes and kinematic joint-part configurations.

In this paper, we introduce a deep parametric generator that outputs 3D shapes with kinematic joint configurations (See Figure 1(Left)). Our approach possesses all advantages of existing shape generators, such as encoding geometric shape details and modeling shape parts. The coupled joint-part configurations enable numerous downstream applications such as parsing a 3D point cloud into a joint-part representation and constrained shape modeling (See Figure 1(Right)).

Central to our approach is a novel graph representation for encoding part shapes and joint-part configurations. Most existing part-based synthesis approaches decouple a shape into part structures (i.e., content) and part geometric shapes (i.e., style) [9, 24, 49]. However, it is challenging to model kinematic joints that impose complex data-dependent constraints among pairs of parts under this representation. In contrast, our graph representation offers a concise way to encode all desired properties. Graph vertices represent a detailed point cloud that models shape details. Edge attributes model part boundaries and joint configurations. A key advantage of this encoding is that it promotes multi-task learning to improve predictions of individual attributes.

Besides learning generative models under the proposed graph representation, we also develop self-supervised losses to enhance generalization. For example, there are underlying cluster structures among the predicted attributes. We show how to employ robust objective functions to promote such cluster structures. As another example, the predicted part-joint configurations of generated shapes offer kinematic deformations. Through the lens of modal analysis [11, 15, 34], such deformations provide effective regularizations on the structure of generated shapes.

We have tested our approach on seven categories of the PartNet dataset [33]. Experimental results show that our approach can synthesize 3D shapes with both detailed geometry and plausible joints. The resulting joints are both geometrically meaningful and physically plausible. An ablation study justifies the usefulness of each component of our approach.

In summary, we present the following contributions:

- A new synthesis task that couples kinematic joints, parts, and shape details.
- A graph representation that models kinematic joint-part configurations and enables geometric regularizations.
- Applications in parsing a 3D point cloud into a 3D shape with joints and shape deformation.

2 Related Works

We decompose the related works into two groups, i.e., part-based shape synthesis and joint modeling for 3D shapes.

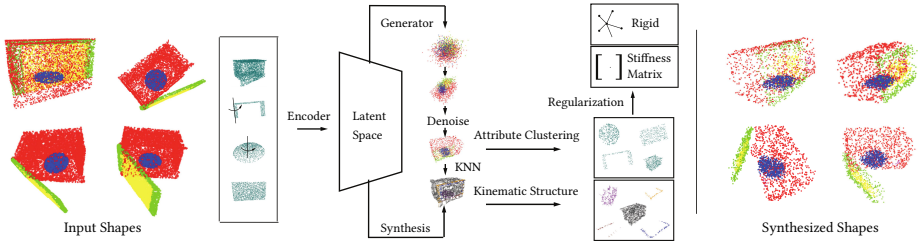

Fig. 2: Overview of our approach. We use a graph representation of shapes with kinematic part-joint configurations. We train an auto-encoder and a latent GAN to synthesize new shapes. The total training loss combines four terms, including two data terms and two regularization terms.

2.1 Part-Based Shape Synthesis

Part-based shape synthesis [4, 8, 19, 21, 46] has been studied extensively in the literature. We refer to [29, 45] for two standard surveys on non-deep learning based techniques. This paper focuses on neural synthesis techniques. A recent survey is [5].

Many neural part-based shape synthesis approaches use recurrent formulations. Different approaches differ with respect to the shape representation and the neural modules. Li et al. [24] build on shape parse trees to synthesize part structures. Zou et al. [56] introduced a recurrent network to augment parts for shape synthesis. Mo et al. [30] use a two-level encoder-decoder to synthesize part structure and part geometry by combining graph neural networks and

point-based representations. Wu et al. [42] perform sequence to sequence modeling for part structure synthesis. Jones et al. [18] use part-based shape programs for shape synthesis. Another work [9] uses two-level decoupled variational auto-encoders to synthesize part structures and part geometries. Yang et al. [49] introduced a novel approach that decouples latent codes of part structure and part geometry. In contrast to these decoupled formulations, we use a unified representation. Different attributes are put under a multi-task learning framework.

In contrast to focusing on parts, several recent works emphasize the relations between adjacent parts. Specifically, Zhan et al. [52] models part relations as a dynamic graph for assembling primitive shapes. Yin et al. [51] studied the geometry of rigid joints. In contrast, this paper emphasizes kinematic joints. Our representation models both part information and detailed shape geometries.

Wang et al. [40] introduced a voxel-based part-representation. It first generates coarse voxel-based part geometry, which is then augmented with shape details. In contrast, our approach uses a unified graph representation to encode both geometric details and part information. It combines the strength of graph representation learning and multi-task learning.

Several recent works have developed various regularization losses for structure-aware shape synthesis. Mezghanni et al. [28] introduced two regularization losses to enhance the topology and physical stability of synthesizing man-made shapes. Koch et al. [20] introduced geometric regularization losses to enforce physical stability. However, the approach only applies to limited shape primitives. Huang et al. [16] introduced an as-rigid-as possible loss to regularize the Jacobian of shape generators. While our approach shares the concept of developing geometric regularizations, we focus on two novel regularizations for joint-aware shape synthesis.

2.2 Joint Modeling for 3D Shapes

Xu et al. [47] pioneered the area of joint-based 3D shape manipulation. More recent works centered around joint modeling fall into two categories. People focus on algorithms for detecting kinematic and functional joints from 3D geometry in one direction. Along this line, Hu et al. [13] introduced an approach to detect mobility patterns of 3D parts for a given 3D shape. Lin et al. [26] introduced an approach for recovering functional parts from a collection of scans of a physical object. Mo et al. [31] introduced a deep learning approach to predict actions of articulated objects that consist of movable parts. In contrast to these approaches, we study learning a generative model of 3D shapes with kinematic joints. Besides predicting joints of existing shapes, our approach can synthesize new shapes with kinematic joints.

In another direction, people have studied converting 3D shapes into articulated robots with functional joints. Lau et al. [22] studied the problem of taking a 3D model of a man-made object as input and automatically generating the parts and connectors needed to build the corresponding physical object. Zhu et al. [55] introduced a method to synthesize mechanical toys solely from the motion of their features. Coros et al. [6] developed a system that allows non-

expert users to create animated mechanical characters, focusing on optimized mechanism that minimizes the gap between virtual designs and physical outputs. Tomaszewski et al. [38] optimizes linkage-based characters by generating a design space that allows users to browse different topology options interactively. Our approach is complementary to this line of works by learning a design space of 3D shapes with functional joints.

3 Problem Statement and Overview

3.1 Problem Statement

The input to our approach is a collection of 3D models $\mathcal{S} = \{S_1, \cdots, S_n\} \subset \overline{\mathcal{S}}$ where $\overline{\mathcal{S}}$ denotes the space of 3D models. Each model S_i has the annotated parts. Joints are associated with pairs of adjacent parts. As illustrated in Figure 3, a joint can be a rigid joint that is not movable or a kinematic joint that is movable (e.g., Revolute and Prismatic c.f. [47]). Under this setup, our goal is to train a shape generator $\boldsymbol{g} : \mathcal{Z} := \mathbb{R}^d \rightarrow \overline{\mathcal{S}}$ that takes a latent code $\boldsymbol{z}$ as input and outputs new shapes with both geometric details and predicted part-joint configurations.

