GMAP: Generalized Manipulation of Articulated Objects in Robotic Using Pre-trained Model

Hongliang Zeng, Ping Zhang*, Fang Li, Qinpeng Yi, Tingyu Ye, Jiahua Wang

South China University of Technology
scutzenghongl@gmail.com, {pzhang, cslifang}@scut.edu.cn

Abstract

Perception and interaction with articulated objects present a unique challenge for service robots. Although recent research has emphasized understanding articulated shapes and affordance proposals, existing methods only address isolated aspects, failing to develop comprehensive strategies for robotic perception and manipulation of articulated objects. To bridge this gap, we propose GMAP, which systematically integrates the entire process from command to perception and manipulation. Specifically, we first perform precise part-level segmentation of the object and identify the geometric and kinematic parameters of articulated joints. Then, by evaluating point-level affordance proposals, we determine the interaction poses for the robot's end-effector. Finally, the robot's execution trajectory is dynamically computed by combining commands with joint parameters and interaction points. Additionally, a key innovation of GMAP is addressing the scarcity of annotated data. We designed a multi-scale point cloud feature extraction module and introduced pre-training and fine-tuning techniques, significantly enhancing the generalization capability of the perception model. Extensive experiments demonstrate that GMAP achieves state-of-the-art (SOTA) performance in both the perception and manipulation of articulated objects and adapts to real-world scenarios.

Code — https://github.com/robhlzeng/GMAP

Introduction

Manipulating articulated objects (Mo et al. 2021; Wang et al. 2022; Chu et al. 2023; Zeng et al. 2024b) is a common and crucial task in the field of service robots. We aim for agents to autonomously determine when and how to perform manipulation tasks. In recent years, large language models (LLMs) (Devlin et al. 2018; Brown et al. 2020; Touvron et al. 2023) have made significant progress across various fields, providing robust support for embodied intelligence (Joublin et al. 2023; Rana et al. 2023; Zhang et al. 2023; Wang et al. 2024; Hazra, Dos Martires, and De Raedt 2024) by translating abstract high-level intents into specific task instructions. For example, when a user expresses the desire to "drink a cold soda", an LLM can parse this into a series of concrete actions: moving to the refrigerator, opening the door, and retrieving the chilled soda. In this process, the LLM acts as the "brain" of the agent, completing the thought process from intent to action. However, to achieve precise skill execution, the agent also needs a "cerebellum" to plan and coordinate movements. Therefore, our goal is to develop a general perception-to-interaction manipulation strategy specifically for articulated object tasks, making it a robust component of the agent's "cerebellum".

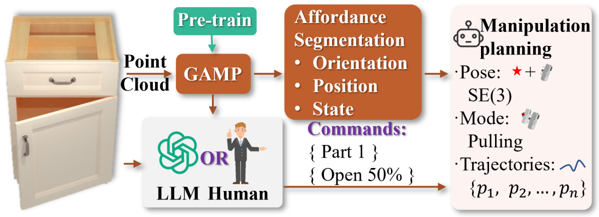

Figure 1: GMAP segments the movable parts of an object, estimates the joint parameters and manipulation affordances, and ultimately plans the operation based on commands.

Although recent research has made progress in estimating joint parameters (Zeng et al. 2024b) of articulated parts, building digital twins (Jiang, Hsu, and Zhu 2022), and visualizing manipulation affordances (Wang et al. 2022), these technologies often focus on a single aspect and fail to meet the practical needs. For example, when an LLM or a human instructs a robot to open a specific part of a cabinet to a degree, understanding the joint parameters of that part is insufficient. The robot needs to comprehend how to locate contact points, adopt poses, and follow a specific trajectory to meet the instruction's requirements. Therefore, we propose GMAP to address this challenge. Our approach integrates part segmentation, joint parameter estimation, and affordance analysis, with a designed multi-scale point cloud feature extraction network and training strategy to enhance the performance of each module. As shown in Fig. 1, GMAP combines operation commands with the joint parameters and affordance features of the part, dynamically planning an operation trajectory to achieve the desired state.

On the other hand, there is a significant contradiction between the high demand for generalization and accuracy in perception models and the scarcity of high-quality annotated

*Corresponding author

data. Previous research (Li et al. 2020; Zeng et al. 2024b) has tended to develop category-level perception models to mitigate data dependence. However, this approach hampers the adaptability of the models, making them challenging to deploy in real-world scenarios. In this study, we explore two approaches to optimize this issue. Firstly, considering the size differences between articulated objects, a single-scale feature extractor struggles to adapt to different categories. Therefore, we partition the point cloud into multiple scales and employ a visual transformer (Vit) (Dosovitskiy et al. 2020) encoder to extract features at each scale. Subsequently, multi-scale features are integrated and propagated to obtain point-wise features of the input point cloud for further inference. This approach significantly enhances the model's ability to capture both global and local features. Secondly, we employ self-supervised learning (SSL) to leverage large-scale unannotated datasets, thereby improving the model's understanding of a wide range of 3D shapes. On this foundation, we perform post-training and fine-tuning on the articulated object dataset to achieve additional performance improvements.

We conducted comprehensive experimental validation on two widely recognized articulated object datasets—PartNet-Mobility (Mo et al. 2019) and Shape2Motion (Wang et al. 2019). GMAP achieved an 80% Intersection over Union (IoU) in the part segmentation task, while in the joint orientation prediction task, the prediction error was maintained at approximately 0.42°. Additionally, we achieved a 36.94% success rate in instruction-based push manipulation in the Sapien (Xiang et al. 2020) simulator and successfully manipulated three different types of articulated objects in real-world environments. In summary, the contributions of this work can be outlined as follows:

- GMAP proposes an innovative multi-scale point cloud feature extraction model, enhanced through pre-training. It achieves SOTA performance in segmentation and joint parameter estimation tasks.
- We constructed a comprehensive process from perception to interaction, enabling robots to perform general-level manipulation planning for articulated objects.
- Experimental results demonstrate that GMAP achieves leading success rates in manipulation tasks and can adapt to complex real-world environments.

