NAP: *N*eural 3D *A*rticulated Object *P*rior

Jiahui Lei[1] **Congyue Deng**[2] **Bokui Shen**[2] **Leonidas Guibas**[2] **Kostas Daniilidis**[13]
[1] University of Pennsylvania [2] Stanford University [3] Archimedes, Athena RC
{leijh, kostas}@cis.upenn.edu, {congyue, willshen, guibas}@cs.stanford.edu

https://www.cis.upenn.edu/~leijh/projects/nap

Abstract

We propose Neural 3D Articulated object Prior (NAP), the first 3D deep generative model to synthesize 3D articulated object models. Despite the extensive research on generating 3D static objects, compositions, or scenes, there are hardly any approaches on capturing the distribution of articulated objects, a common object category for human and robot interaction. To generate articulated objects, we first design a novel articulation tree/graph parameterization and then apply a diffusion-denoising probabilistic model over this representation where articulated objects can be generated via denoising from random complete graphs. In order to capture both the geometry and the motion structure whose distribution will affect each other, we design a graph denoising network for learning the reverse diffusion process. We propose a novel distance that adapts widely used 3D generation metrics to our novel task to evaluate generation quality. Experiments demonstrate our high performance in articulated object generation as well as its applications on conditioned generation, including Part2Motion, PartNet-Imagination, Motion2Part, and GAPart2Object.

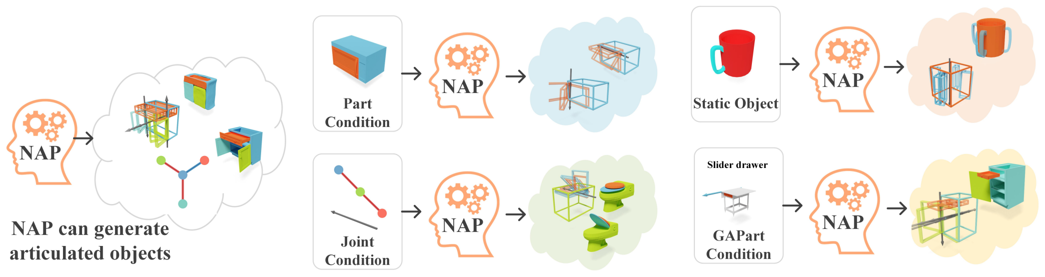


Figure 1: NAP can unconditionally generate articulated objects (left). It can be conditioned on just parts or joints (mid), a subset of parts plus joints, or over-segmented static objects (right).

1 Introduction

Articulated objects are prevalent in our daily life. As humans, we have strong prior knowledge of both object part geometry and their kinematic structures. Such knowledge is most heavily leveraged when a designer designs a cabinet from scratch, creating both its geometry and motion structure. For learning systems, an interesting challenge is to capture such priors as a generative model that can synthesize articulated objects from scratch. While there has been extensive research on generative models for static 3D objects [1–10], compositions [11–18], and scenes [19–30], the study of priors regarding closely linked 3D part geometry and 3D motion structures has been relatively neglected. In this work, we study *how to synthesize articulated objects*, i.e., how to generate a full description of an articulated object, an actual URDF [31], including how each part looks like, which pairs of parts are connected, and with what kind of joint. In contrast to static 3D generation, generating articulated objects involves modeling not only the distribution and composition of geometry but also the motion structures that determine the possible relative movements between rigid parts. Our task is also different than 3D human/hand synthesis [32–35] where the articulation structure is given and only the degrees of freedom are generated when predicting human poses/sequences. The generation of articulation can further impact simulation and mechanism design and be useful for inference when conditioned on either geometry or kinematic structure.

However, there are several challenges for articulated object generation. Existing datasets of articulated objects contain highly irregular data since articulated objects have different numbers of parts, diverse part connectivity topology, and different joint motion types. In order to enable efficient processing of diverse geometry and structures by a neural architecture, we propose a novel unifying articulation tree/graph parameterization (Sec. 3.1) to represent articulated objects. We take advantage of recent progress in diffusion models for 3D generation and develop a diffusion-denoising probabilistic model over our parameterization to faithfully model the irregular distribution of articulated objects (Sec. 3.2). Since we are modeling the joint distribution of the part geometry and the inter-part motion constraints, we design a graph denoising network (Sec. 3.3) to gradually exchange and fuse information between the edge and nodes on our articulation graph.

To evaluate articulated object generation quantitatively, we adopt the widely used 3D shape generation metric for articulated objects by introducing a novel distance measure between two articulated objects. Through extensive comparisons, we demonstrate high performance in articulated object synthesis and further conduct several ablations to understand the effect of each system component. Using the learned prior knowledge, we demonstrate conditioned generation applications, including Part2Motion, PartNet-Imagination, Motion2Part, and GAPart2Object .

In summary, our main contributions are **(1.)** introducing the articulated object synthesis problem; **(2.)** proposing a simple and effective articulation tree parameterization sufficiently efficient for a diffusion denoising probabilistic model that can generate articulated objects; **(3.)** introducing a novel distance for evaluating this new task; and **(4.)** demonstrating our high performance in articulated object generation and presenting several conditioned generation applications.

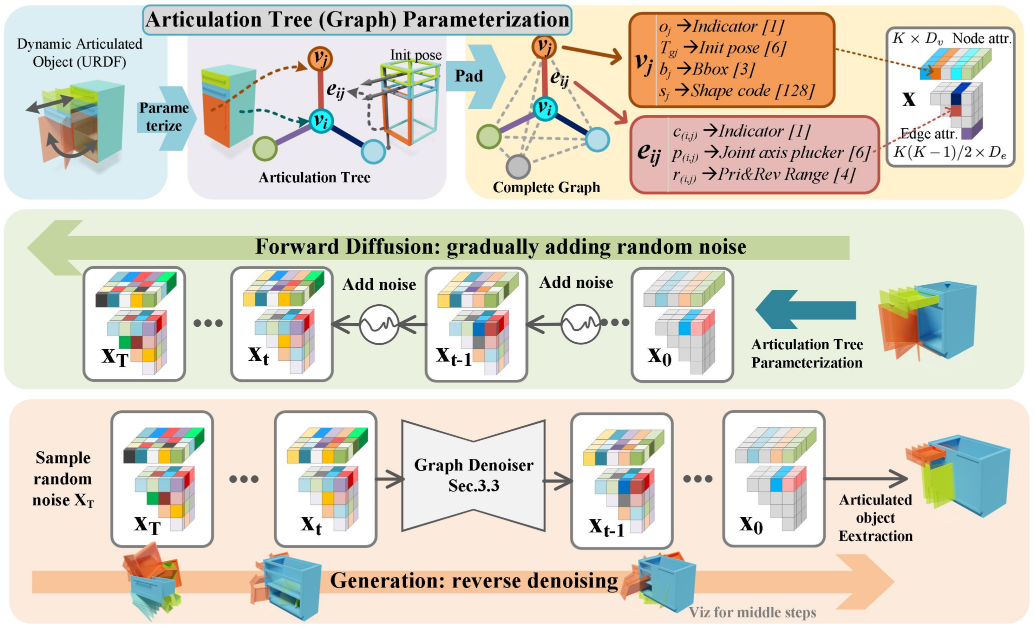
2 Related Work

Articulated object modeling. Modeling articulated objects has been a prolific field with existing works broadly classified into the categories of estimation, reconstruction, simulation, and finally, our generation. Note that there is a wider literature on semi-nonrigid objects, for example, human body [36], hands [37], and animals [38]. This paper focuses on everyday articulated objects, like the ones in PartNet-Mobility [39], which have more diverse and complex structures and are strictly multi-body systems. Estimation focuses on predicting the articulation *joint states (joint angles and displacements)* or *joint parameters (type, axis and limits)* from sensory observations, using approaches ranging from probabilistic models [40–44], interactive perception [45–51] and learning-based inference [52–65]. Reconstruction of articulated objects focuses on reconstructing both articulation and geometry properties of the objects, using techniques ranging from structure from motion [66], learning-based methods [67–69] and implicit neural representations [70–74]. While these methods mainly focus on surface reconstruction and joint states/parameter accuracy or are limited to pre-defined simple kinematic structures, we explicitly model the diverse and complex articulation motion structures. One closely related work to ours is [62], which predicts the joint parameters of articulated objects in PartNet [75] by training on PartNet-Mobility [39]. However, [62] is a single-point regression that does not capture the generative distribution of joints or parts. A growing field in robotics and embodied AI is building interactive environments that support physical interactions between the robot and the scene consisting of articulated objects [75, 39, 76–82]. Differing from all the above approaches, we build a **generative** prior of articulated objects, which extends beyond estimation and reconstruction. Such learned prior hasential further to accelerate the creation of realistic interactive 3D assets.

