V-MAO: Generative Modeling for Multi-Arm Manipulation of Articulated Objects – Supplementary Materials

Xingyu Liu
Robotics Institute
Carnegie Mellon University
`xingyul3@cs.cmu.edu`

Kris M. Kitani
Robotics Institute
Carnegie Mellon University
`kkitani@cs.cmu.edu`

A Object-centric Control for Articulated Objects

After a successful grasp, the next goal is to apply torques on robot arms and grippers to move the object part i to the desired pose. In Section 3.3 of the main paper, we mentioned **Object-centric Control for Articuated Objects (OCAO)** formulation. In this section, we provide more technical details.

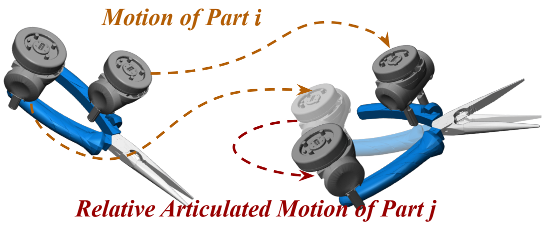


Figure 1: **Motion Decomposition of Object-centric Control.** $\forall i$, the motion of articulated part $i+1$ can be decomposed to rigid motion of articulated part i and relative articulated motion of articulated part j w.r.t. articulated part i. Both components are modeled.

Different from multiple independently moving objects, the articulated object parts are connected with mechanical joints and their motion is under constraints. Our goal is to control the motion of articulated object parts with robot grippers while respecting the mechanical constraints. The core idea is to compute the desired change of the 6D poses of robot grippers given the desired change of the 6D pose of one of the object rigid parts and the changes of all joint angles. As illustrated in Figure 1, to satisfy the mechanical constraints, the rigid motion of one object part has to depend on the rigid motion of another connected object part. We define a dependency graph for the articulated object where each node is an object rigid part and two nodes are connected by an edge if there is a mechanical joint connecting the two object parts. The edge directions in the graph define the dependency of rigid motion decomposition during control. We assume there is no cycle in object joint connections, i.e. the graph is acyclic. Therefore, we can find a dependency in the graph such that each node has exactly one dependency.

Suppose $\mathcal{E} = \{(\mathcal{M}_j, \mathcal{M}_i)\}$ is the set of edges in the graph and for all i and j, revolute joint $l_{i,j}$ connects object parts $\mathcal{M}_j$ and $\mathcal{M}_i$, where $\mathcal{M}_i$ is represented by its 6D pose $[R_i, T_i]$ including rotation component $R_i \in \mathbb{R}^{3\times 3}$ and translation component $T_i \in \mathbb{R}^3$, and $l_{i,j} = (\mathbf{a}_{i,j}, \vec{d}_{i,j}, \theta_{i,j})$ is represented by joint axis anchor $\mathbf{a}_{i,j} \in \mathbb{R}^3$, joint axis direction $\vec{d}_{i,j} \in \mathbb{R}^3$, and joint angle $\theta_{i,j}$.

Suppose $\mathcal{M}_i = [R_i, T_i]$ moves to $\mathcal{M}'_i = [R'_i, T'_i]$ and joint angle $\theta_{i,j}$ changes to $\theta'_{i,j}$ after motion. Suppose the 6D poses of robot gripper i before and after motion are $[R_{g_i}, T_{g_i}]$ and $[R'_{g_i}, T'_{g_i}]$. To ensure that the grasps are always valid for gripper i, there should be no relative motion between robot gripper i and the object part being grasped $\mathcal{M}_i$. Therefore we have

$$\begin{aligned} R'_{g_i} &= R'_i R_i^{-1} \cdot R_{g_i} \\ T'_{g_i} &= T'_i + R'_i R_i^{-1}(T_{g_i} - T_i) \end{aligned} \tag{1}$$

which represents applying the rigid transformation of object part i on gripper i.

5th Conference on Robot Learning (CoRL 2021), London, UK.