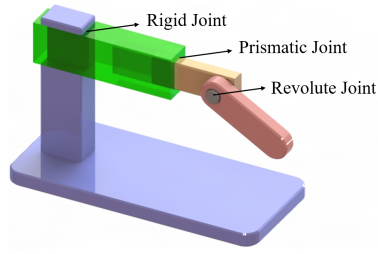

Fig. 3: Illustrations of different joint types (rigid, revolute, and prismatic).

3.2 Approach Overview

Similar to most 3D synthesis tasks, the technical challenge of synthesizing shapes with part-joint configurations is to determine a suitable 3D representation (c.f. [5]). We propose to develop a graph representation. Graph vertices represent a point cloud that encodes detailed geometric shapes; vertex attributes encode predictions of part index and parameters of the closest joint. Edge attributes are predicted based on vertex attributes, i.e. an edge being intra-part or inter-part (e.g., rigid and revolve joints) can be inferred from vertex attributes. Note that we do not assume parts are labeled consistently among the input shapes. Similar to [7, 39, 54], the optimal order emerges during training.

The proposed network is trained by combining data terms for learning generative models and geometric regularization terms (See Figure 2). Our point generator adopts ShapeGF [3], a recent point-based generative model that preserves shape details. A novelty of our approach is to impose a loss defined on graph edges. This approach uses annotations of edge attributes to train the point generator. Similar to multi-task learning, the hybrid loss terms enable us to learn better feature representations for shape generation.

While the data terms consider training instances, the regularization terms operate on synthetic instances. Specifically, the first regularization term prioritizes

that vertex attributes form clusters. The second regularization term considers a stiffness matrix derived from the graph representation. For objects with kinematic joints, the corresponding infinitesimal kinematic motions are eigenvectors of the stiffness matrix whose corresponding eigenvalues are approximately zero. We formulate the second regularization term to minimize eigenvalues of stiffness matrices of synthetic shapes to promote kinematic structures. We introduce an effective multi-stage training approach to minimize the total training loss.

4 Approach

This section presents the technical details of our approach. We begin with the shape representation in Section 4.1. We then introduce the learning formulation in Section 4.2. Finally, we present the training procedure in Section 4.3.

4.1 Shape Representation

Our main representation is a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The vertex set $\mathcal{V}$ ($|\mathcal{V}| = m$ and $m = 2048$) represents an un-ordered set of points (See Fig. 4). Each vertex v_j has a position $\boldsymbol{p}_j$, a part indicator vector $\boldsymbol{a}_j^{\mathcal{P}}$, and a joint indicator vector $\boldsymbol{a}_j^{\mathcal{J}}$. $\boldsymbol{a}_j^{\mathcal{P}}$ is a binary vector that specifies the association between v_j with each part. Its length is the maximum number of parts of a shape in the training set. $\boldsymbol{a}_j^{\mathcal{J}}$, represented as rotation and translation vector, encodes the joint type the vertex associates to.

The edge set $\mathcal{E}$ is introduced to construct a stiffness matrix to analyze the kinematic structure of a generated shape. It also provides a representation for learning the shape generator, emphasizing inter-part edges that are critical for joint-part structures. Motivated from DGCNN [41], we begin with connecting nearest neighbors among the current point cloud (We used 15 nearest neighbors in our experiments). On each generated shape, we first adopt the clustering method on part indicator vectors to assign each vertex v_j a part label l_j. Then, we assign each edge $e = (i, j) \in \mathcal{E}(\boldsymbol{z})$ an inter-part attribute $\boldsymbol{l}_e = h(\{l_i, l_j\})$. h is a hash function such that $\{l_i, l_j\} = \{l_j, l_i\}$ and $\{l_i, l_j\} \neq \{l_i, l_k\}$ if and only if $j \neq k$. (Note that $\{l_i, l_j\} = \{l_j, l_i\}$, since $\{l_i, l_j\}$ represents the inter-part edge between part l_i and l_j.)

The induced inter-part edge attributes will be used to construct the stiffness matrix for defining the geometric regularization term (Section 4.2).

4.2 Training Objectives

We adopt ShapeGF [3], which combines a score-based denoising decoder and a latent GAN to sample the latent distribution of the training instances (See Section 4.2).

Our total training loss consists of three terms. The first term align predicted vertex attributes (Section 4.2), and the second term promotes the underlying

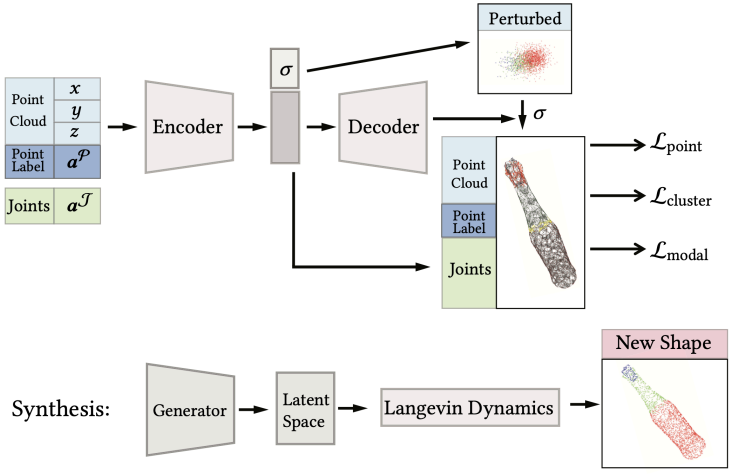

Fig. 4: Illustration of the network architecture of our approach. We combine four loss terms to train an auto-encoder. A latent GAN is trained to capture the latent distribution, which is then used for synthesizing new shapes.

cluster structure of part attributes (Section 4.2). The third terms are the global regularization on the underlying kinematic deformation structure, emphasizing the joint-part configuration (Section 4.2). Below we elaborate the technical details.

Point-Based Data Term Let vector $\boldsymbol{v}_i^j, j \in [|\mathcal{V}| = m]$ collect the point position and point attributes on the j-th vertex of the i-th training instance S_i. With $\boldsymbol{g}^\theta(\boldsymbol{z})$ we denote the output of the point cloud generator with latent parameter $\boldsymbol{z}$. In this paper, we adopt the recent point generation model ShapeGF [3] to define $\boldsymbol{g}(\boldsymbol{z})$, due to its ability in preserving shape details. Specifically, our goal is to train $\boldsymbol{g}^\theta(\boldsymbol{z})$ together with an encoder $\boldsymbol{h}^\phi$ [3] by minimizing the following loss:

$$\mathcal{L}_{\text{point}} = \frac{1}{n}\sum_{i=1}^{n}\Big(\frac{1}{m}\sum_{j=1}^{m}\|(\boldsymbol{g}^\theta \circ \boldsymbol{h}^\phi)(\tilde{\boldsymbol{v}}_i^j, \sigma, S_i) - \frac{\boldsymbol{v}_i^j - \tilde{\boldsymbol{v}}_i^j}{\sigma^2}\|_2^2\Big), \tag{1}$$

where

$$\tilde{\boldsymbol{v}}_i^j \sim \mathcal{N}(\boldsymbol{v}_i^j, \sigma^2 I).$$

Specifically, (1) estimates the gradient fields of the log density of a smoothed distribution that approximates the empirical distribution of $\{\boldsymbol{v}_i^j, i \in [m]\}$. Hyper-parameter σ controls the noise level of Gaussian Kernel to smooth the discrete and discontinuous distribution over $\{\boldsymbol{v}_i^j, i \in [m]\}$. The estimated gradient fields allow generating novel shape with multiple attributes via Langevin dynamics directly. The same as [3], we train a latent GAN that learns the distribution of the training instances in the latent space, which we can further sample a new latent code from.