Generative models for structures. Generating structured articulated objects is closely related to generative models for structured data [83]. Part-based 3D object generation [11–18] has been widely studied with the main modeling target being the hierarchy of static rigid parts with or without a semantic label. Scene layout generation [19–30] utilizes compact scene parameterizations for indoor scene synthesis, where diffusion models on scene graphs have been recently introduced in [21, 22]. The generative diffusion approach has also been applied widely to 2D floor plans [84], protein structures [85], and graphs [86] .etc. Unlike all the existing works, our work focuses on modeling a new type of target – articulation structure plus part shapes, which requires joint reasoning between 3D geometry and kinematic structure.

Diffusion models in 3D. We mainly focus on the literature on diffusion models for 3D shape and motion generation. **Shape:** Diffusion models show impressive results in generating point clouds [1–3], meshes [87], implicit surfaces [4–10], neural radiance fields [88–91], or 4D non-rigid shapes [92]. However, these methods mostly focus on the single object level shape quality and do not pay attention to the kinematic structure. **Motion:** Diffusion models have seen many recent applications in motion generation given an articulation model. Such works generate text-conditioned human motion [32–35], and physically-viable [93], audio-driven [94, 95], scene-aware [96], multi-human [97] or animation [98] trajectories. Diffusion models for motion have also been applied to trajectory planning [99], visuomotor control [100], and rearrangement tasks [101]. Different from ours, existing works in motion diffusion rely on known geometries with known motion structures. We instead jointly model geometry and motion structure priors to create articulation models.

3 Method



Articulation Tree (Graph) Parameterization
Dynamic Articulated Object (URDF)
Parameterize
v_j
e_{ij}
v_i
Init pose
Articulation Tree
Pad
Complete Graph
v_j o_j →Indicator [1] T_{gj} →Init pose [6] b_j →Bbox [3] s_j →Shape code [128]
e_{ij} $c_{(i,j)}$ →Indicator [1] $p_{(i,j)}$ →Joint axis plucker [6] $r_{(i,j)}$ →Pri&Rev Range [4]
$K \times D_v$ Node attr.
$\mathbf{x}$
Edge attr. $K(K-1)/2 \times D_e$
Forward Diffusion: gradually adding random noise
$\mathbf{x}_T$
$\mathbf{x}_t$
Add noise
$\mathbf{x}_{t-1}$
Add noise
$\mathbf{x}_0$
Articulation Tree Parameterization
Sample random noise $\mathbf{X}_T$
Graph Denoiser Sec.3.3
Articulated object Eextraction
Generation: reverse denoising
Viz for middle steps

Figure 2: **Method**: **Parameterization (Top)**: We parameterize the articulated object as a tree, whose nodes are rigid parts and edges are joints; we then pad the tree to a complete graph of maximum node number and store it in the articulation graph attribute list $\mathbf{x}$. **Forward Diffusion (Middle)**: The parameterized attribute list $\mathbf{x}$ is gradually diffused to random noise. **Generation (Bottom)**: A Graph Denoiser (Fig 3, Sec. 3.3) samples a random articulation graph $\mathbf{x}_T$, gradually removes noise, and finally predicts $\mathbf{x}_0$. An object extraction stage (Sec. 3.2) including a minimum-spanning-tree algorithm is applied to the generated graph in the end to find the kinematic tree structure and the output articulated object.

We learn the articulated object priors using a diffusion model. However, an articulated structure must first be parameterized into a vector that can be the target of the diffusion. We introduce the parameterization of the articulated object as a tree (graph) in Sec. 3.1; the diffusion model on such parameterization in Sec. 3.2; and the denoising network in Sec. 3.3. An overview of our method is shown in Fig. 2.

3.1 Articulation Tree Parameterization

Graph-based representation. We follow the natural articulated object parameterization as in URDF [31] format where each object is defined by a graph with nodes being the parts and edges being the articulation joints. We make two assumptions: (1) *Tree assumption:* We assume no kinematic motion loop (cycle) exists in the graph and that the graph is connected. (2) *Screw joints:* We assume each edge is a screw [60] with at most one prismatic translation and one revolute rotation. Most real-world articulated objects [39, 76, 82, 102] satisfy the above assumptions and their limitations are further discussed in Sec. 5.

Nodes. As shown in Fig. 2-top, we represent each rigid part as one node in the tree. A node captures its rigid part shape by (1) a shape latent code $s_i \in \mathbb{R}^F$ (with F being channels, i being the part index) that can decode the SDF [103] of the part surface, and (2) the bounding box edge lengths $b_i \in \mathbb{R}^3$ that can scale the decoded SDF properly into the part's scale. We pre-train an SDF shape Auto-Encoder [104, 103] for obtaining s_i, further details are provided in our supplementary material. Importantly, one should also specify how to assemble these parts into an object before further modeling the articulation motion. We do so by adding to the node attributes a part initial rigid transformation $T_i \in SE(3)$ in the global object frame representing the part initial configuration. T_i comprises an axis-angle rotation and a translation, which can be written as a 6-dimensional vector. As articulated objects can be instantiated to different configurations so any of these configurations can be used as T_i, in practice, we consistently choose T_i by exploiting the canonicalization in the large annotated dataset: we use the part poses in their rest states (zero joint angles and displacements) for T_i. Since we observe current datasets often align their rest states consistently to static un-articulated objects, for example, PartNet-Mobility [39] aligns with static PartNet [75]. To model the varying number of parts across objects, we define a maximum number of K parts as well as a per-part binary indicator $o_i \in \{0, 1\}$ of part i existence. In summary, a node i has an attribute vector $v_i = [o_i, T_i, b_i, s_i]$ with dimension $D_v = 1 + 6 + 3 + F$, and the overall node component of an object graph is a feature of dimension $K \times D_v$.

Edges. The edge in the graph represents the motion constraint of each articulation joint. A joint possesses a 3D axis (a directed line), around which the revolute joint can rotate and along which the prismatic joint can translate. Inspired by [60], such a joint axis is represented by Plücker coordinates ($l \in \mathbb{S}^2, m \in \mathbb{R}^3$). Here the 3D directional vector l is from a unit sphere $\mathbb{S}^2$ and the momentum m is perpendicular to l. Such representation avoids defining additional local joint coordinate frames with ambiguity (e.g. translating the joint coordinate frame along a revolute axis leads to equivalent joints). Such 6-dimensional Plücker coordinates $p_{(i,j)} = (l_{(i,j)}, m_{(i,j)})$ for the joint from part i to j are defined in the global object frame when parts are in their initial rest configuration T_i, T_j. To fully define the joint motion constraint, we also incorporate two joint state ranges $r_{(i,j)} \in \mathbb{R}^{2\times 2}$ for both the prismatic translation and the revolute angle components (the left limit of the range can be negative). A purely prismatic joint will have its revolute component range set to $[0, 0]$ and vice versa. Following the node padding, we also pad the edges to a complete graph and use an indicator $c_{(i,j)}$ denoting the edge existence. Note that the above joint axis and range have a parent-child direction from part i to j. When the direction flips, a notable benefit of expressing $p_{(i,j)}$ in the global object frame, rather than the local part frame, is the inherent relationship $p_{(i,j)} = -p_{(j,i)}$ and $r_{(i,j)} = r_{(j,i)}$, which motivates us to model the padded graph with only $K(K-1)/2$ edges for all $i < j$ pairs. Since the nodes have no specific order, to avoid the sign flipping in $p_{(i,j)} = -p_{(j,i)}$ when permuting the nodes and to help the network learn a more stable prior, we explicitly embed the sign inside $p_{(i,j)} = -p_{(j,i)}$ into the edge existence indicator $c_{(i,j)} \in \{-1, 0, +1\}$. Here 0 indicates the non-existence of an edge and $+1, -1$ indicate the chiralities of existing edges, leading to $\hat{p}_{(i,j)} = \hat{p}_{(j,i)}$ (where $\hat{p}$ denotes the actual prediction target). In essence, each edge is characterized by an attribute vector $e_{(i,j)} = [c_{(i,j)}, p_{(i,j)}, r_{(i,j)}]$ with a dimension of $D_e = 1 + 6 + 4$. The overall edge parameterization of an object has shape $K(K-1)/2 \times D_e$. We will see later in Sec. 3.2 how to extract an articulated object model from the diffusion model prediction via the Minimum Spanning Tree.