Related Works

Pre-training for Point Cloud. Due to the relative scarcity of large-scale, high-quality annotated 3D datasets, pre-training methods have garnered widespread attention in the field of point cloud feature extraction. Following contrastive learning methods (Xie et al. 2020; Zhang et al. 2021), inspired by the success of masked modeling in natural language processing (NLP) (Devlin et al. 2018) and 2D computer vision (CV) (He et al. 2022), masked point block modeling (MPM) (Yu et al. 2022; Pang et al. 2022; Zhang et al. 2022) has gradually become the mainstream paradigm. PointGPT (Chen et al. 2023) addresses the issue of global shape information leakage by using an autoregressive generation mechanism. Point-MGE (Zeng et al. 2024a) introduces generative learning into self-supervised training, achieving SOTA pre-training performance. Our pre-training method is based on Point-MGE, as it helps us overcome point cloud sampling bias, which is crucial. Subsequently, we added a fine-tuning phase to enable the model to gain a deeper understanding of the 3D shape features of articulated objects.

Articulated Part Analysis. Simulators (Todorov, Erez, and Tassa 2012; Xiang et al. 2020; Szot et al. 2021) and open datasets (Chang et al. 2015; Mo et al. 2019; Wang et al. 2019; Geng et al. 2023) play a crucial role in advancing the analysis of articulated objects. Previous studies (Yi et al. 2018; Abbatematteo, Tellex, and Konidaris 2019; Shi, Cao, and Zhou 2021; Jain et al. 2021; Siarohin et al. 2021; Jiang, Hsu, and Zhu 2022) have achieved joint parameter estimation using a two-stage dynamic observation as input, but in most real-world scenarios, robots can only rely on static input. MARS (Zeng et al. 2024b) enhances feature extraction through multimodal input but simultaneously increases the complexity of data collection and the strictness of data alignment. Cart (Chu et al. 2023) encodes commands as model input and predicts the dynamic parameters for manipulation; however, it still lacks the capability to directly drive the robot to manipulate articulated objects. It is noteworthy that these methods are mostly trained on small sample datasets, leading to insufficient generalization capability. In contrast, our method leverages point cloud SSL to enhance the generalization ability of the few-shot learning model and can plan manipulation trajectories based on perception results.

Learning Interactive Affordance. The core of learning interactive affordances lies in deeply analyzing the interaction processes between other agents and the environment to identify and understand potential action strategies and interaction patterns (Redmon and Angelova 2015; Qin et al. 2020a; Nagarajan and Grauman 2020; Wu et al. 2021; Xu, He, and Song 2022). In this field, researchers have developed various visual affordance representation techniques, including detecting key parts of objects (Mandikal and Grauman 2021), identifying critical contact points during interactions (Qin et al. 2020b), and generating heatmaps to guide interactive actions (Nagarajan and Grauman 2020; Mo et al. 2021; Wang et al. 2022). While these visual affordance representations perform well in simple interaction tasks, they fall short in completing the manipulation of complex articulated objects. A key challenge is generating a coherent execution trajectory to accurately complete specific operation commands. To address this issue, our research combines interactive affordance representation with joint parameter estimation and calculates the transformation matrix required for the manipulation point to reach the target state.

Method

In this section, we will introduce the components and pipeline of the GMAP framework (see Fig. 2). GMAP starts with a pre-trained multi-scale feature extraction (MSFE) module, followed by point-wise feature extraction (PFE) through feature propagation. This framework comprises three core perception modules: the part segmentation network (Seg-Net), the joint parameter estimation network

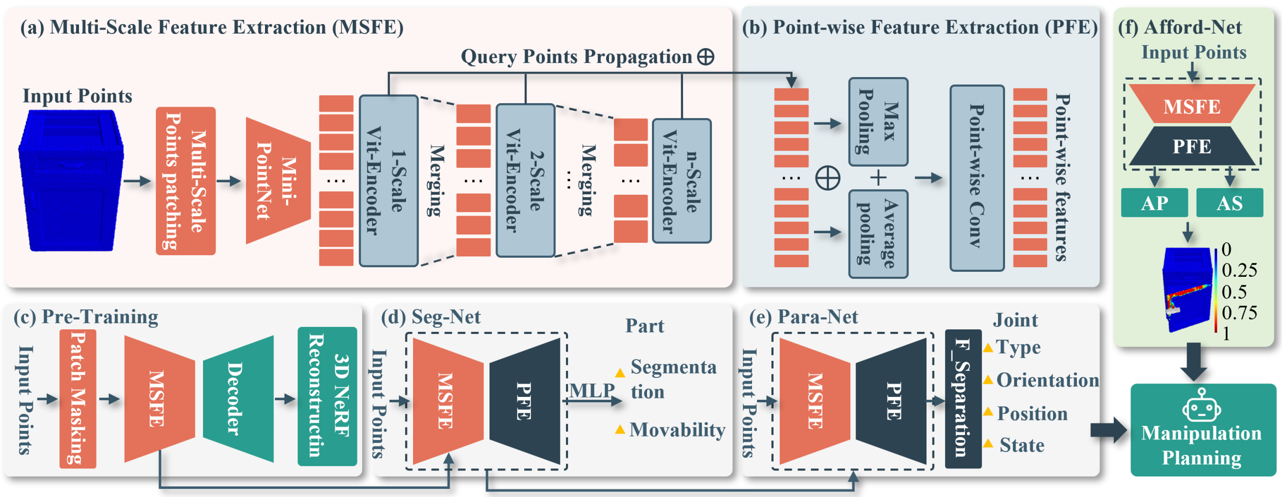

Figure 2: **Overview of GMAP framework.** (a) **MSFE:** The input points are processed into multi-scale point patches (see Fig. 3) and rich feature representations are extracted through ViT encoders. (b) **PFE:** Obtain query point features and perform multi-scale feature interpolation using propagation techniques. (c) **Pre-Training:** Utilizes point cloud SSL for effective pre-training of the MSFE module. (d) **Seg-Net:** Combines MSFE and PFE for part segmentation and mobility prediction. (e) **Para-Net:** Estimates the joint type, direction, position, and state of selected parts. (f) **Afford-Net:** Estimates point-level affordances to guide interaction planning.