Attribute Clustering Term

We first consider an attribute clustering term to penalize that the differences between predictions of part attributes and joint attributes among adjacent ver-

tices. The motivation is that these predictions form clusters, and differences only occur among few edges across different clusters. In the same spirit as robust optimization for clustering [36], we use the following regularization term to promote the underlying cluster structure:

$$\mathcal{L}_{\text{cluster}} = \mathop{\mathbb{E}}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}} \sum_{e=(i,j)\in\mathcal{E}} f_r\Big(\big(\boldsymbol{a}_i^{\mathcal{P}}(\boldsymbol{z}); \boldsymbol{a}_i^{\mathcal{J}}(\boldsymbol{z})\big) - \big(\boldsymbol{a}_j^{\mathcal{P}}(\boldsymbol{z}); \boldsymbol{a}_j^{\mathcal{J}}(\boldsymbol{z})\big)\Big) \tag{2}$$

where $\big(\boldsymbol{a}_i^{\mathcal{P}}(\boldsymbol{z}); \boldsymbol{a}_i^{\mathcal{J}}(\boldsymbol{z})\big)$ is the column vector that concatenates part attributes $\boldsymbol{a}_i^{\mathcal{P}}(\boldsymbol{z})$ and joint attributes $\boldsymbol{a}_i^{\mathcal{J}}(\boldsymbol{z})$. $f_r(\boldsymbol{x}) = \sigma^2/(\sigma^2+\|\boldsymbol{x}\|^2)$ is the Geman-McClure robust function ($\sigma = 0.05$ in this paper). $p_{\boldsymbol{z}}$ is the latent distribution specified by the latent GAN [3].

The data terms consider training instances. The promise is that with implicit regularizations of neural network training, the network weights capture patterns among the training data, which are then used to generate new instances. We propose to enhance generative modeling by formulating priors on generated shapes as regularization terms. These regularization terms guide the generator to learn generalizable feature patterns, leading to improved generators. Specifically, this paper develops regularization terms for enhancing part-joint structures.

Modal Analysis Regularization Term

In contrast to the clustering regularization term that focuses on local properties of output shapes, the third regularization term prioritizes that output shapes possess part-joint configurations. To this end, we employ modal analysis [11, 15, 34, 53] that connects spectral properties of the stiffness matrix with properties of infinitesimal deformations that reveal geometric and physical structures of the underlying shape.

Specifically, we introduce a new stiffness matrix $H(\boldsymbol{z}) \in \mathbb{R}^{3m\times 3m}$ for each shape. $H(\boldsymbol{z})$ satisfies the property that the first few non-trivial eigenvectors[1] of $H(\boldsymbol{z})$ correspond to infinitesimal deformations of the underlying part-joint configurations, and such configurations are exact if and only if the corresponding eigenvalues are zero. Given $H(\boldsymbol{z})$, we define the second regularization term as

$$\mathcal{L}_{\text{modal}} = \mathop{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}} \sum_{k=7}^{L} \lambda_k^{\alpha}\big(H(\boldsymbol{z})\big) \tag{3}$$

In our experiments, we set $\alpha = 0.5$ and $L - 6$ as the maximum number of total degrees of freedom of a training shape. The robust norm λ_k^{α} addresses generated shapes with fewer degrees of freedom.

Motivated from [15, 50], we propose to study a variational formulation to define $H(\boldsymbol{z})$. Specifically, let $\boldsymbol{u} \in \mathbb{R}^{3m}$ encode the infinitesimal deformation where $\boldsymbol{u}_i \in \mathbb{R}^3$ represents the velocity associated with the i-th vertex. For each edge $e = (i, j) \in \mathcal{E}(\boldsymbol{z})$, we define a residual vector as

$$\boldsymbol{r}_e(\boldsymbol{z}) := \boldsymbol{c}_i \times \big(\boldsymbol{p}_i(\boldsymbol{z}) - \boldsymbol{p}_j(\boldsymbol{z})\big) - (\boldsymbol{u}_i - \boldsymbol{u}_j) - A_e(\boldsymbol{z})\boldsymbol{s}_{\boldsymbol{l}_e} \tag{4}$$

[1] The six eigenvectors correspond to rigid motions (c.f. [15])

where $\boldsymbol{c}_i$ is the latent variable that encodes the local rigid transformation in $\boldsymbol{u}$; $\boldsymbol{p}_i(\boldsymbol{z})$ is the position of vertex v_i on the graph-represented shape decoded from latent z; $\boldsymbol{s}_{\boldsymbol{l}_e}$ is the latent variable that collects edges of the same inter-part label $\boldsymbol{l}_e$; $A_e(\boldsymbol{z})$ is the linear term of joint parameters $\boldsymbol{a}_i^{\mathcal{J}}$ and $\boldsymbol{a}_j^{\mathcal{J}}$ on the edge $e = (i, j)$. For example, if the decoded joint parameters of vertex v_i and v_j from latent z are $\boldsymbol{a}_i^{\mathcal{J}} = (\boldsymbol{c}_i, \boldsymbol{o}_i)$ and $\boldsymbol{a}_j^{\mathcal{J}} = (\boldsymbol{c}_j, \boldsymbol{o}_j)$ respectively, i.e., v_i rotates around $\boldsymbol{o}_j$ following rotation vector $\boldsymbol{c}_j$, $A_e(\boldsymbol{z}) = (\boldsymbol{c}_i \times \boldsymbol{o}_i - \boldsymbol{c}_j \times \boldsymbol{o}_j) - (\boldsymbol{c}_i \times -\boldsymbol{c}_j\times)\boldsymbol{p}_j$. Note that the intuition of this form comes directly from: with $\boldsymbol{u}_i = \boldsymbol{c}_i \times (\boldsymbol{p}_i - \boldsymbol{o}_i)$ being the local rigid transformation of v_i and $\boldsymbol{u}_j = \boldsymbol{c}_j \times (\boldsymbol{p}_j - \boldsymbol{o}_j)$ being the local rigid transformation of v_j,

$$\boldsymbol{u}_i - \boldsymbol{u}_j \tag{5}$$

$$= \boldsymbol{c}_i \times \big(\boldsymbol{p}_i(\boldsymbol{z}) - \boldsymbol{p}_j(\boldsymbol{z})\big) - (\boldsymbol{u}_i - \boldsymbol{u}_j) - ((\boldsymbol{c}_i \times \boldsymbol{o}_i - \boldsymbol{c}_j \times \boldsymbol{o}_j) - (\boldsymbol{c}_i \times -\boldsymbol{c}_j\times)\boldsymbol{p}_j) \tag{6}$$

$$= \boldsymbol{c}_i \times \big(\boldsymbol{p}_i(\boldsymbol{z}) - \boldsymbol{p}_j(\boldsymbol{z})\big) - (\boldsymbol{u}_i - \boldsymbol{u}_j) - A_e(\boldsymbol{z}). \tag{7}$$

We can see that $A_e(\boldsymbol{z})$ vanishes when $\boldsymbol{u}_i$ and $\boldsymbol{u}_j$ follows the same rigid transformation, in other words, in the same part. Therefore, $A_e(\boldsymbol{z})\boldsymbol{s}_{\boldsymbol{l}_e}$ encodes additional degrees of freedom specified by the joint associated with e and vanishes for fixed joints. With this setup, we define $H(\boldsymbol{z})$ so that

$$\boldsymbol{u}^T H(\boldsymbol{z})\boldsymbol{u} := \min_{\{\boldsymbol{c}_i\},\{\boldsymbol{s}_{\boldsymbol{l}_e}\}} \sum_{e=(i,j)\in\mathcal{E}(\boldsymbol{z})} \|\boldsymbol{r}_e(\boldsymbol{z})\|^2. \tag{8}$$

Similar to [15], $H(\boldsymbol{z})$ admits a simple explicit expression and we defer the details and the proof to the supp. material.