3.2 Diffusion-Based Articulation Tree Generation

Our goal is to learn the distribution of articulated objects parameterized by the complete articulation graphs $\mathbf{x} = (\{v_i\}, \{e_{(i,j)}\}) \in \mathbb{R}^{KD_v + K(K-1)D_e/2}$. We apply a diffusion denoising probabilistic model [105] directly over the distribution of $\mathbf{x}$.

Forward diffusion. Given an articulation graph $\mathbf{x}_0$ from an object distribution $q(\mathbf{x}_0)$, we gradually add Gaussian noise with variance schedule $\beta_1 < \cdots < \beta_T$ and obtain a sequence $\mathbf{x}_1, \cdots, \mathbf{x}_T$ ended with a Gaussian distribution $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$. The joint distribution of the forward process is:

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) := \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}), \quad q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}). \tag{1}$$

A notable property is that $\mathbf{x}_t$ at arbitrary timestep t can be directly sampled from $\mathbf{x}_0$ with

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}) \tag{2}$$

where $\alpha_t := 1 - \beta_t$ and $\bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$.

Reverse process. Starting from a standard Gaussian distribution $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, we aim to learn a series of Gaussian transitions $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ parameterized by a neural network with learnable weight θ that gradually removes the noise. The distribution of the reverse process is:

$$p_\theta(\mathbf{x}_{0:T}) := p(\mathbf{x}_T)\prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t), \quad p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) := \mathcal{N}(\mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)). \tag{3}$$

Following [105], we set $\Sigma_\theta(\mathbf{x}_t, t) = \sigma_t^2\mathbf{I}$ and model this reverse process with Langevin dynamics

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}_t, t)\right) + \sigma_t\mathbf{z}, \quad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{4}$$

where $\epsilon(\mathbf{x}_t, t)$ is a learnable network approximating the per-step noise on $\mathbf{x}_t$.

Training objective. We optimize the variational bound on the negative log-likelihood

$$L := \mathbb{E}_q\left[-\log\frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t\geq 1}\log\frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] \geq \mathbb{E}[-\log p_\theta(\mathbf{x}_0)] \tag{5}$$

With Eq. 4, the objective simplifies to

$$\mathbb{E}_{\mathbf{x}_0,\epsilon}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}\left\|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)\right\|^2\right] \tag{6}$$

w.r.t. learnable network ϵ_θ. We refer the readers to [105] for more details.

Output Extraction. The above-described Euclidean space denoising intermediate steps as well as the final output may not strictly lie on the manifold of articulated objects parameterized as in Sec. 3.1 (e.g. the generated joint Plücker coordinates may not be valid because l and m may not be orthogonal). We found that utilizing the diffusion as in Euclidean space and projecting the generated $\mathbf{x}$ back to a valid articulation graph in the final step already leads to practically good results. As mentioned in Sec. 3.1, once we have completed the denoising process, a post-processing step is applied to obtain the final articulated object model from the generated $\mathbf{x}$. First, we identify the existing nodes by the generated nodes indicator o with a threshold and make sure there are at least two foreground nodes. Then we use the predicted edge chirality $|c|$ as the edge value to find the minimum spanning tree in the generated graph as the output tree topology. As in Sec. 3.1, we model half of the edges in the complete graph with node index $i < j$ and put the edge direction explicitly to the chirality sign. During the object extraction stage, if $c < 0$, we flip the predicted joint direction on this edge and if $c > 0$ we keep the joint direction, which is equivalent to flipping the parent and child order of this edge when $c < 0$. Finally, the predicted joint coordinates are projected from $\mathbb{R}^6$ to Plücker coordinates by normalizing the predicted l to unit vector and subtracting the parallel component of m on l to make predicted m orthogonal to l. We decode the part shape code to an SDF and extract the part mesh via marching cubes. We alternatively can retrieve the nearest part in the training set as most scene generation methods do [23, 22].

Conditioned Generation. A favorable property of diffusion models is that conditions can be directly added to the inference processes using Bayes' rule without any modification to the training process. Here we perform conditioned generation by fixing the known part of variable $\mathbf{x}$ as in the image inpainting [106] and shape completion works. For a variable $\mathbf{x} = m\odot\mathbf{x}^{\text{known}} + (1-m)\odot\mathbf{x}^{\text{unknown}}$ ($\odot$ means element-wise multiplication) with known entries $\mathbf{x}^{\text{known}}$ fixed unknown entries $\mathbf{x}^{\text{unknown}}$ to be completed which are separated by mask m, we can sample the known entries directly following Eq. 2 by adding Gaussian noise to the known input and generating the unknown entries using reverse diffusion,

$$q(\mathbf{x}_{t-1}^{\text{known}}|\mathbf{x}_0) = \mathcal{N}(\sqrt{\bar{\alpha}}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}), \quad p_\theta(\mathbf{x}_{t-1}^{\text{unknown}}|\mathbf{x}_t) = \mathcal{N}(\mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)). \tag{7}$$

Since our method directly diffuses in graph space, we can apply precise control of the parts and joints condition, enabling, thus, a disentanglement. In Sec. 4.4, we will show applications with different $\mathbf{x}^{\text{known}}$ and $\mathbf{x}^{\text{unknown}}$.

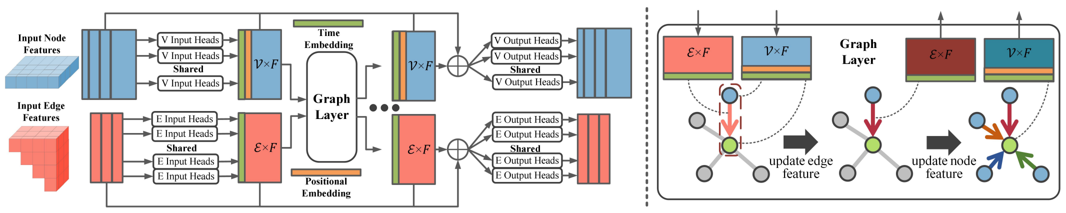


Figure 3: Network architecture. Left: input the node and edge list of a noisy articulation graph, a stack of graph layers will fuse and exchange information on the graph and output the noise that has to be removed. Right: details in the graph layer.

3.3 Denoising Network

Network architecture. Since $\mathbf{x}$ represents both the part geometry and the joint motion constraints, we utilize a graph denoising network as shown in Fig. 3, which exchanges and fuses information between parts and joints. The network inputs a noisy articulate graph $\mathbf{x}$ and outputs noise in the same shape as $\mathbf{x}$. The input attributes for every node are first encoded by input heads shared across nodes to a list of node features $\{f_i\}$. Similarly, edge attributes are encoded to a list of edge features $\{g_{(i,j)}\}$. Then the node and edge feature lists are updated via the graph layers (Fig. 3 right). Finally, all hidden features, including the input attributes, are concatenated together and decoded to the outputs via shared output heads. Note that the network is shared across all time steps through denoising, and the time step is encoded and concatenated to the hidden features. Similar to recent work on scene graph generation [22], we also append the positional encoding of the part instance id to the node features to provide stronger guidance in the early denoising stages when the part pose information is ambiguous. To learn with positional encoding, we randomly permute the nodes' order during training.

Graph layer. The key building block of the denoising network is the graph layer shown in Fig. 3 right. The edge feature is first updated via an edge MLP by fusing the input edge feature $g_{(i,j)}$ and neighboring nodes i, j's features: $g'_{(i,j)} = \text{MLP}(f_i, f_j, g_{(i,j)})$. Then, we aggregate the updated edge features to the nodes by attention weights. We compute the query $Q(f_i)$ and key $K(f_i)$ from the input node features via two MLPs and use their inner product as the attention weights; the graph attention update of the node i is: $f'_i = \sum_{j=1}^{K} \text{softmax}_j(Q(f_i)^T K(f_j)) g'_{(i,j)}$. We additionally do a PointNet [107]-like global pooling over all the graph node features after the attention aggregation to capture more global information.

4 Experiments

We examine 4 important questions with our experiments: (1) How can we evaluate articulated object generation? (Sec. 4.1) (2) How well does NAP capture the distribution of articulated objects? (Sec. 4.2) (3) How effective is each of NAP's components? (Sec. 4.3) (4) What applications can NAP enable? (Sec. 4.4)

4.1 Evaluation Metrics

Since we are the first to study articulated objects in a generative setting, we propose a new distance metric between two articulated objects for adopting widely used shape generation metrics. Since a generated object's shape and motion structure are dependent, we can not evaluate them separately. Such a challenge is poorly addressed in existing works. In articulated object modeling, existing works either consider a fixed kinematic structure [71, 70] or a given geometry [108]. In graph generation works [109–111], structures are examined by themselves without need to measure geometry. Thus, we propose a new distance metric, **Instantiation Distance (ID)**, to measure the distance between two articulated objects considering both the part geometry and the overall motion structure.