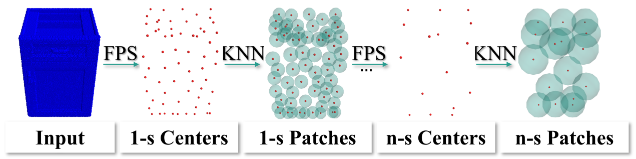

Figure 3: The points is divided into multi-scale patches using multi-level FPS and KNN.

(Para-Net), and the point-level manipulation affordance proposal network (Afford-Net).

Multi-Scale Feature Extraction

For the points $X \in \mathbb{R}^{N\times 3}$, we employ the MSFE module depicted in Fig. 2(a) to extract its multi-scale features. This process encompasses three critical steps: normalizing the point cloud; patching it at multiple scales; and encoding through a series of ViT encoders.

Normalizing. To accommodate objects of different scales, we establish a coordinate system with the center point of X. Then, we translate and scale X as follows:

$$X' = \frac{X - \frac{1}{N}\sum_{i=1}^{N} x_i}{\max_i \|x_i - \frac{1}{N}\sum_{j=1}^{N} x_j\|} \tag{1}$$

Patching. As shown in Fig. 3, for any scale i, we employ Farthest Point Sampling (FPS) to obtain $m_i \in M$ center points C_i, followed by the K-Nearest Neighbors (KNN) algorithm to determine $k_i \in K$ neighboring points for each patch P_i. This can be represented as:

$$P_i = \text{KNN}(\text{FPS}(C_{i-1}), C_{i-1}), \quad P_i \in \mathbb{R}^{m_i\times k_i\times 3}, \tag{2}$$

where $C_0 = X'$, the number of scales n, and the values of M and K are predefined.

Encoding. We employ a mini-PointNet (Qi et al. 2017a) to extract preliminary features of the point patch at the first scale and use a Multi-Layer Perceptron (MLP) for position encoding of the center points. The input tokens T_1 for the ViT encoder at the first scale are:

$$T_1 = \text{Mini-PointNet}(P_1) + \theta_{enc}(C_1), \quad T_1 \in \mathbb{R}^{m_1\times d_1}. \tag{3}$$

For the i-th scale, the input tokens T_i are obtained by merging the features of the previous scale contained in P_i:

$$T_i = \mathcal{M}(\theta_{merge}(T_{i-1}^{k_i})) + \theta_{enc}(C_i), \quad T_i \in \mathbb{R}^{m_i\times d_i}, \tag{4}$$

where $\mathcal{M}(\cdot)$ denotes the maximum pooling operation and d_i is the feature dimension.

Point-wise Feature Extraction

The PFE module first treats each point in the point set $Y \in \mathbb{R}^{N'\times 3}$ as a query. Based on this, the module employs a feature propagation technique similar to that in PointNet++ (Qi et al. 2017b), integrating the output tokens $\hat{T}_i$ from ViT encoders at different scales to obtain point-wise features f:

$$f = \bigoplus_{i=1}^{n}(\mathcal{P}(\hat{T}_i)), \quad f \in \mathbb{R}^{N\times \sum_{i=1}^{n} d_i}, \tag{5}$$

where $\bigoplus$ and $\mathcal{P}(\cdot)$ denote feature concatenate and feature propagation, respectively. Then, we perform max pooling and average pooling on the feature f, and concatenate the sum of these pooled results with the original feature. Finally, we employ multi-layer 1D convolutions to reduce the dimensionality of the features, which can be expressed as:

$$F = \theta_{\text{conv}}((\mathcal{M}(f) + \mathcal{A}(f)) \bigoplus f)), \quad F \in \mathbb{R}^{N\times D}. \tag{6}$$

Pre-training

We adhere to the Point-MGE (Zeng et al. 2024a) by employing a two-stage strategy to pre-train our MSFE. Firstly, we train a Vector Quantized Variational Autoencoder (VQ-VAE) (Van Den Oord, Vinyals et al. 2017) to map point patch features into a discrete codebook. Subsequently, we apply variable masking to the point patches and then input them into the MSFE module. The decoder is responsible for reconstructing the 3D shape based on the visible parts. Further details can be found in (Zeng et al. 2024a) and appendix.

Seg-Net

We utilize the pre-trained MSFE to extract multi-scale features from the input points X, and employ a PFE module to propagate features for each point $y \in Y$. This design allows Seg-Net to extract features from the sparse points X and perform segmentation on the dense points Y to save computational cost, where $N \ll N'$. Building on this foundation, we deploy two MLP prediction heads to predict the part to which each point belongs and whether the point is movable,

$$\tau = \theta_{mov}\left(F\right) \text{ and } \gamma = \theta_{seg}\left(F\right), \ \ \tau = \{0, 1\}, \tag{7}$$

where $\tau = 0$ means the part is immovable and $\tau = 1$ means the part is movable.

optimization target. For the prediction of movability, we employ binary cross-entropy (BCE) as the optimization objective for binary classification. For the segmentation task, we also utilize the cross-entropy loss function. The combined training objective can be represented as follows:

$$\mathcal{L}_{\text{seg}} = \mathcal{L}_{\text{bce}}(\tau, \hat{\tau}) + \mathcal{L}_{\text{ce}}(\gamma, \hat{\gamma}), \tag{8}$$

where $\hat{\tau}$ and $\hat{\gamma}$ denote the ground truth for mobility as well as part segmentation, respectively.