4.3 Training Details

Combining (1), (2), and (3), we arrive at the following training loss for learning the encoder $\boldsymbol{h}^\phi$ and the decoder $\boldsymbol{g}^\theta$:

$$\min_{\boldsymbol{g}^\theta, \boldsymbol{h}^\phi} \quad \mathcal{L}_{\text{point}} + \lambda_1 \mathcal{L}_{\text{cluster}} + \lambda_2 \mathcal{L}_{\text{modal}} \tag{9}$$

where we set $\lambda_1 = 0.25$, $\lambda_2 = 1$, and $\lambda_3 = 1$ in this paper.

We perform network training at three stages. The first stage drops the regularization terms and use $\mathcal{L}_{\text{point}} + \lambda_1 \mathcal{L}_{\text{cluster}}$ to initialize the network weights. The second stage imposes the local regularization term, ensuring that the stiffness matrix is well-conditioned. The third stage fine-tunes the generators by minimizing (9).

5 Experimental Results

This section presents an experimental evaluation. Section 5.1 begins with the experimental setup. Section 5.2 analyzes the results. Section 5.3 presents an ablation study. Finally, Section 5.4 shows applications.

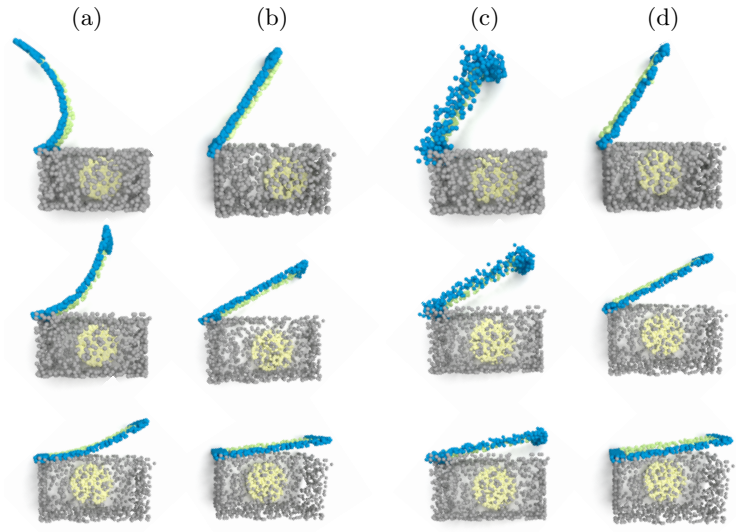

Fig. 5: Illustration of geometric regularization of our approach on Microwave. (a) unfeasible motion. (b) after regularization. (c) unfeasible motion. (d) after regularization. The geometric regularization term $\mathcal{L}_{\text{modal}}$ helps preserve the shape of parts and offers more plausible deformations.

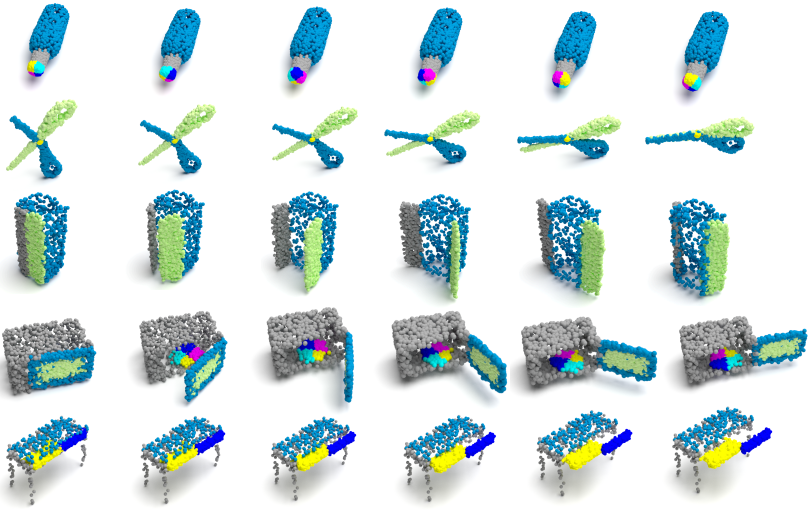

Fig. 6: We show a random set of generated shapes from each category. For the cap of Bottle and the tray of Microwave, we color the corresponding points differently to illustrate the underlying revolute motions. From top to bottom: Bottle, Scissors, Refrigerator, Microwave, and Table.

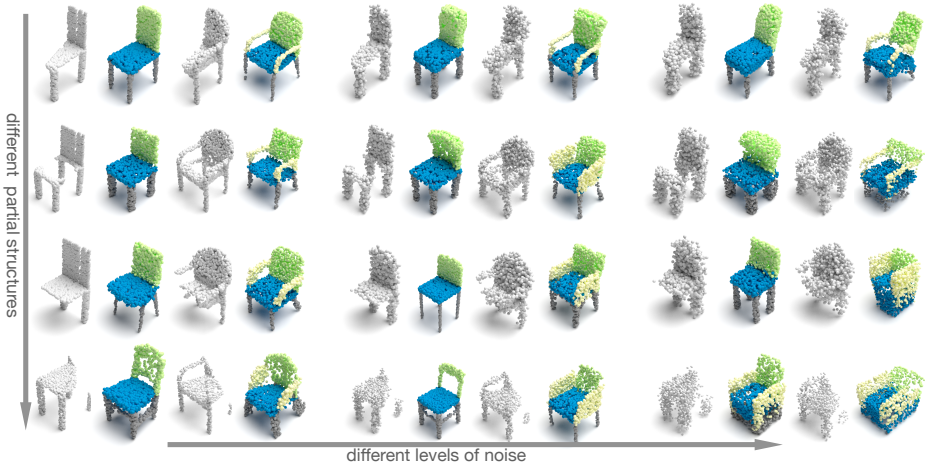

Fig. 7: Functional shape parsing results under different levels of inputs. Our approach can complete and clean the inputs and predict meaningful joint configurations.

5.1 Experimental Setup

Architecture Following ShapeGF setup, we build up the encoder using a 4-layer PointNet [35] style network and a 5-layer ResNet [12] style network for the decoder.

Datasets Although the SAPIEN dataset [44] has already provided the 3D shapes with motion annotations, we find that each class only contains around 50 shapes on average. To ensure the architecture to capture enough training samples for generating shapes, we extend a new dataset based on the PartNet dataset [33]. We train a model for each of the seven categories respectively: Bottle, Laptop, Microwave, Scissors, Refrigerator, Table and Chair. The point clouds are uniformly sampled from the mesh surface of each part. The part label and joint axis label are manually annotated according to the hierarchical annotation of PartNet. In our experiments, only revolute and prismatic joints are considered, which are sufficient to cover all kinematic joints in these categories. For each shape, we sample $10K$ points with coupled part and joint label, and downsample 2048 points during training.