We treat an articulated object O as a template that, given the joint states $q \in \mathcal{Q}_O$ in object's joint range $\mathcal{Q}_O$, it returns the overall articulate mesh $\mathcal{M}(q)$ and the list of part poses $\mathcal{T}(q) = \{T_{\text{part}} \in SE(3)\}$. We compute the distance between two articulated objects in different joint states by

$$\tilde{d}(O_1, q_1, O_2, q_2) = \min_{T_i \in \mathcal{T}_1(q_1), T_j \in \mathcal{T}_2(q_2)} \left\{ D(T_i^{-1}\mathcal{M}_1(q_1), T_j^{-1}\mathcal{M}_2(q_2)) \right\}, \tag{8}$$

Table 1: Articualted object synthesis comparison with Instantiation Distance

	Part SDF Shape			Part Retrieval Shape		
Generative Paradigm/Method	MMD $\downarrow$	COV $\uparrow$	1-NNA $\downarrow$	MMD $\downarrow$	COV $\uparrow$	1-NNA $\downarrow$
Auto-Decoding (StructNet)	0.0435	0.1871	0.8820	0.0390	0.2316	0.8675
Variational Auto-Encoding (StructNet)	0.0311	0.3497	0.8085	0.0289	0.3363	0.7918
Autoregressive (ATISS-Tree)	0.0397	0.3808	0.6860	0.0333	0.4120	0.6782
Latent Diffusion (StructNet)	0.0314	0.4365	0.6269	0.0288	0.4477	0.6102
Articulation Graph Diffusion (Ours)	**0.0268**	**0.4944**	**0.5690**	**0.0215**	**0.5234**	**0.5412**

where $T_i^{-1}\mathcal{M}_1(q_1)$ means canonicalizing the mesh using its ith part pose, and D is a standard distance that measures the distance between two static meshes. Specifically, we sample $N = 2048$ points from two meshes and compute their Chamfer Distance. Intuitively, the above distance measures the minimum distance between two posed articulated objects by trying all possible canonicalization combinations. Then, we define the instantiation distance between O_1 and O_2 as:

$$\begin{aligned} ID(O_1, O_2) =& \mathbb{E}_{q_1 \in \mathcal{U}(\mathcal{Q}_{O_1})}\left[\inf_{q_2 \in \mathcal{Q}_{O_2}} \left(\tilde{d}(O_1, q_1, O_2, q_2)\right)\right] \\ &+ \mathbb{E}_{q_2 \in \mathcal{U}(\mathcal{Q}_{O_2})}\left[\inf_{q_1 \in \mathcal{Q}_{O_1}} \left(\tilde{d}(O_1, q_1, O_2, q_2)\right)\right], \end{aligned} \tag{9}$$

where $q \in \mathcal{U}(\mathcal{Q}_O)$ means uniformly sample joint poses from the joint states range. The instantiation distance measures the two-side expectation of minimum distance to the other object over all possible joint configurations. However, the $\inf$ inside the expectation requires expensive registration between two articulated objects so it is non-trackable in practice when computing all distance pairs between the reference and sampled object sets. In practice, we approximate the above distance by uniformly sampling M joint poses $Q_1 = \{q_k | q_k \in \mathcal{U}(\mathcal{Q}_{O_1}), k = 1, \dots, M\}$ and the approximated distance is:

$$ID(O_1, O_2) \approx \frac{1}{M}\sum_{q_1 \in Q_1}\left[\min_{q_2 \in Q_2}\left(\tilde{d}(O_1, q_1, O_2, q_2)\right)\right] + \frac{1}{M}\sum_{q_2 \in Q_2}\left[\min_{q_1 \in Q_1}\left(\tilde{d}(O_1, q_1, O_2, q_2)\right)\right], \tag{10}$$

and we set $M = 10$ in our ID for all evaluations. This pairwise distance can be plugged into the standard metrics for shape generation. Specifically, we adopt the following three metrics [112] for our evaluation: **minimum matching distance (MMD)** that measures the generation quality, **coverage (COV)** that tests the fraction the reference set is covered, and **1-nearest neighbor accuracy (1-NNA)** that measures the distance between the two distributions by 1-nn classification accuracy.

4.2 Articulated Object Synthesis

Baselines. We adapt existing models in related tasks and compare our method with them. Specifically, we adapt architecture designs from semantic-part-based shape generation [11] and scene-graph-based indoor scene synthesis [23], and equip them with different generative paradigms including auto-decoding [113], VAE [11], autoregressive models [23], and latent diffusion [4, 5]. We refer to our supplementary for adaptation and details of these baselines.

Setups. We train all the methods on PartNet-Mobility [39] across all categories jointly, with a maximum 8 rigid parts ($K = 8$) and a train-val-test split ratio $[0.7, 0.1, 0.2]$. Since the dataset does not include parts orientation and all initial part poses have rotation $\boldsymbol{I}$, we ignore the rotation in the parameterization (Sec. 3.1) for all the methods. For a fair comparison, all the methods are controlled to a similar number of learnable parameters. We evaluate the generated articulated object models with both the reconstructed meshes from the generated shape code and the retrieved part meshes from the training set.

Comparison. We report quantitative results in Tab. 1, and qualitative comparisons in Fig. 4. While the Auto-Decoding baseline fits the training set, its generation performance is poor when sampling in the latent space as shown in Fig. 4-AD, which suggests a weak regularization in the latent space. Using VAE to replace auto-decoding brings about better regularization of the latent space, resulting in an increase in all metrics. Sampling from the prior of VAE also leads to more meaningful generations, as shown in Fig. 4-VAE. For the autoregressive ATISS-Tree baseline, we see a relative