Supposed $(\mathcal{M}_i, \mathcal{M}_j) \in \mathcal{E}$, i.e. the motion of $\mathcal{M}_j$ depends on $\mathcal{M}_i$. We decompose the rigid motion of $\mathcal{M}_j$ into two parts: 1) the rigid motion of $\mathcal{M}_i$ and 2) relative articulated motion due to the change of joint angle $\theta_{i,j}$. Suppose the 6D poses of robot gripper j before and after motion are $[R_{g_j}, T_{g_j}]$ and $[R'_{g_j}, T'_{g_j}]$. Similarly, to ensure that the grasps are always valid for gripper j, there should be no relative motion between robot gripper j and the object part being grasped $\mathcal{M}_j$. Therefore we have

$$\begin{aligned}
R'_j &= R'_i R_i^{-1} \cdot R(\vec{d}, \theta'_{i,j} - \theta_{i,j}) \cdot R_j \\
T'_j &= T'_i + R'_i R_i^{-1} \underbrace{[\underbrace{R(\vec{d}, \theta'_{i,j} - \theta_{i,j}) \cdot (T_j - \mathbf{a}_{i,j}) + \mathbf{a}_{i,j}}_{\text{Relative Articulated Motion Due to Change of } \theta_{i,j}} - T_i]}_{\text{Rigid Motion of Object Part } i} \\
R'_{g_j} &= R'_i R_i^{-1} \cdot R(\vec{d}, \theta'_{i,j} - \theta_{i,j}) \cdot R_{g_j} \\
T'_{g_j} &= T'_i + R'_i R_i^{-1} \underbrace{[\underbrace{R(\vec{d}, \theta'_{i,j} - \theta_{i,j}) \cdot (T_{g_j} - \mathbf{a}_{i,j}) + \mathbf{a}_{i,j}}_{\text{Relative Articulated Motion Due to Change of } \theta_{i,j}} - T_i]}_{\text{Rigid Motion of Object Part } i}
\end{aligned} \tag{2}$$

where $R(\vec{u}, \phi) \in \mathbb{R}^{3\times 3}$ is the matrix of rotation of angle ϕ with respect to axis $\vec{u}$.

Note that Equation (2) can be iteratively applied to all object parts based on specified dependencies in the object part graph. After all goal poses $[R'_{g_i}, T'_{g_i}], \forall i$ are computed, we use Operational Space Control (OSC) to move the all grippers i to the goal poses, which will also move the articulated objects to the desired configurations.

Since the dependencies between the object parts are usually not unique, the intermediate trajectory of $[R'_{g_i}, T'_{g_i}]$ and $[R'_{g_j}, T'_{g_j}]$ in Equations (1) and (2) can be different given different object part dependencies. We leave the study of the choice of the object part dependencies and its impact on manipulation as future work.

B Exploration of Contact Point Combination

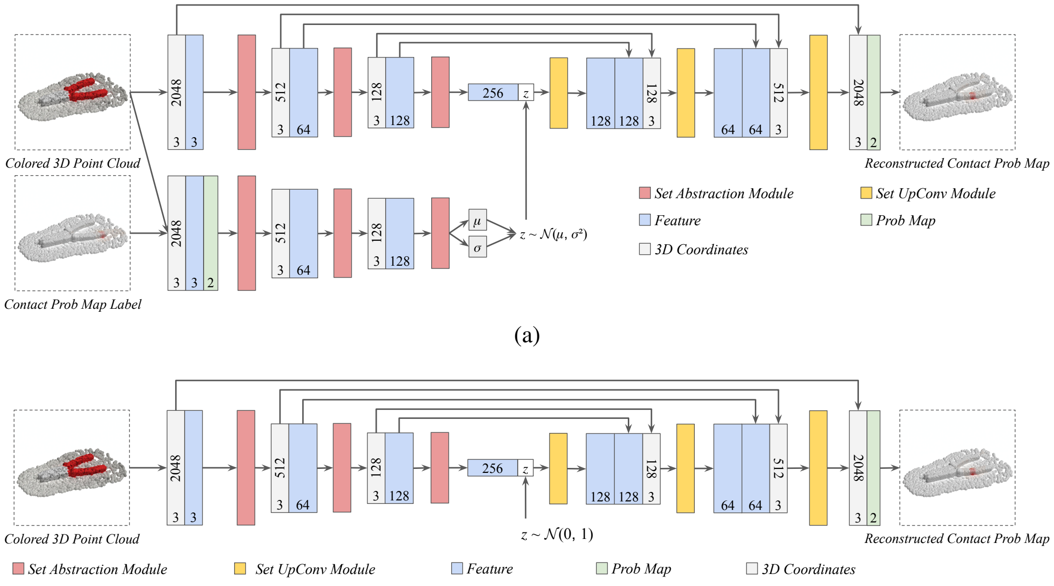
In Section 3.4 of the main paper, we mentioned **automatic contact label generation from exploration**. In this section, we provide more technical details of it. The algorithm of random exploration of contact point combinations is illustrated in Algorithm 1.