Evaluation protocols For evaluation, we use Coverage (COV) and Minimum Matching Distance (MMD) following the convention of prior works [1, 3, 25, 48]. COV measures the percentage of point clouds in the training dataset that can be matched as the closest shape by at least one generated point cloud. Both Chamfer Distance (CD) and Earth Mover's Distance (EMD) can be used to measure the closeness, thus yielding COV-CD and COV-EMD. MMD measures the distance between the point clouds in the training dataset with their nearest neighbors in the generated shape collection. Similarly, both MMD-CD and MMD-EMD are used.

5.2 Analysis of Results

		Bot.	Lap.	Mic.	Sci.	Ref.	Tab.	Cha.
COV ↑ (CD)	S.O	43.34	50.34	38.50	33.65	38.50	41.13	46.21
	S+P	43.80	49.42	42.39	34.72	38.82	41.44	45.91
	Ours	**49.98**	**51.64**	**44.56**	**36.08**	**40.10**	**44.13**	**47.14**
COV ↑ (EMD)	S.O	47.01	47.57	47.28	36.63	40.10	46.22	46.13
	S+P	48.16	45.03	**52.17**	35.41	37.72	47.48	46.10
	Ours	**48.62**	**48.57**	44.02	**37.12**	**41.17**	**49.24**	**47.26**
MMD ↓ (CD)	S.O	11.03	10.73	23.57	25.74	**17.77**	**28.21**	40.14
	S+P	10.93	**11.57**	23.32	**25.86**	20.30	34.06	39.21
	Ours	**9.47**	11.80	**22.25**	25.88	18.12	31.15	**36.52**
MMD ↓ (EMD)	S.O	7.29	10.03	**15.99**	17.45	13.62	20.53	14.36
	S+P	7.14	11.10	17.37	18.02	13.71	20.66	14.21
	Ours	**6.34**	**9.78**	16.57	**16.91**	**12.98**	**19.87**	**13.72**

Table 1: Evaluation results on our extended PartNet. ↑ means the higher the better. ↓ means the lower the better. S.O: shape-only. S+P: shape + part. MMD-CD is multiplied by 10^4. MMD-EMD is multiplied by 10^2.

Figure 1 and Figure 11 show qualitative results of our approach. The generated shapes exhibit geometric details, and the overall shapes are plausible. Moreover, the generated shapes are different from the closest training shapes, meaning the generator utilizes the important visual features to synthesize new shapes. Moreover, the predicted parts align with the underlying geometric shape. The predicted joints are semantically meaningful with respect to shape parts.

Table 1 presents a quantitative assessment of our approach. We consider two baseline approaches for experimental evaluation. The first approach (S.O) simply generates raw point clouds without part and joint predictions. The second approach (S+P) predicts part annotations but without joint predictions. Table 1 shows that all metrics improve with additional predictions of parts and joints. This is expected as annotations of parts and joints bring additional information that is correlated with shape geometry. Moreover, our approach leads to more noticeable improvements than merely synthesizing parts than the performance gap between synthesizing points and part attributes and synthesizing points. Besides the additional signals from the joint supervision, such improvements come from the clustering regularization term and geometric regularizations derived from analyzing the stiffness matrix.

5.3 Ablation Study

This section presents an ablation study of the proposed approach.

Without part-encoding clustering Our approach combines two regularization losses. The first one performs part-encoding clustering under a robust norm.

As shown in Figure 8(Left), the part boundaries become fuzzy when dropping the clustering term. An explanation is that the regularization loss is critical for enforcing compatibility among adjacent edge predictions, providing clean inputs for extracting part-joint configurations.

Without geometric regularizations Figure 8(Right) and Figure 5 show the effects of dropping the geometric regularization term in the training and post-processing steps respectively. Visually, the effects are more salient. Some shape parts are distorted when dropping the geometric regularization term. This performance gap justifies the importance of the geometric regularization term, which extracts kinematic motions to regularize variations among generated shapes.

GAN based model as backbones In our work, we use ShapeGF as the backbone. We show here that when changing the backbone to GAN based model, we can consistently get improvement when involving our methods. We choose PDGN [17] as GAN base and evaluate both of them on Bottle class. We see in Table 2, our method consistently improves generative quality across different backbones.

BackBone		COV(CD)	COV(EMD)	MMD(CD)	MMD(EMD)
PDGN	S.O.	40.17	45.14	14.96	11.54
	S+P	41.30	46.76	13.01	10.76
	Ours	**42.07**	**47.32**	**12.45**	**10.78**

Table 2: Compare with different backbones. S.O: shape-only. S+P: shape + part. MMD-CD is multiplied by 10^4. MMD-EMD is multiplied by 10^2.

5.4 Applications

We proceed to describe two applications of the learned functional shape generator.

Joint-based shape synthesis The first application is to use the learned joints to synthesize new 3D shapes as shown in Figure 1(Right). Compared to sampling new shapes using the learned generator, this approach allows us to explore the sub-space defined by kinematic motions of a shape in the shape space defined by the generator. The outputs are semantically plausible.

Function-based shape parsing and completion The second application uses the learned generator to parse and then complete a partial shape that does not have part-joint configurations. Given a partial shape S and the learned generator $g^{\theta^\star}(\boldsymbol{z})$, we seek to optimize the latent code

$$\boldsymbol{z}^\star = \underset{\boldsymbol{z}}{\operatorname{argmin}}\ d^2_{g^{\theta^\star}(\boldsymbol{z})}(S)$$

where $d_{g^{\theta^\star}(\boldsymbol{z})}$ is the distance field of $g^{\theta^\star}(\boldsymbol{z})$. We select the chair class, which contains the most number of samples, as the example to conduct the visualization. Figure 7 shows the parsing results from a full spectrum of inputs, including partial shapes under different levels of completeness and noise. Our approach can complete and clean the inputs and predict meaningful functional parts and joints. More examples in other categories are available in supp. material.

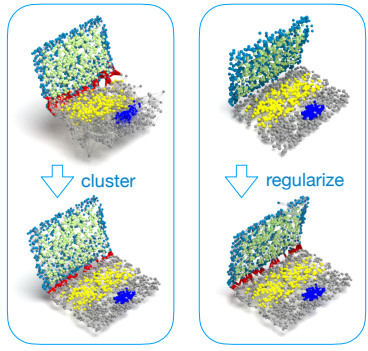

Fig. 8: (Left) Predictions of joint configurations and shape details become less accurate when dropping the part-encoding clustering term. (Right) The geometric regularization term is critical for ensuring the generalzability of shape details on generated shapes.

5.5 Connection with two-stage synthesize method

Another baseline is conducting the two-stage shape synthesize to get the shape with motion. The two-step synthesis method firstly generate the 3D shapes using a generative model and then using an end-to-end pretrained network to predict joint and motion information based on the 3D shape input [14, 32]. Comparing with the two-stage synthesis method, our pipeline is more flexible and can further improve the generative quality in shape itself. Due to lack of the metric to evaluate the generated motions, we conduct a user study comparing the two-stage synthesis method and our method, please refer to supp. material.

6 Conclusions, Limitations, and Future Work

This paper introduces the problem of synthesizing functional objects with part-joint configurations. We demonstrate the effectiveness of a graph-based shape representation. We also show the importance of developing geometric regularization based on kinematic motions derived from part-joint configurations. Benchmark evaluations and applications demonstrate the usefulness of our approach. One limitation of our approach is that it requires labeled joint information. In the future, we plan to address this issue by developing self-supervised losses to automatically infer joint motions from data. Another limitation comes from the point cloud representation itself, which does not model continuous surfaces. We propose to address this issue by combining hybrid surface representations.