Para-Net

Feature Separator. After training the Seg-Net, we transfer the weights of the MSFE and PFE as a whole to the Para-Net. The network selects a movable part κ based on the segmentation results and command requirements, and predicts the joint type ρ, orientation o, position u, and current state s of the selected part. After obtaining point-level features, we use a feature separator to aggregate features belonging to points of the part and features of points not belonging to the part, respectively. These two sets of features are then concatenated to serve as inputs to different MLP prediction heads. The feature separation process can be represented as:

$$F_\kappa = \mathcal{A}(F|\gamma = \kappa) \bigoplus \mathcal{A}(F|\gamma \neq \kappa), \tag{9}$$

$$\{\rho, o, u, s\} = \theta_{para}\left(F_\kappa\right), \ \ \rho = \{0, 1\}. \tag{10}$$

optimization target. We use binary cross-entropy loss $\mathcal{L}_{\text{type}}$ to estimate the type of joint ρ, where $\rho = 0$ represents a revolute joint and $\rho = 1$ represents a prismatic joint. Since the position of prismatic joints does not affect the planning of the manipulation trajectory, we only compute the loss for the position estimation of revolute joints. This loss is defined as the projection distance between the estimated position and the true joint axis:

$$\mathcal{L}_{\text{pos}}(u, \hat{u}, \hat{o}) = \|u - \hat{u} - (\frac{u - \hat{u}}{\|u - \hat{u}\|} \cdot \hat{o})\|_2, \tag{11}$$

where $\hat{u}$ and $\hat{o}$ represent the true position and the unit orientation vector of the joint axis, respectively. Additionally, we calculate the arccos value of the angle between o and $\hat{o}$, and minimize this value as the orientation training objective:

$$\mathcal{L}_{\text{ori}}(o, \hat{o}) = \arccos(o, \hat{o}). \tag{12}$$

Finally, we employ the L1 norm loss to calculate the discrepancy between the estimated joint state s and the true state $\hat{s}$ as the loss function $\mathcal{L}_{\text{state}}$. Specifically, similar to Cart (Chu et al. 2023), we normalize the state $s \in [0, 1]$. The overall optimization objective can be represented as:

$$\mathcal{L}_{\text{para}} = \mathcal{L}_{\text{type}} + (1 - \rho) \cdot \mathcal{L}_{\text{pos}} + \mathcal{L}_{\text{ori}} + \mathcal{L}_{\text{state}}. \tag{13}$$

Afford-Net

As shown in Fig. 2(f), the GMAP framework evaluates point-wise action affordance using two MLP prediction heads—Action Proposal (AP) and Action Scoring (AS). Similarly, MSFE and PFE extract point-level features from the object points Y as inputs for the prediction heads. The objective of AP is to determine the interaction parameters R_z between the robot's end-effector and the manipulation point with a high recall rate. It takes point-level features and randomly sampled Gaussian noise z as inputs to predict the 3-DoF orientation of the end-effector in the $SO(3)$ space:

$$R_z = \theta_{\text{AP}}(F, z, a^t), \ \ R_z \in \mathcal{R}^{N' \times 3 \times 3}, \tag{14}$$

where a^t denotes the action type, with $a^t = 0$ representing a push action and $a^t = 1$ representing a pull action. Following the Where2Act (Mo et al. 2021), we collected extensive interaction data samples $\{(S_i, p_i, R_i, a_i^t, r_i)\}_i$ in the simulator for training. Here, $p|S$ represents a point p on the sample S, R is the end-effector orientation, and $r = \{0, 1\}$ is used to distinguish between positive and negative samples. The Min-of-N (Fan, Su, and Guibas 2017) loss function for AP module is defined as:

$$\mathcal{L}_{\text{AP}} = r \cdot \min_{j=1,\ldots,100} \text{dist}(R_{z_j}, R). \tag{15}$$

Additionally, we use the standard binary cross-entropy loss to train the AS head, which is defined as:

$$a_s = \theta_{\text{AS}}(F, R, a^t), \ \ a_s = \{0, 1\}, \tag{16}$$

$$\begin{aligned} \mathcal{L}_{\text{AS}} = \, & r \cdot \log(\theta_{\text{AS}}(F_{p|S}, R, a^t)) + \\ & (1 - r) \cdot \log(1 - \theta_{\text{AS}}(F_{p|S}, R, a^t)) \end{aligned} \tag{17}$$

Trajectory Planning

Trajectory planning selects the point p with the highest action score for interaction. It calculates motion parameters using predicted joint parameters $\{\rho, o, u, s\}$ and command requirements $\{p_{id}, s'\}$. The required rotation angle or movement distance is determined by comparing the difference between the current predicted state of the joint and the desired

Datasets	Methods	Segmentation (mIoU%)↑	Joint Parameter				
			Revolute			Prismatic	
			Ori ↓	Pos ↓	State ↓	Ori ↓	State ↓
PartNet-Mobility (Mo et al. 2019)	RPM-Net (Yan et al. 2020)	59.7	10.36°	0.16	16.15°	10.19°	0.27
	ANCSH (Li et al. 2020)	-	4.15°	0.08	12.14°	7.74°	0.24
	Ditto (Jiang, Hsu, and Zhu 2022)	-	1.88°	0.04	5.81°	2.77°	0.08
	Cart (Chu et al. 2023)	72.3	1.81°	0.04	4.96°	2.52°	0.08
	MARS (Zeng et al. 2024b)	-	1.58°	**0.03**	3.74°	0.87°	0.09
	Ours	**79.6**	**0.45°**	0.04	**0.21°**	**0.35°**	**0.05**
	w/o Pre-training	77.0	1.06°	0.05	0.22°	1.65°	**0.05**
Shape2Motion (Wang et al. 2019)	RPM-Net (Yan et al. 2020)	61.2	9.85°	0.13	14.28°	11.37°	0.31
	ANCSH (Li et al. 2020)	-	3.71°	0.12	11.72°	7.92°	0.26
	Ditto (Jiang, Hsu, and Zhu 2022)	-	1.74°	0.05	5.94°	3.04°	0.09
	Cart (Chu et al. 2023)	70.9	1.96°	0.04	4.87°	2.58°	0.08
	MARS (Zeng et al. 2024b)	-	1.43°	0.04	2.95°	1.94°	0.08
	Ours	**81.4**	**0.39°**	**0.03**	**0.27°**	**0.49°**	**0.04**
	w/o Pre-training	77.6	0.94°	0.05	0.38°	1.48°	0.05

Table 1: Quantitative evaluation results of segmentation and joint parameters on the PartNet-Mobility (Mo et al. 2019) and Shape2Motion (Wang et al. 2019) datasets. The best results are shown in bold, and the second-best results are underlined.