To facilitate the collection of successful contact point combinations for grasping, we first a set of human expert demonstrations of feasible contact point combinations $\mathcal{D}$. The human demonstrations are selected intuitively, e.g. two points on the opposite sides of the plier handle. The training label set $\mathcal{T}$ will be initialized with $\mathcal{D}$.

We randomly sample a successful contact point tuple from the training label set. Then from the sampled contact points, we randomly sample points within the neighborhood of radius r of the successful contact points to form a new contact point tuple. Using the new contact point tuple, manipulation actions including inverse kinematics, planning, and Operational Space Control will be executed. If the manipulation is successful, the new contact point tuple will be considered a success and added to the training label set. The above process is repeated until the size of the training label set is large enough.

Algorithm 1: Contact Point Random Exploration

```
Input: Initial demonstration set
       $\mathcal{D} = \{(\mathbf{x}_{g_1}^{(i)}, \mathbf{x}_{g_2}^{(i)}, \ldots, \mathbf{x}_{g_n}^{(i)}) \mid i \in \mathbb{Z}\}$,
       object point cloud $\mathbf{p} = \{\mathbf{x}_i \in \mathbb{R}^3, i \in \mathbb{Z}\}$
Output: Training label set $\mathcal{T}$
$\mathcal{T} \leftarrow \mathcal{D}$ ;
while $|\mathcal{T}| < N$ do
    contact point tuple $\mathbf{s} \sim \mathcal{T}$ ;
     ▷ randomly sample from $\mathcal{T}$
    new contact point tuple $\mathbf{s}' = ()$ ;
    for $i \leftarrow 1$ to $n$ do
        $\mathbf{x}_{g_i} \leftarrow \mathbf{s}[i]$ ;
        $\mathbf{n} \leftarrow \{\mathbf{x} \in \mathbf{p} \mid ||\mathbf{x} - \mathbf{x}_{g_i}||_2 < r\}$
            ▷ neighborhood points
        $\mathbf{x} \sim \mathbf{n}$  ▷ randomly sample
            from neighborhood
        $\mathbf{s}'[i] \leftarrow \mathbf{x}$ ;
    end
    $r \leftarrow$ Execute contact point tuple $\mathbf{s}'$ ;
    if $r =$ Success then
        $\mathcal{T} = \mathcal{T} \cup \{\mathbf{s}'\}$
    end
end
```



(a)

(b)

Figure 2: **Architecture of V-MAO Neural Network** during (a) training and (b) inference. Note that the probability map has two channels because the robot gripper we use in the experiment has two fingers. The initial three-channel feature is the RGB values.

C Neural Network Architecture

We illustrate the details of the CVAE architecture used in V-MAO in Figure 2. The neural network architecture is instantiated by set abstraction layers from [1] and set upconv layers from [2]. During training, we send both the 3D point cloud appended with RGB values and the same 3D point cloud appended with probability map into the neural network while during inference, only the 3D point cloud appended with RGB values are fed into the network.

During decoding, each layer of set upconv [2] iteratively up-sample the point cloud to recover the resolution before down-sampling and eventually recover the original full 3D point cloud. The cross entropy loss $\mathcal{L}_{\text{recon}} = H(\mathbf{p}_c, \hat{\mathbf{p}}_c)$ between the two point clouds $\mathbf{p}_c$ and $\hat{\mathbf{p}}_c$ is the average/sum of the cross entropy between the contact probability values of each corresponding point pair from the two point clouds.

The architecture of the deterministic baseline "Top-1" is the same as the Figure 2(b), i.e. the architecture of V-MAO used during inference, except for there is no latent code z included. This ensures V-MAO is fairly compared against the deterministic baseline.

D Experiments on Objects with Three Rigid Parts

In the main paper, we conduct experiments on two robots and articulated objects with two rigid parts. In this section, we conduct an additional experiment on three robots and an object with three parts. The object consists of three rectangular sticks in color of red, white and blue respectively that can rotate with respect to the same axis. The articulated object is placed on the table surface with random initial positions and joint angles. We use three Sawyer robots to manipulate the object. Two of the three robots face the table in parallel and the other robot faces the table in opposite direction. Similar to the experiment in the main paper, the goal of the manipulation is to grasp the object parts and move them to reach the desired positions and joint angles within a certain error threshold.