There are other opportunities for future work. One potential direction is to incorporate friction modeling into the formulation. Another direction is to explore the learned generator as a design space for designing articulated objects from 3D models. Finally, we would like to extend functional shape modeling to study kinematic motions among objects in a scene.

References

1. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.: Learning representations and generative models for 3d point clouds. In: Dy, J., Krause, A. (eds.) Proceedings of the 35th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 80, pp. 40–49. PMLR, Stockholm, Sweden (10–15 Jul 2018)
2. Bächer, M., Bickel, B., James, D.L., Pfister, H.: Fabricating articulated characters from skinned meshes. ACM Trans. Graph. **31**(4) (jul 2012)
3. Cai, R., Yang, G., Averbuch-Elor, H., Hao, Z., Belongie, S., Snavely, N., Hariharan, B.: Learning gradient fields for shape generation. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16. pp. 364–381. Springer, Springer International Publishing, Cham (2020)
4. Chaudhuri, S., Kalogerakis, E., Guibas, L.J., Koltun, V.: Probabilistic reasoning for assembly-based 3d modeling. ACM Trans. Graph. **30**(4), 35 (2011)
5. Chaudhuri, S., Ritchie, D., Wu, J., Xu, K., Zhang, H.R.: Learning generative models of 3d structures. Comput. Graph. Forum **39**(2), 643–666 (2020)
6. Coros, S., Thomaszewski, B., Noris, G., Sueda, S., Forberg, M., Sumner, R.W., Matusik, W., Bickel, B.: Computational design of mechanical characters. ACM Trans. Graph. **32**(4) (jul 2013)
7. Fan, H., Su, H., Guibas, L.J.: A point set generation network for 3d object reconstruction from a single image. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017. pp. 2463–2471. IEEE Computer Society, Honolulu, HI, USA (2017)
8. Funkhouser, T., Kazhdan, M., Shilane, P., Min, P., Kiefer, W., Tal, A., Rusinkiewicz, S., Dobkin, D.: Modeling by example. ACM Trans. Graph. **23**(3), 652–663 (aug 2004)
9. Gao, L., Yang, J., Wu, T., Yuan, Y.J., Fu, H., Lai, Y.K., Zhang, H.: Sdm-net: Deep generative network for structured deformable mesh. ACM Trans. Graph. **38**(6) (nov 2019)
10. Guo, J., Yan, D.M., Li, E., Dong, W., Wonka, P., Zhang, X.: Illustrating the disassembly of 3d models. Comput. Graph. **37**, 574–581 (2013)
11. Hauser, K.K., Shen, C., O'Brien, J.F.: Interactive deformation using modal analysis with constraints. In: Proceedings of the Graphics Interface 2003 Conference, Halifax, Nova Scotia, Canada, June 11-13, 2003. pp. 247–256. Canadian Human-Computer Communications Society, Nova Scotia, Canada (2003)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
13. Hu, R., Li, W., Van Kaick, O., Shamir, A., Zhang, H., Huang, H.: Learning to predict part mobility from a single static snapshot. ACM Trans. Graph. **36**(6) (nov 2017)
14. Huang, J., Wang, H., Birdal, T., Sung, M., Arrigoni, F., Hu, S.M., Guibas, L.J.: Multibodysync: Multi-body segmentation and motion estimation via 3d scan synchronization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7108–7118 (2021)
15. Huang, Q., Wicke, M., Adams, B., Guibas, L.J.: Shape decomposition using modal analysis. Comput. Graph. Forum **28**(2), 407–416 (2009)
16. Huang, Q., Huang, X., Sun, B., Zhang, Z., Jiang, J., Bajaj, C.: Arapreg: An as-rigid-as possible regularization loss for learning deformable shape generators.

In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 5815–5825. IEEE Computer Society, Montreal, Canada (October 2021)
17. Hui, L., Xu, R., Xie, J., Qian, J., Yang, J.: Progressive point cloud deconvolution generation network. In: European Conference on Computer Vision. pp. 397–413. Springer (2020)
18. Jones, R.K., Barton, T., Xu, X., Wang, K., Jiang, E., Guerrero, P., Mitra, N.J., Ritchie, D.: Shapeassembly: Learning to generate programs for 3d shape structure synthesis. ACM Trans. Graph. **39**(6) (nov 2020)
19. Kalogerakis, E., Chaudhuri, S., Koller, D., Koltun, V.: A probabilistic model for component-based shape synthesis. ACM Trans. Graph. **31**(4), 55:1–55:11 (2012)
20. Koch, J., Haraké, L., Jung, A., Dachsbacher, C.: Extending structurenet to generate physically feasible 3d shapes. In: VISIGRAPP (1: GRAPP). pp. 221–228. SCITEPRESS, Vienna, Austria (2021)
21. Kraevoy, V., Julius, D., Sheffer, A.: Model composition from interchangeable components. In: PG. pp. 129–138. IEEE Computer Society, Maui, HI, USA (2007)
22. Lau, M., Ohgawara, A., Mitani, J., Igarashi, T.: Converting 3d furniture models to fabricatable parts and connectors. In: ACM SIGGRAPH 2011 Papers. SIGGRAPH '11, Association for Computing Machinery, New York, NY, USA (2011)
23. Li, J., Niu, C., Xu, K.: Learning part generation and assembly for structure-aware shape synthesis. In: AAAI. pp. 11362–11369. AAAI Press, Vancouver, Canada (2020)
24. Li, J., Xu, K., Chaudhuri, S., Yumer, E., Zhang, H., Guibas, L.: Grass: Generative recursive autoencoders for shape structures. ACM Trans. Graph. **36**(4) (jul 2017)
25. Li, R., Li, X., Hui, K.H., Fu, C.W.: Sp-gan: Sphere-guided 3d shape generation and manipulation. ACM Transactions on Graphics (TOG) **40**(4), 1–12 (2021)
26. Lin, M., Shao, T., Zheng, Y., Mitra, N.J., Zhou, K.: Recovering functional mechanical assemblies from raw scans. IEEE Transactions on Visualization and Computer Graphics **24**, 1354–1367 (2018)
27. Megaro, V., Thomaszewski, B., Nitti, M., Hilliges, O., Gross, M., Coros, S.: Interactive design of 3d-printable robotic creatures. ACM Trans. Graph. **34**(6) (oct 2015)
28. Mezghanni, M., Boulkenafed, M., Lieutier, A., Ovsjanikov, M.: Physically-aware generative network for 3d shape modeling. In: CVPR. pp. 9330–9341. Computer Vision Foundation / IEEE, Nashville, Tennessee, USA (2021)
29. Mitra, N., Wand, M., Zhang, H.R., Cohen-Or, D., Kim, V., Huang, Q.X.: Structure-aware shape processing. In: SIGGRAPH Asia 2013 Courses. SA '13, Association for Computing Machinery, New York, NY, USA (2013)
30. Mo, K., Guerrero, P., Yi, L., Su, H., Wonka, P., Mitra, N.J., Guibas, L.J.: Structurenet: hierarchical graph networks for 3d shape generation. ACM Trans. Graph. **38**(6), 242:1–242:19 (2019)
31. Mo, K., Guibas, L.J., Mukadam, M., Gupta, A., Tulsiani, S.: Where2act: From pixels to actions for articulated 3d objects. CoRR **abs/2101.02692**, 6813–6823 (2021)
32. Mo, K., Guibas, L.J., Mukadam, M., Gupta, A., Tulsiani, S.: Where2act: From pixels to actions for articulated 3d objects. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 6813–6823 (October 2021)
33. Mo, K., Zhu, S., Chang, A.X., Yi, L., Tripathi, S., Guibas, L.J., Su, H.: Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object under-