	Methods	Train	Test				
		cabinet	Oven	Fridge	Table	Micro	Safe
Ori ↓	MARS	3.17°	15.97°	4.83°	7.49°	6.26°	12.31°
	Ours	0.36°	**5.91°**	**0.42°**	**0.16°**	**2.41°**	**2.55°**
	w/o Pre	1.43°	13.08°	0.67°	8.61°	5.78°	18.98°
Pos ↓	MARS	0.19	0.16	0.21	0.15	0.08	0.09
	Ours	0.03	**0.08**	0.05	**0.08**	**0.05**	**0.06**
	w/o Pre	0.03	0.12	**0.04**	**0.08**	0.06	0.11
State ↓	MARS	0.09	2.74°	6.21°	3.42°	4.94°	3.15°
	Ours	0.24°	0.13°	**0.42°**	**0.04°**	**0.15°**	0.61°
	w/o Pre	0.07	**0.11°**	0.49°	0.09°	0.23°	**0.49°**

Table 2: Trained on cabinet and tested on five unseen categories to evaluate the method's generalization performance.

state, $l = (s' - s) \cdot L$, where L represents the extreme value when the state of the joint is $s = 1$. Based on this, we compute the movement trajectory of the manipulation point and obtain the motion parameters of the end-effector in SE(3) space through interpolation over multiple time steps. For prismatic joints, the motion trajectory is:

$$p(t) = p + t \cdot (l \cdot o), \quad t \in [0, 1]. \tag{18}$$

Using the Rodriguez formula to calculate the rotation matrix $R(\cdot)$, the motion trajectory of the revolute joint is:

$$p(t) = R(o, l \cdot t)(p - u) + u, \quad t \in [0, 1], \tag{19}$$

Experimental

In this section, we detail the experimental setup and demonstrate GMAP's performance in simulated and real-world conditions. The network model's training process and hyperparameter setting are documented in the Appendix.

Experimental Setup

Datasets. We evaluated our method on two recognized articulated object datasets, PartNet-Mobility (Mo et al. 2019) and Shape2Motion (Wang et al. 2019), comparing it to benchmarks. We chose 14 object categories from PartNet-Mobility and 4 from Shape2Motion, each with over 30 instances. Each instance was imported into the simulator, with joint states of movable parts adjusted randomly 20 times. Using a depth camera, we collected point cloud data in various states, dividing it 9:1:1 for training, validation, and testing. We also tested our method's manipulation planning on PartNet-Mobility instances in the Sapien (Xiang et al. 2020).

Pre-training. We pre-trained the MSFE module on ShapeNet (Chang et al. 2015) to boost its 3D shape feature extraction. This dataset offers over 50,000 3D models across 55 categories. For MSFE, we set three scales with point patches $M = \{512, 256, 64\}$ and points per patch $K = \{32, 8, 8\}$. Using AdamW (Loshchilov and Hutter 2017), we pre-trained for 300 epochs as Point-MGE (Zeng et al. 2024a). To deepen the model's grasp of articulated objects, we added 100 epochs post-training on an articulated object dataset.

Baselines. We compared six baseline methods for part segmentation and joint parameter estimation: RPM-Net (Yan et al. 2020), ANCSH (Li et al. 2020), Ditto (Jiang, Hsu, and Zhu 2022), Cart (Chu et al. 2023), MARS (Zeng et al. 2024b), and a GMAP version without pre-training. Ditto creates digital twins via a two-stage strategy. Cart estimates parameters based on commands. MARS is the SOTA in joint parameter estimation prior to our work. For manipulation, we also compared Where2Act (Mo et al. 2021), which uses affordance estimation, and its benchmarks and an improved version, AdaAfford (Wang et al. 2022). These methods, unlike GMAP, mainly perform fixed actions and

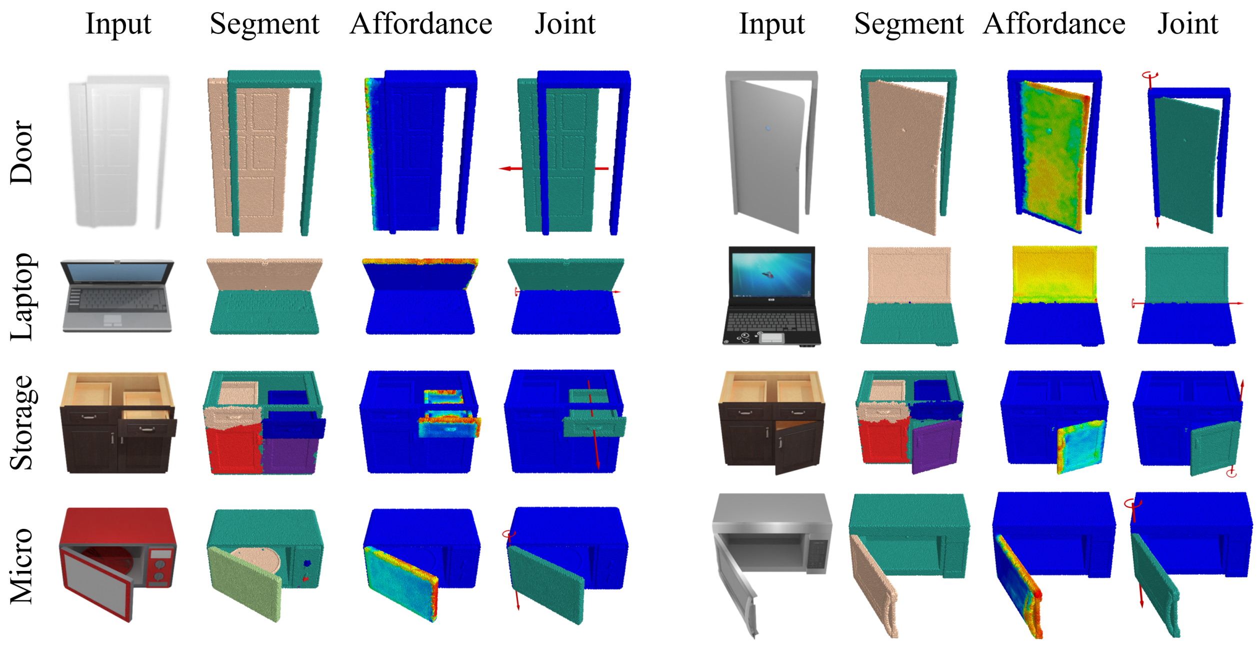

Figure 4: We present the visualization results of segmentation, joint parameter estimation, and affordance proposals for eight instances across four categories.