We explore three types of grasping strategy for the three-rigid-part object: sequential grasp, sequential-parallel grasp, and parallel grasp. In **sequential grasp**, the object parts are grasped and moved in a sequential fashion. In the i-th step, robot gripper i grasps the i-th object part while leaving parts $i+1$ through N uncontrolled. We iterate the above step until all parts are grasped

robot	method	grasp order	success rate
Sawyer	Top-1 Point	parallel	80.0
		sequential-parallel	**86.0**
		sequential	90.0
	V-MAO (Ours)	parallel	**86.0**
		sequential-parallel	**86.0**
		sequential	**92.0**

Table 1: **Success rate of Manipulation.** The baseline we compare V-MAO with is selecting the point with the maximum contact probability from deterministic prediction. We report the results averaged from 50 runs.

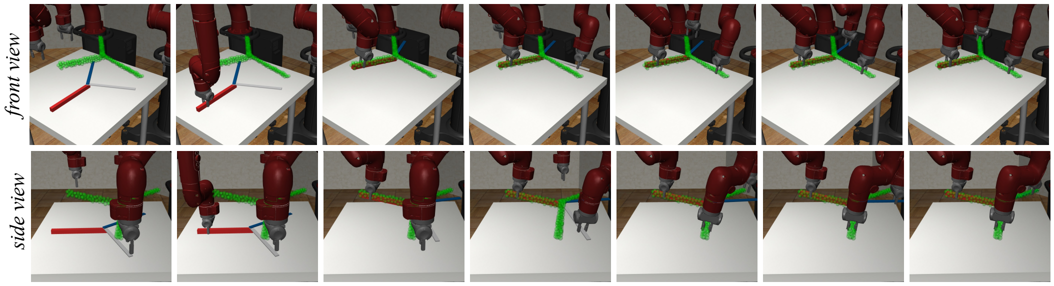


Figure 3: **Visualization of sequential grasp and manipulation using three robot arms.** Rows 1 and 2 illustrate the front and side views of the manipulation of the three-rigid-part object using three Sawyer robots. The object configuration goal of manipulation is marked by the green semi-transparent point cloud.

and controlled. In **sequential-parallel grasp**, the object parts 1 through i are grasped and moved by robots 1 through i respectively in a sequential fashion. Then the rest of the robots $i+1$ through N simultaneously grasp and move the rest of the object parts to the goal locations together. In **parallel grasp**, all object parts are grasped simultaneously. Then the robots move the object parts to the goal locations together.

We report the success rate of manipulation in Table 1. Our V-MAO can achieve more than 80% success rate with Sawyer robot. The performance gap between the V-MAO and the deterministic prediction baseline is small. A possible explanation is that the object parts are long sticks, so it is unlikely for the robots to collide with each other during manipulation. Manipulation fails when some goal locations are not possible for a gripper to reach, in which case both deterministic "Top-1 Point" baseline and our method will fail to reach the goal.

In Figure 3, we visualize the executed sequential manipulation using the contact points sampled from the generative model. The sampled contact points combinations for all robots can avoid collision between robots when all three grippers are grasping the object. After grasping, all three robots can reliably control the object to reach the goal configurations marked by green point cloud.