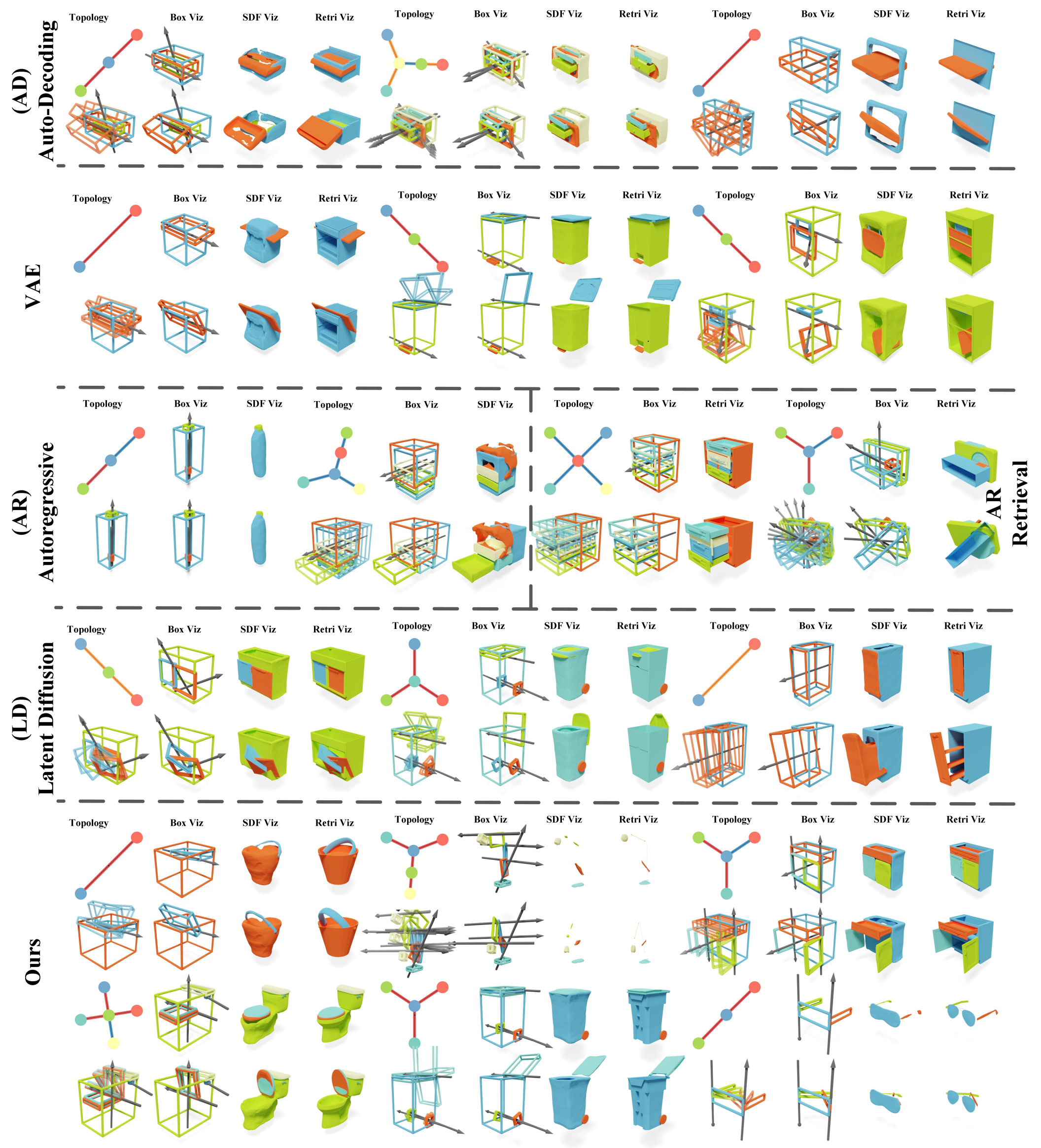


Figure 4: **Articulated object generation results.** Each generated object is visualized with (1) graph topology (top left), where the edge color means blue–prismatic, red–revolute, and orange–hybrid; (2) the predicted part bounding boxes and joints under different joint states (second column), and the overlay of multiple states reflecting the possible motion (bottom left); (3) reconstructed part meshes from the generated shape code (third column); (4) retrieved part meshes (right column).

increase in a majority of metrics compared to VAE. Part shape retrieval leads to further improvement in both motion structure and part shape since using retrieval at each autoregressive step can decrease the deviation from the training distribution. Interestingly, as shown in Fig. 4, we observe that the autoregressive method also has a tendency to append too many nodes to the tree, resulting in overlapping parts. Latent diffusion works the best among our baselines. We hypothesize that it is due to the superiority of the diffusion model as a sampler in the latent space, mapping the prior Gaussian to reliable regions where the trained decoder performs well near training samples. However, as the generation happens in the latent space and the generated latent code have to be decoded, slight error or changes in the latent space may lead to unrealistic or wrong articulations, which is shown in Fig. 4-LD. Different from the latent diffusion, our method directly applies diffusion in the articulation tree space, which can generate diverse and high-quality articulation models and achieves a better performance comparing to baselines.

Table 2: Ablation studies with Instantiation Distance

	Part SDF Shape			Part Retrieval Shape		
Ablation	MMD ↓	COV ↑	1-NNA ↓	MMD ↓	COV ↑	1-NNA ↓
Full	**0.0268**	**0.4944**	0.5690	**0.0215**	0.5234	**0.5412**
No PE	0.0282	0.4766	**0.5490**	0.0227	**0.5457**	0.5557
No Attn. on Edge	0.0286	0.4766	0.5668	0.0232	0.5234	0.5568
No Graph Conv	0.0331	0.4432	0.6570	0.0281	0.4722	0.6481

4.3 Ablation Studies

We verify our denoising network design by ablating components in our full network, and the comparison is shown in Tab. 2. Specifically, we examine the effectiveness of positional encoding, attention weights on edges, and graph convolutions. First, removing the positional encoding will slightly worsen performance, since its presence can help to distinguish the nodes at the early stages of reverse diffusion [22]. Second, replacing attention weights with mean pooling when aggregating the neighboring nodes' information results in a drop in performance. Finally, we justify the importance of node-edge information exchange in the graph convolution by removing the graph convolution layer and replacing it with a PointNet [107]-like global pooling layer, where the connectivity of the graph is ignored and the information exchanges through a symmetric global pooling. We observe a larger performance decrease, which justifies our graph-based processing.

4.4 Applications with Conditioned Generation

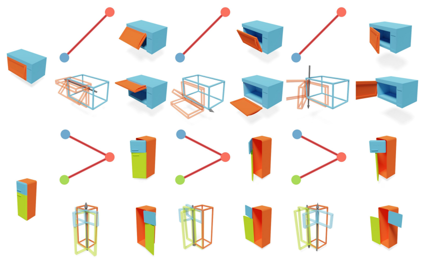


Figure 5: **Part2Motion**: Known part condition on the left, diverse motion proposals on the right.

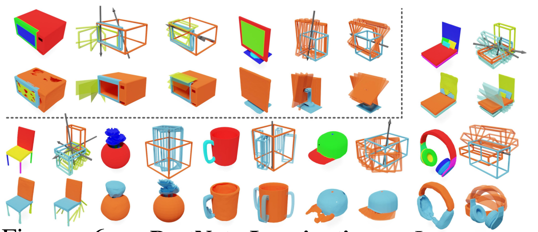


Figure 6: **PartNet Imagination**: Input over-segmented static PartNet shapes (top left) are grouped into rigid parts (bottom left) and be hallucinated with articulations (right). Both training categories (encircled by dashed lines) and out-of-distribution objects are shown.

Once trained, our method can be used directly for conditioned generation. Following Sec. 3.2, we demo applications conditioning on various known attributes in $\mathbf{x}$.

Part2Motion. We first show that when knowing the static part attributes, how NAP can suggest motion structures, i.e., with $\mathbf{x}^{\text{known}} = \{v_i\}$ and $\mathbf{x}^{\text{unknown}} = \{e_{(i,j)}\}$ in Eq. 7. We use the unseen object part attributes from the test set as conditions and the generated motion structure is in Fig. 5. We observe diverse and plausible motion suggestions that cover the ambiguity of the closed doors.

PartNet Imagination. NAP uses PartNet-Mobility [39] for training, which is only a small subset of the large-scale but static PartNet [75]. We show that NAP can be used to imagine possible motion structures in static PartNet [75]. Starting from the finest semantic part labeled in PartNet [75] (as an over-segmentation), we use a simple contrastive learned grouping encoder (see Suppl.) to group the fine semantic parts into rigid parts and then apply the same node condition as in Part2Motion. Fig. 6 shows our predictions. Interestingly, we find the learned articulation prior can be applied to out-of-distribution object categories and generate intriguing motion structures that adhere to some human recognizable logic (e.g., the mug in Fig. 6 bottom, where the handle rotates around the body). This further suggests that our model captures an underlying articulation prior.

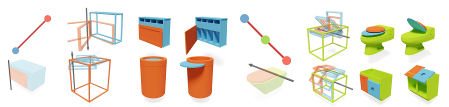


Figure 7: **Motion2Part**: Known motion structure (left) and suggested parts (right).

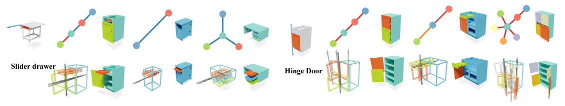


Figure 8: **GAPart2Object**: GAParts [81] (left) completed into full articulated objects (right).

Motion2Part. Similar to Part2Motion, we can flip the conditions and suggest possible parts from the motion structure. Given known edge attributes and graph topology $\mathbf{x}^{\text{known}} = \{e_{(i,j)}\} \bigcup \{o_i\}$ and $\mathbf{x}^{\text{unknown}} = \{T_{gi}, b_i, f_i\}$, NAP can generate possible parts that fit the motion structure in Fig. 7.
GAPart2Object. As Part2Motion and Motion2Part are pure edge or node conditions, we can also combine the node and edge conditions to complete an articulated object from one part. GAPart [81] is human-labeled semantically generalizable parts and one GAPart has a part geometry plus a joint type. We use the GAPart [81] from our testset as the known condition. Specifically, we set $\mathbf{x}^{\text{known}} = \{[o_0, b_0, f_0], [o_1]\} \bigcup \{[c_{(0,1)}, l_{(0,1)}, r_{(0,1)}]\}$ otherwise unknown, where we ensure that a GAPart is part-0 and there must be one part-1 connected to it. We constrain the joint direction $l_{(0,1)}$ but free the momentum $m_{(0,1)}$ and also let the non-zero joint range be freely denoised. The completion results are shown in Fig. 8.

5 Conclusions

We introduce the articulated object synthesis problem to our community and propose the first deep neural network that is able to generate articulated object models. Such generation is achieved via a diffusion model over a novel articulation graph parameterization with a graph denoising network. We further introduced a new Instantiation Distance to adopt the widely used 3D shape generation metrics to this task. Our learned neural 3D articulation can be used under different conditions for diverse application settings. We believe this initial step towards articulated object understanding and generation will bring more opportunities to understand and simulate our dynamic and complex real world.