Methods	Pushing-Succ(%) ↑	Pulling-Succ(%) ↑
B-Random	3.79	1.55
B-Normal	11.57	4.18
B-PCPNet	9.15	3.71
Where2Act	14.74	7.85
AdaAfford	27.71	10.45
Our-Afford	18.23	9.39
Our	**36.94**	**21.72**

Table 3: Evaluation of manipulation results for sampled interaction points. "Our-Afford" uses GMAP's affordance module for action suggestions, applying predefined "push" and "pull" without joint parameters for trajectory planning.

lack dynamic planning, potentially limiting complex task handling.

Main Results

Segmentation and Joint Parameters Estimation. As detailed in Table 1, our approach excels in part segmentation on both PartNet-Mobility (Mo et al. 2019) and Shape2Motion (Wang et al. 2019) datasets, with IoU scores of 79.6% and 81.4%, leading all compared methods. In joint parameter estimation, while our method narrowly falls behind MARS (Zeng et al. 2024b) in the specific task of estimating revolute joint positions on PartNet-Mobility, it surpasses it in the remaining key metrics. Our method particularly excels in rotational joint state estimation, outperforming MARS by 3.53° and 2.68° on the respective datasets. The pre-training strategy significantly enhances GMAP's performance across all metrics, outperforming the non-pre-trained version. This highlights the importance of pre-training. Even without it, GMAP maintains a substantial lead over other baselines in both tasks, affirming the efficacy of our multi-scale feature extraction framework for articulated objects.

Generalizability results. We assessed GMAP's generalization by training Para-Net on the cabinet category and validating it on five new categories. Despite facing unseen categories, our method only shows a minor dip in performance, as illustrated in Table 2, particularly in joint orientation estimation, where it remains competitive with MARS (Zeng et al. 2024b) and our ablation model. This robust generalization is due to two main factors: our implementation of a resilient multi-scale feature extraction network that captures universal 3D shape features, and a pre-training strategy that equips the model to understand and apply cross-category 3D shape principles, ensuring consistent accuracy with unfamiliar categories.

Manipulation results. After adeptly accomplishing part segmentation and joint parameter estimation, we proceeded to assess GMAP's proficiency in carrying out manipulation commands. While benchmark methods like Where2Act (Mo et al. 2021) and AdaAfford (Wang et al. 2022) rely on manually selected "push" or "pull" models aligned with specific command requirements, GMAP distinguishes itself by intelligently selecting actions based on the discrepancy between the joint's current and desired states. To isolate and exhibit the effectiveness of GMAP's individual components,

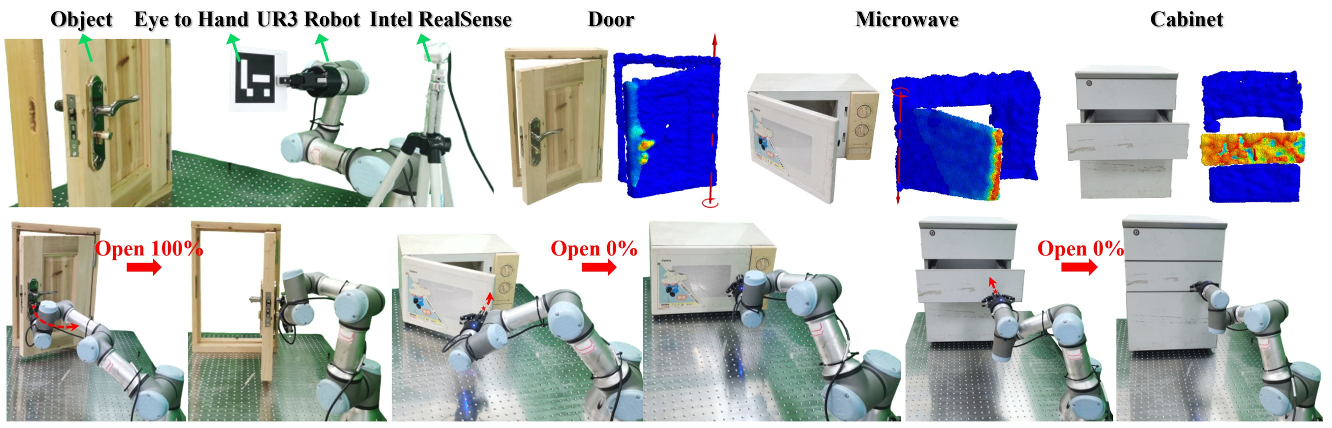

Figure 5: Real-world robot experiments. The upper left corner of the figure shows our experimental setup, while the upper right corner provides detailed visualizations of the joint parameter estimations and affordance proposals for the input objects. The images below compare the start and end keyframes of the manipulation process.

we introduced an ablation study utilizing solely the Afford-Net. This version aligns with the benchmarks by employing predefined motion primitives for action proposals. Yet, GMAP transcends this by dynamically generating and executing motion trajectories informed by the joint parameters and command specifics.