E More Visualization of Contact Probability maps

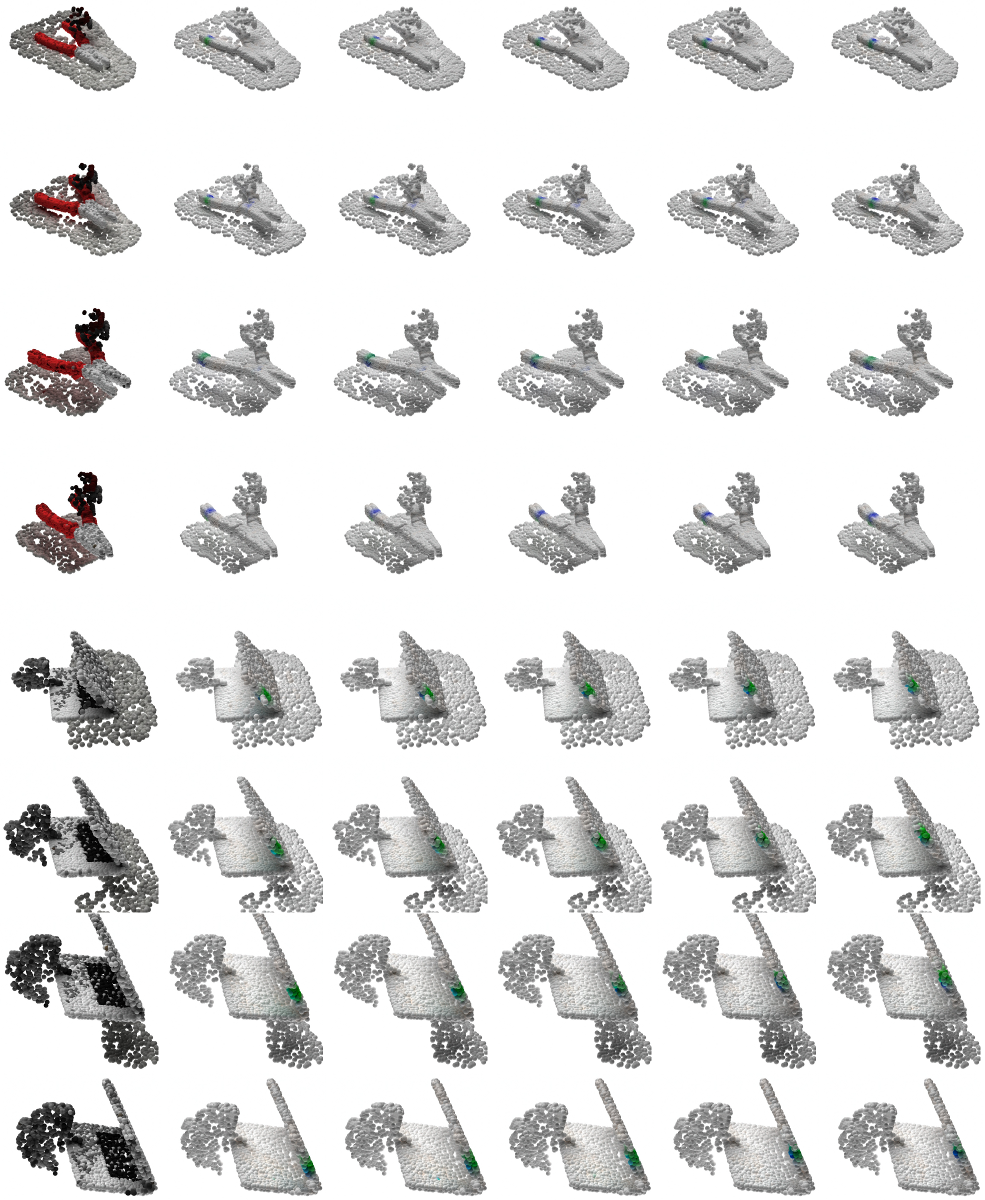
In Section 4.3 of the main paper, we provide visualizations of sampled contact probability map from our generative model. In this section, we provide more visualizations of sampled contact probability map and shows how the geometric features affect the generated maps.

As illustrated in Figure 4, the articulated objects are grasped and lifted by the first gripper. The grasping from the second gripper has not started yet. The sampled contact probability maps of the second gripper not only depend on latent code z, but also depend on the geometry of the objects and the scene. For example, after the pliers are lifted from the table, the sampled contact probability map shows that grasping from both horizontal and vertical directions is feasible, while only grasping from the vertical direction is allowed before the pliers are lifted. It further shows that our generative model can effectively incorporate geometric features to learn the contact point distribution on articulated objects.

References

[1] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NIPS*, 2017.

[2] X. Liu, C. R. Qi, and L. J. Guibas. Flownet3d: Learning scene flow in 3d point clouds. In *CVPR*, 2019.



(a) (b) $z = -1.0$ (c) $z = -0.5$ (d) $z = 0.0$ (e) $z = 0.5$ (f) $z = 1.0$

Figure 4: **Visualization of sampled and generated 3D probability maps for the second grasp after the first grasp.** From left to right in each row: (a) the colored 3D point cloud at the current manipulation state; (b)-(f) generated probability map of the next grasp given specific latent code z, where green and blue denote the finger 1 and 2 of the second gripper in the second grasping step. Zoom in for better a view of the figures.