standing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 909–918. IEEE, Long Beach, CA, USA (2019)
34. Nealen, A., Müller, M., Keiser, R., Boxerman, E., Carlson, M.: Physically based deformable models in computer graphics. In: Chrysanthou, Y., Magnor, M.A. (eds.) 26th Annual Conference of the European Association for Computer Graphics, Eurographics 2005 - State of the Art Reports, Dublin, Ireland, August 29 - September 2, 2005. pp. 71–94. Eurographics Association, Dublin, Ireland (2005)
35. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 652–660 (2017)
36. Shah, S., Koltun, V.: Robust continuous clustering. Proceedings of the National Academy of Sciences **114**, 201700770 (08 2017)
37. Song, P., Fu, Z., Liu, L., Fu, C.W.: Printing 3d objects with interlocking parts. Computer Aided Geometric Design **35-36** (03 2015)
38. Thomaszewski, B., Coros, S., Gauge, D., Megaro, V., Grinspun, E., Gross, M.: Computational design of linkage-based characters. ACM Trans. Graph. **33**(4) (jul 2014)
39. Tulsiani, S., Su, H., Guibas, L.J., Efros, A.A., Malik, J.: Learning shape abstractions by assembling volumetric primitives. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017. pp. 1466–1474. IEEE Computer Society, Honolulu, HI, USA (2017)
40. Wang, H., Schor, N., Hu, R., Huang, H., Cohen-Or, D., Huang, H.: Global-to-local generative model for 3d shapes. ACM Trans. Graph. **37**(6) (dec 2018)
41. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph cnn for learning on point clouds. ACM Trans. Graph. **38**(5) (oct 2019)
42. Wu, R., Zhuang, Y., Xu, K., Zhang, H., Chen, B.: PQ-NET: A generative part seq2seq network for 3d shapes. In: CVPR. pp. 826–835. Computer Vision Foundation / IEEE, Seattle, Washington (2020)
43. Wu, Z., Wang, X., Lin, D., Lischinski, D., Cohen-Or, D., Huang, H.: Sagnet: Structure-aware generative network for 3d-shape modeling. ACM Trans. Graph. **38**(4) (jul 2019)
44. Xiang, F., Qin, Y., Mo, K., Xia, Y., Zhu, H., Liu, F., Liu, M., Jiang, H., Yuan, Y., Wang, H., Yi, L., Chang, A.X., Guibas, L.J., Su, H.: SAPIEN: A simulated part-based interactive environment. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020)
45. Xu, K., Kim, V.G., Huang, Q., Mitra, N., Kalogerakis, E.: Data-driven shape analysis and processing. In: SIGGRAPH ASIA 2016 Courses. SA '16, Association for Computing Machinery, New York, NY, USA (2016)
46. Xu, K., Zhang, H., Cohen-Or, D., Chen, B.: Fit and diverse: Set evolution for inspiring 3d shape galleries. ACM Trans. Graph. **31**(4) (jul 2012)
47. Xu, W., Wang, J., Yin, K., Zhou, K., van de Panne, M., Chen, F., Guo, B.: Joint-aware manipulation of deformable models. In: ACM SIGGRAPH 2009 Papers. SIGGRAPH '09, Association for Computing Machinery, New York, NY, USA (2009)
48. Yang, G., Huang, X., Hao, Z., Liu, M.Y., Belongie, S., Hariharan, B.: Pointflow: 3d point cloud generation with continuous normalizing flows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 4541–4550. IEEE, Long Beach, CA, USA (2019)
49. Yang, J., Mo, K., Lai, Y., Guibas, L.J., Gao, L.: Dsm-net: Disentangled structured mesh net for controllable generation of fine geometry. CoRR **abs/2008.05440** (2020)

50. Yang, Y.L., Yang, Y.J., Pottmann, H., Mitra, N.J.: Shape space exploration of constrained meshes. In: Proceedings of the 2011 SIGGRAPH Asia Conference. SA '11, Association for Computing Machinery, New York, NY, USA (2011)
51. Yin, K., Chen, Z., Chaudhuri, S., Fisher, M., Kim, V.G., Zhang, H.R.: COALESCE: component assembly by learning to synthesize connections. In: 3DV. pp. 61–70. IEEE, Los Alamitos, CA, USA (2020)
52. Zhan, G., Fan, Q., Mo, K., Shao, L., Chen, B., Guibas, L.J., Dong, H.: Generative 3d part assembly via dynamic graph learning. In: NeurIPS. pp. 6315–6326. Curran Associates, Inc., Vancouver, BC, Canada (2020)
53. Zhang, H., van Kaick, O., Dyer, R.: Spectral mesh processing. Comput. Graph. Forum **29**(6), 1865–1894 (2010)
54. Zhang, Z., Yang, Z., Ma, C., Luo, L., Huth, A., Vouga, E., Huang, Q.: Deep generative modeling for scene synthesis via hybrid representations. ACM Trans. Graph. **39**(2), 17:1–17:21 (2020)
55. Zhu, L., Xu, W., Snyder, J., Liu, Y., Wang, G., Guo, B.: Motion-guided mechanical toy modeling. ACM Trans. Graph. **31**(6) (nov 2012)
56. Zou, C., Yumer, E., Yang, J., Ceylan, D., Hoiem, D.: 3d-prnn: Generating shape primitives with recurrent neural networks. In: ICCV. pp. 900–909. IEEE Computer Society, Venice, Italy (2017)

A Details of the Modal Analysis Regularization Term

This section presents additional details of the modal analysis based regularization term. In Section A.1, we describe the constraints along inter-part edges with respect to different types of joints. In Section A.2, we discuss the explicit expression of the stiffness matrix.

A.1 Constraints along Inter-Part Edges

There are three types of joints considered in this paper, namely, rigid joints, revolute joints, and prismatic joints. Here we consider an edge (i, j) whose end vertex positions are $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$; the latent rotation variable associated with vertex i is given by $\boldsymbol{c}_i$; the infinitesimal velocities of i and j are given by $\boldsymbol{u}_i$ and $\boldsymbol{u}_j$, respectively.

Rigid Joint For a rigid joint, the residual vector is given by

$$\boldsymbol{r}_e = \boldsymbol{c}_i \times (\boldsymbol{p}_i - \boldsymbol{p}_j) - (\boldsymbol{u}_i - \boldsymbol{u}_j).$$

This model reduces to the standard setting of typical joints.

Revolute Joint For a revolute joint, the additional variable is the rotation vector of $\boldsymbol{p}_j$ around the rotation axis of the joint:

$$\boldsymbol{r}_e = \boldsymbol{c}_i \times (\boldsymbol{p}_i - \boldsymbol{p}_j) - (\boldsymbol{u}_i - \boldsymbol{u}_j) - (s_e \boldsymbol{v}_e) \times \big(\boldsymbol{p}_j - \boldsymbol{o}_e\big)$$

where $\boldsymbol{o}_e$ is the rotation center; $\boldsymbol{v}_e$ represents the rotation axis; s_e is the latent variable that encodes the rotation velocity. Note that the order of $e = (i, j)$ is made consistent across all edges in the same cluster.