As depicted in Table 3, adhering to where2act's protocol, we documented the success rates from interaction point tests across five categories. The findings reveal that Our-Afford, limited to the Afford-Net, markedly enhances proposal quality over Where2Act, attributable to its refined point-level feature analysis. Despite this, Our-Afford slightly lags behind AdaAfford, which is bolstered by few-shot interaction adjustments. Nonetheless, it is GMAP's capacity for dynamic manipulation trajectory computation that sets it apart, allowing it to nimbly navigate and surmount obstacles at interaction points where static motion primitives falter, consequently boosting the manipulation success rate.

Visualization results. To intuitively demonstrate the performance of our method in part segmentation, joint parameter estimation, and affordance proposals, we randomly selected 4 categories from the test set, each containing 2 instances for display. As shown in Fig. 4, GMAP effectively identifies and segments parts with non-zero articulation states. Although GMAP's segmentation edges may exhibit some jaggedness for parts in a closed state and tightly adjacent to others, this does not hinder the precise estimation of joint direction and position. Moreover, our method can intelligently select the most appropriate "push" or "pull" action based on state requirements, propose interaction positions, and visualize them through the use of heatmaps.

Real-world Experiments

We directly applied the network trained on synthetic data to experiments with real-world data. As shown in the top left corner of Fig. 5, our real-world experimental setup is equipped with a RealSense2 RGB-D camera, which is used to capture depth images of objects. Following this, we performed calibration between the robot coordinate system and the camera coordinate system. The acquired object point cloud is downsampled and normalized for input into the network. Based on Seg-Net's segmentation results, a movable articulated part is selected, and specific manipulation instructions, such as "open 100%", are provided as needed. The top right corner of Fig. 5 shows the visualized results of joint parameter estimation and affordance proposals for three real samples. Despite the presence of noise in real-world data, our method still estimates the position, direction, and current state of joints with relatively high accuracy. By sorting interaction scores, we selected the points with the highest interaction scores for trajectory planning. During the execution phase, we used compliance control techniques to mitigate the impact of estimation errors. The bottom of Fig. 5 shows two key time frames at the start and end of the manipulation, visually demonstrating that our method can adapt to real environments and successfully complete the tasks.

Conclusion and Limitations

We propose GMAP, an innovative framework for robotic manipulation of articulated objects. To achieve general manipulation planning, we meticulously designed a systematic workflow that encompasses key steps including object part segmentation, joint parameter and affordance estimation. Our approach is fortified by point cloud SSL, which significantly reduces the reliance on large-scale labeled data through fine-tuning strategies. This not only enhances the model's ability to generalize from synthetic to real-world data but also streamlines its adaptability to various articulated objects. Experimental outcomes confirm GMAP's exceptional capabilities in perception and interaction, with a smooth transition to real-world scenarios. While our integration of joint parameters into dynamic manipulation planning markedly boosts success rates, further refining the accuracy of interaction affordance proposals is an ongoing challenge.

Acknowledgments

This work was supported by the Guangdong Major Project of Basic and Applied Basic Research (2023B0303000016).

References

Abbatematteo, B.; Tellex, S.; and Konidaris, G. 2019. Learning to generalize kinematic models to novel objects. In *Proceedings of the 3rd Conference on Robot Learning*.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 1877–1901. Curran Associates, Inc.

Chang, A. X.; Funkhouser, T.; Guibas, L.; Hanrahan, P.; Huang, Q.; Li, Z.; Savarese, S.; Savva, M.; Song, S.; Su, H.; et al. 2015. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*.

Chen, G.; Wang, M.; Yang, Y.; Yu, K.; Yuan, L.; and Yue, Y. 2023. PointGPT: Auto-regressively Generative Pre-training from Point Clouds. In *Advances in Neural Information Processing Systems*, volume 36, 29667–29679.

Chu, R.; Liu, Z.; Ye, X.; Tan, X.; Qi, X.; Fu, C.-W.; and Jia, J. 2023. Command-Driven Articulated Object Understanding and Manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8813–8823.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Fan, H.; Su, H.; and Guibas, L. J. 2017. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 605–613.

Geng, H.; Xu, H.; Zhao, C.; Xu, C.; Yi, L.; Huang, S.; and Wang, H. 2023. Gapartnet: Cross-category domain-generalizable object perception and manipulation via generalizable and actionable parts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7081–7091.

Hazra, R.; Dos Martires, P. Z.; and De Raedt, L. 2024. Saycanpay: Heuristic planning with large language models using learnable domain knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 20123–20133.

He, K.; Chen, X.; Xie, S.; Li, Y.; Dollár, P.; and Girshick, R. 2022. Masked Autoencoders Are Scalable Vision Learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 16000–16009.

Jain, A.; Lioutikov, R.; Chuck, C.; and Niekum, S. 2021. Screwnet: Category-independent articulation model estimation from depth images using screw theory. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 13670–13677. IEEE.

Jiang, Z.; Hsu, C.-C.; and Zhu, Y. 2022. Ditto: Building digital twins of articulated objects from interaction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5616–5626.

Joublin, F.; Ceravola, A.; Smirnov, P.; Ocker, F.; Deigmoeller, J.; Belardinelli, A.; Wang, C.; Hasler, S.; Tanneberg, D.; and Gienger, M. 2023. CoPAL: corrective planning of robot actions with large language models. *arXiv preprint arXiv:2310.07263*.

Li, X.; Wang, H.; Yi, L.; Guibas, L. J.; Abbott, A. L.; and Song, S. 2020. Category-level articulated object pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 3706–3715.

Loshchilov, I.; and Hutter, F. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Mandikal, P.; and Grauman, K. 2021. Learning dexterous grasping with object-centric visual affordances. In *2021 IEEE international conference on robotics and automation (ICRA)*, 6169–6176. IEEE.

Mo, K.; Guibas, L. J.; Mukadam, M.; Gupta, A.; and Tulsiani, S. 2021. Where2act: From pixels to actions for articulated 3d objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 6813–6823.

Mo, K.; Zhu, S.; Chang, A. X.; Yi, L.; Tripathi, S.; Guibas, L. J.; and Su, H. 2019. PartNet: A Large-Scale Benchmark for Fine-Grained and Hierarchical Part-Level 3D Object Understanding. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Nagarajan, T.; and Grauman, K. 2020. Learning affordance landscapes for interaction exploration in 3d environments. *Advances in Neural Information Processing Systems*, 33: 2005–2015.