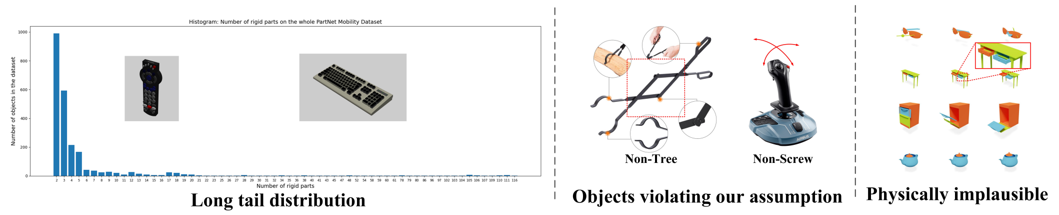


Figure 9: Limitations: **Left** The number of rigid parts in the current dataset has a long tail distribution, leading to extremely challenging objects, for example, a remote or a keyboard. **Middle**: Object violating tree or screw joint assumption. **Right**: The generated objects do not strictly obey physical constraints, for example, self-collision, making them not directly simulatable.

Limitations and future work. Although we have taken initial steps towards this new challenge, there are still many issues that require further exploration.

1. Long tail distribution and unbalanced structures: We noticed that our methods and baselines work better for small graphs. Large graphs with many parts are rare in the training set, making the generative model biased towards small graphs.

2. Overfitting: Like many other generative models, we do observe that NAP slightly overfits the training dataset. One bottleneck is that the current dataset still contains too few objects (around 2k) for training generative models. Studying how to learn an articulation prior from the camera observations or building a larger dataset will help to address this issue.

3. Tree and screw assumption: As shown in Fig. 9-Mid. NAP currently can not model kinematic cycles (pairs) or model several degrees of freedom on a single joint. Introducing dummy nodes and synchronization between joints may solve this limitation.

4. More structured diffusion: Although demonstrated to be effective, the diffusion we applied is straightforward and may be improved by introducing discretization, for example, the MST in the intermediate steps or by constraining the diffusion to the Plücker manifold.

5. Physically plausible generation: As shown in Fig. 9-Right, another exciting direction is how to generate directly simulatable objects whose joints and parts geometry fulfill physical constraints. This may require novel intersection loss and optimization techniques between parts and joints.

Broader Impacts. We do not see any direct negative ethical aspects or societal consequences of our work. Our paper can broadly improve the perception, digitization, and simulation of our physical world.

Acknowledgments and Disclosure of Funding

The authors appreciate the support of the following grants: NSF FRR 2220868, NSF IIS-RI 2212433, NSF TRIPODS 1934960, NSF CPS 2038873 awarded to UPenn; TRI University 2.0 grant, ARL grant W911NF-21-2-0104, and a Vannevar Bush Faculty Fellowship awarded to Stanford University.

References

[1] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2837–2845, 2021.

[2] Linqi Zhou, Yilun Du, and Jiajun Wu. 3d shape generation and completion through point-voxel diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5826–5835, 2021.

[3] Michał J Tyszkiewicz, Pascal Fua, and Eduard Trulls. Gecco: Geometrically-conditioned point diffusion models. *arXiv preprint arXiv:2303.05916*, 2023.

[4] Gimin Nam, Mariem Khlifi, Andrew Rodriguez, Alberto Tono, Linqi Zhou, and Paul Guerrero. 3d-ldm: Neural implicit 3d shape generation with latent diffusion models. *arXiv preprint arXiv:2212.00842*, 2022.

[5] Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. Lion: Latent point diffusion models for 3d shape generation. *arXiv preprint arXiv:2210.06978*, 2022.

[6] Heewoo Jun and Alex Nichol. Shap-e: Generating conditional 3d implicit functions. *arXiv preprint arXiv:2305.02463*, 2023.

[7] Muheng Li, Yueqi Duan, Jie Zhou, and Jiwen Lu. Diffusion-sdf: Text-to-shape via voxelized diffusion. *arXiv preprint arXiv:2212.03293*, 2022.

[8] Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *arXiv preprint arXiv:2301.11445*, 2023.

[9] Ka-Hei Hui, Ruihui Li, Jingyu Hu, and Chi-Wing Fu. Neural wavelet-domain diffusion for 3d shape generation. In *SIGGRAPH Asia 2022 Conference Papers*, pages 1–9, 2022.

[10] Xin-Yang Zheng, Hao Pan, Peng-Shuai Wang, Xin Tong, Yang Liu, and Heung-Yeung Shum. Locally attentional sdf diffusion for controllable 3d shape generation. *arXiv preprint arXiv:2305.04461*, 2023.

[11] Kaichun Mo, Paul Guerrero, Li Yi, Hao Su, Peter Wonka, Niloy Mitra, and Leonidas Guibas. Structurenet: Hierarchical graph networks for 3d shape generation. *ACM Transactions on Graphics (TOG), Siggraph Asia 2019*, 38(6):Article 242, 2019.

[12] Lin Gao, Jie Yang, Tong Wu, Yu-Jie Yuan, Hongbo Fu, Yu-Kun Lai, and Hao Zhang. Sdm-net: Deep generative network for structured deformable mesh. *ACM Transactions on Graphics (TOG)*, 38(6):1–15, 2019.

[13] Rundi Wu, Yixin Zhuang, Kai Xu, Hao Zhang, and Baoquan Chen. Pq-net: A generative part seq2seq network for 3d shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 829–838, 2020.

[14] Guanqi Zhan, Qingnan Fan, Kaichun Mo, Lin Shao, Baoquan Chen, Leonidas J Guibas, Hao Dong, et al. Generative 3d part assembly via dynamic graph learning. *Advances in Neural Information Processing Systems*, 33:6315–6326, 2020.

[15] Ka-Hei Hui, Ruihui Li, Jingyu Hu, and Chi-Wing Fu. Neural template: Topology-aware reconstruction and disentangled generation of 3d meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18572–18582, 2022.

[16] Kamal Gupta, Justin Lazarow, Alessandro Achille, Larry S Davis, Vijay Mahadevan, and Abhinav Shrivastava. Layouttransformer: Layout generation and completion with self-attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1004–1014, 2021.

[17] Yichen Li, Kaichun Mo, Yueqi Duan, He Wang, Jiequan Zhang, Lin Shao, Wojciech Matusik, and Leonidas Guibas. Category-level multi-part multi-joint 3d shape assembly. *arXiv preprint arXiv:2303.06163*, 2023.

[18] Jun Li, Chengjie Niu, and Kai Xu. Learning part generation and assembly for structure-aware shape synthesis. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 11362–11369, 2020.

[19] Zaiwei Zhang, Zhenpei Yang, Chongyang Ma, Linjie Luo, Alexander Huth, Etienne Vouga, and Qixing Huang. Deep generative modeling for scene synthesis via hybrid representations. *ACM Transactions on Graphics (TOG)*, 39(2):1–21, 2020.

[20] Helisa Dhamo, Fabian Manhardt, Nassir Navab, and Federico Tombari. Graph-to-3d: End-to-end generation and manipulation of 3d scenes using scene graphs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16352–16361, 2021.

[21] Qiuhong Anna Wei, Sijie Ding, Jeong Joon Park, Rahul Sajnani, Adrien Poulenard, Srinath Sridhar, and Leonidas Guibas. Lego-net: Learning regular rearrangements of objects in rooms. *arXiv e-prints*, pages arXiv–2301, 2023.

[22] Jiapeng Tang, Yinyu Nie, Lev Markhasin, Angela Dai, Justus Thies, and Matthias Nießner. Diffuscene: Scene graph denoising diffusion probabilistic model for generative indoor scene synthesis. *arXiv preprint arXiv:2303.14207*, 2023.

[23] Despoina Paschalidou, Amlan Kar, Maria Shugrina, Karsten Kreis, Andreas Geiger, and Sanja Fidler. Atiss: Autoregressive transformers for indoor scene synthesis. *Advances in Neural Information Processing Systems*, 34:12013–12026, 2021.

[24] Pulak Purkait, Christopher Zach, and Ian Reid. Sg-vae: Scene grammar variational autoencoder to generate new indoor scenes. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIV 16*, pages 155–171. Springer, 2020.

[25] Haitao Yang, Zaiwei Zhang, Siming Yan, Haibin Huang, Chongyang Ma, Yi Zheng, Chandrajit Bajaj, and Qixing Huang. Scene synthesis via uncertainty-driven attribute synchronization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5630–5640, 2021.

[26] Ming-Jia Yang, Yu-Xiao Guo, Bin Zhou, and Xin Tong. Indoor scene generation from a collection of semantic-segmented depth images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15203–15212, 2021.