Prismatic Joint For a prismatic joint, the additional variable is a translation $\boldsymbol{t}_e$ specified by the joint. Note that this latent variable is shared among all edges that belong to the same cluster:

$$\boldsymbol{r}_e = \boldsymbol{c}_i \times (\boldsymbol{p}_i - \boldsymbol{p}_j) - (\boldsymbol{u}_i - \boldsymbol{u}_j) - \boldsymbol{s}_e \boldsymbol{t}_e.$$

A.2 Explicit Expression of the Stiffness Matrix

Let $\boldsymbol{y}$ collect the latent variables of $\boldsymbol{c}_i$ and $\boldsymbol{s}_e$. We can define the quadratic form:

$$\boldsymbol{u}^T H(\boldsymbol{z}) \boldsymbol{u} := \min_{\boldsymbol{y}} \begin{pmatrix} \boldsymbol{u} \\ \boldsymbol{y} \end{pmatrix}^T \begin{pmatrix} L \otimes I_3 & E \\ E^T & D \end{pmatrix} (\boldsymbol{u}\boldsymbol{y}) \tag{10}$$

where L is the unnormalized Laplacian matrix of the graph representation. E is a sparse matrix that specifies the relations between $\boldsymbol{u}$ and the latent variables.

Method	Laptop	Table	Microwave
Two-stage	**1.45**	**1.4**	1.575
Ours	1.55	1.6	**1.425**

Table 3: Human Evaluation result, score closes to 1 means better quality.

D is a block diagonal matrix. All interior vertices have its own block. Each joint and the associated boundary vertices form a block. It is easy to see that the optimal solution of (10) is given by $\boldsymbol{y} = -D^{-1}E^T\boldsymbol{u}$. Therefore,

$$\begin{aligned}\boldsymbol{u}^T H(\boldsymbol{z})\boldsymbol{u} &= \boldsymbol{u}^T(L \otimes I_3)\boldsymbol{u} + 2\boldsymbol{u}^T E\boldsymbol{y} + \boldsymbol{y}^T D\boldsymbol{y} \\ &= \boldsymbol{u}^T(L \otimes I_3)\boldsymbol{u} - 2\boldsymbol{u}^T ED^{-1}E^T\boldsymbol{u} + \boldsymbol{u}^T ED^{-1}E^T\boldsymbol{u} \\ &= (\boldsymbol{u})^T\big(L \otimes I_3 - ED^{-1}E^T\big)\boldsymbol{u}. \end{aligned} \tag{11}$$

Due to the block diagonal structure of D, the inverse D^{-1} can be computed efficiently.

B Additional visualization result for shape parsing and completion

We add three examples for Laptop, Table and Microwave in Figure 9. We can get reasonable completion result for various of classes.

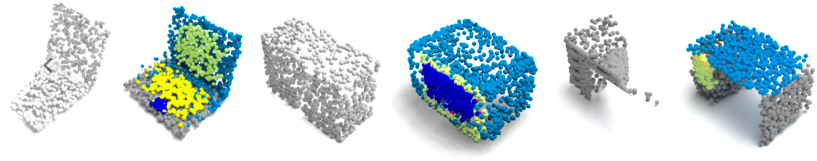

Fig. 9: More examples for shape completion

C Human Evaluation

We conduct a human evaluation to compare our method with two-stage synthesize method. For the two-stage synthesize method, we firstly use generate the Shape Only result. Then, we apply MultiBodySync [14] to classify the motion and parts. We do this evaluation with-in a group of volunteers of 10. We randomly generate 4 samples for the class Laptop, Table and Microwave for each

method. We let each volunteers to rank 1 and 2 for each samples. 1 represent the better one. We see from Table 3 that we get a compatible result comparing with two-stage end-to-end method. However, our one-stage generated model have a simpler pipeline.

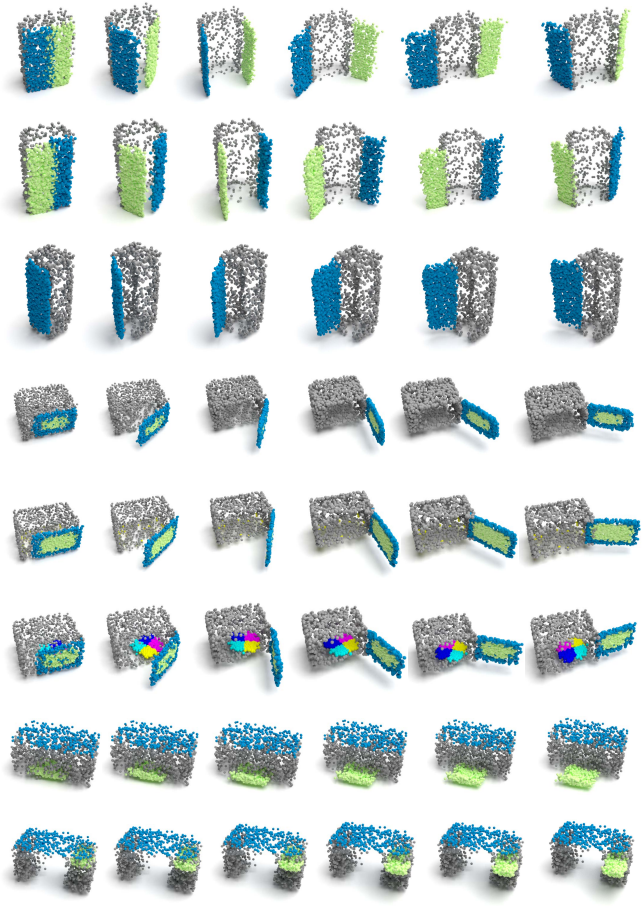

Fig. 10: We show a random set of generated shapes from each category. For the tray of Microwave, we color the corresponding points differently to illustrate the underlying revolute motions. From top to bottom: Refrigerator, Microwave, and Table.

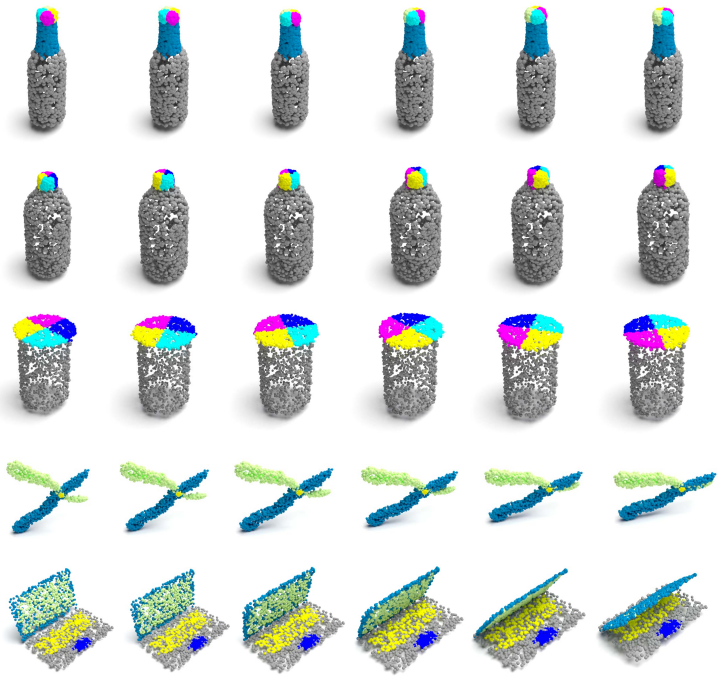

Fig. 11: We show a random set of generated shapes from each category. For the cap of Bottle, we color the corresponding points differently to illustrate the underlying revolute motions. From top to bottom: Bottle, Scissors, and Laptop.