Pang, Y.; Wang, W.; Tay, F. E.; Liu, W.; Tian, Y.; and Yuan, L. 2022. Masked autoencoders for point cloud self-supervised learning. In *European conference on computer vision*, 604–621. Springer.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017a. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017b. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30.

Qin, Y.; Chen, R.; Zhu, H.; Song, M.; Xu, J.; and Su, H. 2020a. S4g: Amodal single-view single-shot se (3) grasp detection in cluttered scenes. In *Conference on robot learning*, 53–65. PMLR.

Qin, Z.; Fang, K.; Zhu, Y.; Fei-Fei, L.; and Savarese, S. 2020b. Keto: Learning keypoint representations for tool manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 7278–7285. IEEE.

Rana, K.; Haviland, J.; Garg, S.; Abou-Chakra, J.; Reid, I.; and Suenderhauf, N. 2023. SayPlan: Grounding Large Language Models using 3D Scene Graphs for Scalable Robot Task Planning. In *7th Annual Conference on Robot Learning*.

Redmon, J.; and Angelova, A. 2015. Real-time grasp detection using convolutional neural networks. In *2015 IEEE international conference on robotics and automation (ICRA)*, 1316–1322. IEEE.

Shi, Y.; Cao, X.; and Zhou, B. 2021. Self-Supervised Learning of Part Mobility from Point Cloud Sequence. In *Computer Graphics Forum*, volume 40, 104–116. Wiley Online Library.

Siarohin, A.; Woodford, O. J.; Ren, J.; Chai, M.; and Tulyakov, S. 2021. Motion representations for articulated animation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 13653–13662.

Szot, A.; Clegg, A.; Undersander, E.; Wijmans, E.; Zhao, Y.; Turner, J.; Maestre, N.; Mukadam, M.; Chaplot, D. S.; Maksymets, O.; et al. 2021. Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in Neural Information Processing Systems*, 34: 251–266.

Todorov, E.; Erez, T.; and Tassa, Y. 2012. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, 5026–5033. IEEE.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Van Den Oord, A.; Vinyals, O.; et al. 2017. Neural discrete representation learning. *Advances in neural information processing systems*, 30.

Wang, S.; Han, M.; Jiao, Z.; Zhang, Z.; Wu, Y. N.; Zhu, S.-C.; and Liu, H. 2024. LLMˆ 3: Large Language Model-based Task and Motion Planning with Motion Failure Reasoning. *arXiv preprint arXiv:2403.11552*.

Wang, X.; Zhou, B.; Shi, Y.; Chen, X.; Zhao, Q.; and Xu, K. 2019. Shape2Motion: Joint Analysis of Motion Parts and Attributes From 3D Shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wang, Y.; Wu, R.; Mo, K.; Ke, J.; Fan, Q.; Guibas, L. J.; and Dong, H. 2022. Adaafford: Learning to adapt manipulation affordance for 3d articulated objects via few-shot interactions. In *European conference on computer vision*, 90–107. Springer.

Wu, R.; Zhao, Y.; Mo, K.; Guo, Z.; Wang, Y.; Wu, T.; Fan, Q.; Chen, X.; Guibas, L.; and Dong, H. 2021. Vat-mart: Learning visual action trajectory proposals for manipulating 3d articulated objects. *arXiv preprint arXiv:2106.14440*.

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.; and Su, H. 2020. SAPIEN: A SimulAted Part-based Interactive ENvironment. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Xie, S.; Gu, J.; Guo, D.; Qi, C. R.; Guibas, L.; and Litany, O. 2020. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *Computer Vision – ECCV 2020,Lecture Notes in Computer Science*, 574–591. Springer.

Xu, Z.; He, Z.; and Song, S. 2022. Universal manipulation policy network for articulated objects. *IEEE robotics and automation letters*, 7(2): 2447–2454.

Yan, Z.; Hu, R.; Yan, X.; Chen, L.; Van Kaick, O.; Zhang, H.; and Huang, H. 2020. RPM-Net: recurrent prediction of motion and parts from point cloud. *arXiv preprint arXiv:2006.14865*.

Yi, L.; Huang, H.; Liu, D.; Kalogerakis, E.; Su, H.; and Guibas, L. 2018. Deep part induction from articulated object pairs. *arXiv preprint arXiv:1809.07417*.

Yu, X.; Tang, L.; Rao, Y.; Huang, T.; Zhou, J.; and Lu, J. 2022. Point-BERT: Pre-Training 3D Point Cloud Transformers With Masked Point Modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 19313–19322.

Zeng, H.; Zhang, P.; Li, F.; Wang, J.; Ye, T.; and Guo, P. 2024a. Masked Generative Extractor for Synergistic Representation and 3D Generation of Point Clouds. *arXiv preprint arXiv:2406.17342*.

Zeng, H.; Zhang, P.; Wu, C.; Wang, J.; Ye, T.; and Li, F. 2024b. MARS: Multimodal Active Robotic Sensing for Articulated Characterization. In Larson, K., ed., *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, 1634–1642.

Zhang, J.; Zhang, J.; Pertsch, K.; Liu, Z.; Ren, X.; Chang, M.; Sun, S.-H.; and Lim, J. J. 2023. Bootstrap Your Own Skills: Learning to Solve New Tasks with Large Language Model Guidance. In *7th Annual Conference on Robot Learning*.

Zhang, R.; Guo, Z.; Gao, P.; Fang, R.; Zhao, B.; Wang, D.; Qiao, Y.; and Li, H. 2022. Point-m2ae: multi-scale masked autoencoders for hierarchical point cloud pre-training. *Advances in neural information processing systems*, 35: 27061–27074.

Zhang, Z.; Girdhar, R.; Joulin, A.; and Misra, I. 2021. Self-Supervised Pretraining of 3D Features on Any Point-Cloud. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 10252–10263.