[27] Yinyu Nie, Angela Dai, Xiaoguang Han, and Matthias Nießner. Learning 3d scene priors with 2d supervision. *arXiv preprint arXiv:2211.14157*, 2022.

[28] Xinpeng Wang, Chandan Yeshwanth, and Matthias Nießner. Sceneformer: Indoor scene generation with transformers. In *2021 International Conference on 3D Vision (3DV)*, pages 106–115. IEEE, 2021.

[29] Daniel Ritchie, Kai Wang, and Yu-an Lin. Fast and flexible indoor scene synthesis via deep convolutional generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6182–6190, 2019.

[30] Kai Wang, Yu-An Lin, Ben Weissmann, Manolis Savva, Angel X Chang, and Daniel Ritchie. Planit: Planning and instantiating indoor scenes with relation graph and spatial prior networks. *ACM Transactions on Graphics (TOG)*, 38(4):1–15, 2019.

[31] Morgan Quigley, Brian Gerkey, and William D Smart. *Programming Robots with ROS: a practical introduction to the Robot Operating System*. " O'Reilly Media, Inc.", 2015.

[32] Jihoon Kim, Jiseob Kim, and Sungjoon Choi. Flame: Free-form language-based motion synthesis & editing. *arXiv preprint arXiv:2209.00349*, 2022.

[33] Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Amit H Bermano, and Daniel Cohen-Or. Human motion diffusion model. *arXiv preprint arXiv:2209.14916*, 2022.

[34] Mingyuan Zhang, Zhongang Cai, Liang Pan, Fangzhou Hong, Xinying Guo, Lei Yang, and Ziwei Liu. Motiondiffuse: Text-driven human motion generation with diffusion model. *arXiv preprint arXiv:2208.15001*, 2022.

[35] Zhiyuan Ren, Zhihong Pan, Xin Zhou, and Le Kang. Diffusion motion: Generate text-guided 3d human motion by diffusion model. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[36] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, October 2015.

[37] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6), November 2017.

[38] Silvia Zuffi, Angjoo Kanazawa, David Jacobs, and Michael J. Black. 3D menagerie: Modeling the 3D shape and pose of animals. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[39] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11097–11107, 2020.

[40] Anthony Dearden and Yiannis Demiris. Learning forward models for robots. In *IJCAI*, volume 5, page 1440, 2005.

[41] Jürgen Sturm, Christian Plagemann, and Wolfram Burgard. Adaptive body scheme models for robust robotic manipulation. In *Robotics: Science and systems*. Zurich, 2008.

[42] Jurgen Sturm, Christian Plagemann, and Wolfram Burgard. Unsupervised body scheme learning through self-perception. In *2008 IEEE International Conference on Robotics and Automation*, pages 3328–3333. IEEE, 2008.

[43] Jürgen Sturm, Vijay Pradeep, Cyrill Stachniss, Christian Plagemann, Kurt Konolige, and Wolfram Burgard. Learning kinematic models for articulated objects. In *Twenty-First International Joint Conference on Artificial Intelligence*. Citeseer, 2009.

[44] Jürgen Sturm, Cyrill Stachniss, and Wolfram Burgard. A probabilistic framework for learning kinematic models of articulated objects. *Journal of Artificial Intelligence Research*, 41:477–526, 2011.

[45] Neil Nie, Samir Yitzhak Gadre, Kiana Ehsani, and Shuran Song. Structure from action: Learning interactions for articulated object 3d structure discovery. *arXiv preprint arXiv:2207.08997*, 2022.

[46] Dov Katz and Oliver Brock. Manipulating articulated objects with interactive perception. In *2008 IEEE International Conference on Robotics and Automation*, pages 272–277. IEEE, 2008.

[47] Roberto Martin Martin and Oliver Brock. Online interactive perception of articulated objects with multi-level recursive estimation based on task-specific priors. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2494–2501. IEEE, 2014.

[48] Karol Hausman, Scott Niekum, Sarah Osentoski, and Gaurav S Sukhatme. Active articulation model estimation through interactive perception. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3305–3312. IEEE, 2015.

[49] Roberto Martín-Martín, Sebastian Höfer, and Oliver Brock. An integrated approach to visual perception of articulated objects. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 5091–5097. IEEE, 2016.

[50] Jeannette Bohg, Karol Hausman, Bharath Sankaran, Oliver Brock, Danica Kragic, Stefan Schaal, and Gaurav S Sukhatme. Interactive perception: Leveraging action in perception and perception in action. *IEEE Transactions on Robotics*, 33(6):1273–1291, 2017.

[51] Cheng-Chun Hsu, Zhenyu Jiang, and Yuke Zhu. Ditto in the house: Building articulation models of indoor scenes through interactive perception. *arXiv preprint arXiv:2302.01295*, 2023.

[52] Ruizhen Hu, Wenchao Li, Oliver Van Kaick, Ariel Shamir, Hao Zhang, and Hui Huang. Learning to predict part mobility from a single static snapshot. *ACM Transactions on Graphics (TOG)*, 36(6):1–13, 2017.

[53] Li Yi, Haibin Huang, Difan Liu, Evangelos Kalogerakis, Hao Su, and Leonidas Guibas. Deep part induction from articulated object pairs. *arXiv preprint arXiv:1809.07417*, 2018.

[54] Ben Abbatematteo, Stefanie Tellex, and George Konidaris. Learning to generalize kinematic models to novel objects. In *Proceedings of the 3rd Conference on Robot Learning*, 2019.

[55] Xiaolong Li, He Wang, Li Yi, Leonidas J Guibas, A Lynn Abbott, and Shuran Song. Category-level articulated object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3706–3715, 2020.

[56] Qihao Liu, Weichao Qiu, Weiyao Wang, Gregory D Hager, and Alan L Yuille. Nothing but geometric constraints: A model-free method for articulated object pose estimation. *arXiv preprint arXiv:2012.00088*, 2020.

[57] Yijia Weng, He Wang, Qiang Zhou, Yuzhe Qin, Yueqi Duan, Qingnan Fan, Baoquan Chen, Hao Su, and Leonidas J Guibas. Captra: Category-level pose tracking for rigid and articulated objects from point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13209–13218, 2021.

[58] Vicky Zeng, Tabitha Edith Lee, Jacky Liang, and Oliver Kroemer. Visual identification of articulated object parts. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2443–2450. IEEE, 2021.

[59] Yuki Kawana, Yusuke Mukuta, and Tatsuya Harada. Unsupervised pose-aware part decomposition for 3d articulated objects. *arXiv preprint arXiv:2110.04411*, 2021.

[60] Ajinkya Jain, Rudolf Lioutikov, Caleb Chuck, and Scott Niekum. Screwnet: Category-independent articulation model estimation from depth images using screw theory. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13670–13677. IEEE, 2021.

[61] Ajinkya Jain, Stephen Giguere, Rudolf Lioutikov, and Scott Niekum. Distributional depth-based estimation of object articulation models. In *Conference on Robot Learning*, pages 1611–1621. PMLR, 2022.

[62] Xueyi Liu, Ji Zhang, Ruizhen Hu, Haibin Huang, He Wang, and Li Yi. Self-supervised category-level articulated object pose estimation with part-level se (3) equivariance. *arXiv preprint arXiv:2302.14268*, 2023.

[63] Zihao Yan, Ruizhen Hu, Xingguang Yan, Luanmin Chen, Oliver Van Kaick, Hao Zhang, and Hui Huang. Rpm-net: recurrent prediction of motion and parts from point cloud. *arXiv preprint arXiv:2006.14865*, 2020.

[64] Ben Eisner, Harry Zhang, and David Held. Flowbot3d: Learning 3d articulation flow to manipulate articulated objects. *arXiv preprint arXiv:2205.04382*, 2022.

[65] Xianghao Xu, Yifan Ruan, Srinath Sridhar, and Daniel Ritchie. Unsupervised kinematic motion detection for part-segmented 3d shape collections. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–9, 2022.

[66] Xiaoxia Huang, Ian Walker, and Stan Birchfield. Occlusion-aware reconstruction and manipulation of 3d articulated objects. In *2012 IEEE International Conference on Robotics and Automation*, pages 1365–1371. IEEE, 2012.

[67] Aljaz Bozic, Pablo Palafox, Michael Zollhofer, Justus Thies, Angela Dai, and Matthias Nießner. Neural deformation graphs for globally-consistent non-rigid reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1450–1459, 2021.

[68] Gengshan Yang, Deqing Sun, Varun Jampani, Daniel Vlasic, Forrester Cole, Huiwen Chang, Deva Ramanan, William T Freeman, and Ce Liu. Lasr: Learning articulated shape reconstruction from a monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15980–15989, 2021.

[69] Gengshan Yang, Minh Vo, Natalia Neverova, Deva Ramanan, Andrea Vedaldi, and Hanbyul Joo. Banmo: Building animatable 3d neural models from many casual videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2863–2873, 2022.

[70] Jiteng Mu, Weichao Qiu, Adam Kortylewski, Alan Yuille, Nuno Vasconcelos, and Xiaolong Wang. A-sdf: Learning disentangled signed distance functions for articulated shape representation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13001–13011, 2021.

[71] Zhenyu Jiang, Cheng-Chun Hsu, and Yuke Zhu. Ditto: Building digital twins of articulated objects from interaction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5616–5626, 2022.

[72] Wei-Cheng Tseng, Hung-Ju Liao, Lin Yen-Chen, and Min Sun. Cla-nerf: Category-level articulated neural radiance field. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 8454–8460. IEEE, 2022.

[73] Fangyin Wei, Rohan Chabra, Lingni Ma, Christoph Lassner, Michael Zollhöfer, Szymon Rusinkiewicz, Chris Sweeney, Richard Newcombe, and Mira Slavcheva. Self-supervised neural articulated shape and appearance models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022.

[74] Jiahui Lei and Kostas Daniilidis. Cadex: Learning canonical deformation coordinate space for dynamic surface representation via neural homeomorphism. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[75] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 909–918, 2019.

[76] Xiaogang Wang, Bin Zhou, Yahao Shi, Xiaowu Chen, Qinping Zhao, and Kai Xu. Shape2motion: Joint analysis of motion parts and attributes from 3d shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8876–8884, 2019.

[77] Matt Deitke, Winson Han, Alvaro Herrasti, Aniruddha Kembhavi, Eric Kolve, Roozbeh Mottaghi, Jordi Salvador, Dustin Schwenk, Eli VanderBilt, Matthew Wallingford, et al. Robothor: An open simulation-to-real embodied ai platform. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3164–3174, 2020.

[78] Bokui Shen, Fei Xia, Chengshu Li, Roberto Martín-Martín, Linxi Fan, Guanzhi Wang, Claudia Pérez-DArpino, Shyamal Buch, Sanjana Srivastava, Lyne Tchapmi, et al. igibson 1.0: A simulation environment for interactive tasks in large realistic scenes. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7520–7527. IEEE, 2021.

[79] Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, et al. Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in Neural Information Processing Systems*, 34:251–266, 2021.

[80] Haoran Geng, Ziming Li, Yiran Geng, Jiayi Chen, Hao Dong, and He Wang. Partmanip: Learning cross-category generalizable part manipulation policy from point cloud observations. *arXiv preprint arXiv:2303.16958*, 2023.

[81] Haoran Geng, Helin Xu, Chengyang Zhao, Chao Xu, Li Yi, Siyuan Huang, and He Wang. Gapartnet: Cross-category domain-generalizable object perception and manipulation via generalizable and actionable parts. *arXiv preprint arXiv:2211.05272*, 2022.

[82] Liu Liu, Wenqiang Xu, Haoyuan Fu, Sucheng Qian, Qiaojun Yu, Yang Han, and Cewu Lu. Akb-48: a real-world articulated object knowledge base. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14809–14818, 2022.

[83] Siddhartha Chaudhuri, Daniel Ritchie, Jiajun Wu, Kai Xu, and Hao Zhang. Learning generative models of 3d structures. In *Computer Graphics Forum*, volume 39, pages 643–666. Wiley Online Library, 2020.

[84] Mohammad Amin Shabani, Sepidehsadat Hosseini, and Yasutaka Furukawa. Housediffusion: Vector floorplan generation via a diffusion model with discrete and continuous denoising. *arXiv preprint arXiv:2211.13287*, 2022.

[85] Jason Yim, Brian L Trippe, Valentin De Bortoli, Emile Mathieu, Arnaud Doucet, Regina Barzilay, and Tommi Jaakkola. Se (3) diffusion model with application to protein backbone generation. *arXiv preprint arXiv:2302.02277*, 2023.

[86] Wenqi Fan, Chengyi Liu, Yunqing Liu, Jiatong Li, Hang Li, Hui Liu, Jiliang Tang, and Qing Li. Generative diffusion models on graphs: Methods and applications. *arXiv preprint arXiv:2302.02591*, 2023.

[87] Zhen Liu, Yao Feng, Michael J Black, Derek Nowrouzezahrai, Liam Paull, and Weiyang Liu. Meshdiffusion: Score-based generative 3d mesh modeling. *arXiv preprint arXiv:2303.08133*, 2023.

[88] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

[89] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. *arXiv preprint arXiv:2211.10440*, 2022.

[90] Miguel Angel Bautista, Pengsheng Guo, Samira Abnar, Walter Talbott, Alexander Toshev, Zhuoyuan Chen, Laurent Dinh, Shuangfei Zhai, Hanlin Goh, Daniel Ulbricht, et al. Gaudi: A neural architect for immersive 3d scene generation. *Advances in Neural Information Processing Systems*, 35:25102–25116, 2022.

[91] Congyue Deng, Chiyu Jiang, Charles R Qi, Xinchen Yan, Yin Zhou, Leonidas Guibas, Dragomir Anguelov, et al. Nerdi: Single-view nerf synthesis with language-guided diffusion as general image priors. *arXiv preprint arXiv:2212.03267*, 2022.

[92] Ziya Erkoç, Fangchang Ma, Qi Shan, Matthias Nießner, and Angela Dai. Hyperdiffusion: Generating implicit neural fields with weight-space diffusion. *arXiv preprint arXiv:2303.17015*, 2023.

[93] Ye Yuan, Jiaming Song, Umar Iqbal, Arash Vahdat, and Jan Kautz. Physdiff: Physics-guided human motion diffusion model. *arXiv preprint arXiv:2212.02500*, 2022.

[94] Jonathan Tseng, Rodrigo Castellon, and C Karen Liu. Edge: Editable dance generation from music. *arXiv preprint arXiv:2211.10658*, 2022.

[95] Simon Alexanderson, Rajmund Nagy, Jonas Beskow, and Gustav Eje Henter. Listen, denoise, action! audio-driven motion synthesis with diffusion models. *arXiv preprint arXiv:2211.09707*, 2022.

[96] Siyuan Huang, Zan Wang, Puhao Li, Baoxiong Jia, Tengyu Liu, Yixin Zhu, Wei Liang, and Song-Chun Zhu. Diffusion-based generation, optimization, and planning in 3d scenes. *arXiv preprint arXiv:2301.06015*, 2023.

[97] Yonatan Shafir, Guy Tevet, Roy Kapon, and Amit H Bermano. Human motion diffusion as a generative prior. *arXiv preprint arXiv:2303.01418*, 2023.

[98] Sigal Raab, Inbal Leibovitch, Guy Tevet, Moab Arar, Amit H Bermano, and Daniel Cohen-Or. Single motion diffusion. *arXiv preprint arXiv:2302.05905*, 2023.

[99] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

[100] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

[101] Weiyu Liu, Tucker Hermans, Sonia Chernova, and Chris Paxton. Structdiffusion: Object-centric diffusion for semantic rearrangement of novel objects. *arXiv preprint arXiv:2211.04604*, 2022.

[102] Roberto Martín-Martín, Clemens Eppner, and Oliver Brock. The rbo dataset of articulated objects and interactions. *The International Journal of Robotics Research*, 38(9):1013–1019, 2019.

[103] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019.

[104] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019.

[105] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[106] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11461–11471, 2022.

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

[108] Gengxin Liu, Qian Sun, Haibin Huang, Chongyang Ma, Yulan Guo, Li Yi, Hui Huang, and Ruizhen Hu. Semi-weakly supervised object kinematic motion prediction. *arXiv preprint arXiv:2303.17774*, 2023.

[109] Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, and Stefano Ermon. Permutation invariant graph generation via score-based generative modeling. In *International Conference on Artificial Intelligence and Statistics*, pages 4474–4484. PMLR, 2020.

[110] Jaehyeong Jo, Seul Lee, and Sung Ju Hwang. Score-based generative modeling of graphs via the system of stochastic differential equations. In *International Conference on Machine Learning*, pages 10362–10383. PMLR, 2022.

[111] Clement Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. Digress: Discrete denoising diffusion for graph generation. *arXiv preprint arXiv:2209.14734*, 2022.

[112] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4541–4550, 2019.

[113] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019.