FlowBotHD: History-Aware Diffuser Handling Ambiguities in Articulated Objects Manipulation

Yishu Li*, Wen Hui Leng*, Yiming Fang*, Ben Eisner, David Held
Robotics Institute, School of Computer Science
Carnegie Mellon University, United States
{yishul, wleng, yimingf, baeisner, dheld}@andrew.cmu.edu

Abstract: We introduce a novel approach to manipulate articulated objects with ambiguities, such as opening a door, in which multi-modality and occlusions create ambiguities about the opening side and direction. Multi-modality occurs when the method to open a fully closed door (push, pull, slide) is uncertain, or the side from which it should be opened is uncertain. Occlusions further obscure the door's shape from certain angles, creating further ambiguities during the occlusion. To tackle these challenges, we propose a history-aware diffusion network that models the multi-modal distribution of the articulated object and uses history to disambiguate actions and make stable predictions under occlusions. Experiments and analysis demonstrate the state-of-art performance of our method and specifically improvements in ambiguity-caused failure modes. Our anonymized website is available at https://flowbothd.github.io/.

Keywords: Ambiguity, Articulated Objects, Diffusion

1 Introduction

Ambiguities are common in manipulation tasks. Consider opening a door as an example: when the door is fully closed, how it opens is ambiguous. Without further information, even experienced human beings can't always tell by sight whether it opens by push, pull or even slide. Another type of ambiguity is caused by occlusions. When the door opens to certain angles, it might look like a line instead of a plane to a single view observer (see Figure 2). Occlusions cause loss of information, introducing ambiguities in the understanding of the object shape.

Faced with multi-modal ambiguities, a human might interact with the object and observe the outcome; for example, a person might push the door and see if it opens, and if not, they can adjust their strategy. Furthermore, momentary occlusions are not typically a problem for humans that can remember their past observations to supplement their current perception. Building on these insights, we expect an intelligent robot to have the ability to learn from multi-modal training data, predict multi-modal solutions for ambiguous observations, and with the help of historical observations, be able to make stable predictions even during occlusions.

To be generalizable and robust across different articulated object kinematics and geometries, we build on prior work that models articulated object manipulation with 3D articulation flow (3DAF) vectors as proposed in Eisner et al. [1]. This approach predicts a set of 3D vectors for each point on the articulated object that indicates its opening direction. Results from this approach have shown impressive abilities to generalize to novel objects in the real world, after training only in simulation. However, this approach fails when facing the ambiguous settings described above.

Therefore, in this paper, we propose a history-aware diffusion method to model multi-modality and handle occlusions. We use diffusion to model the multi-modal distribution and predict multiple possible object opening directions, and we use historical observations to adjust or disambiguate the distribution according to past trials. Our experiments and analysis demonstrate the benefits of our method in such ambiguous situations.

8th Conference on Robot Learning (CoRL 2024), Munich, Germany.

The contributions of this paper include:

(1) A novel approach to handling both multi-modality ambiguity and occlusion ambiguity for articulated object manipulation.
(2) A novel structure for incorporating object history.
(3) Simulated experiments across a wide variety of PartNet-Mobility objects.
(4) Real-world experiments that demonstrate the feasibility of our method in real-world.

2 Related Work

Articulated Object Manipulation: Manipulating articulated objects is an important and challenging task. One line of work investigated building manipulation systems with analytical methods [2, 3, 4, 5, 6]. With the help of large-scale datasets [7, 8], recent lines of work turn to learning-based methods. Several works model manipulation choices with explicit articulation parameters and poses [9, 10, 11, 12]. To improve the model's generalization ability to different kinematics, several recent works learn to predict geometry-aware kinematics-free representations such as visual affordances [7, 13, 14] and articulation flows [1, 15] which achieve significant improvements in object generalization. Different from previous learning-based works which tend to use regression models for prediction, we propose to use diffusion models for better multi-modality handling.

Uncertainty Handling for Robotics: Modeling uncertainty is a well-explored topic in various robotics fields [16, 17]. Previous works have included uncertainty in tasks including visual perception and tracking [18, 19, 20, 21], motion planning [22, 23, 24, 25], safety control [26] and also articulation parameter estimation [27] to build robust models. With similar motivation, our work focuses on building a model robust to visual uncertainty caused by multi-modality and occlusions.

Diffusion Models in Robotics: Diffusion models, a class of generative models, learns to denoise an input to approach a given distribution. Following its impressive success in image generation, diffusion has also shown great potential in robotics tasks. Recent works have applied diffusion models to various tasks of robotics, including reinforcement learning [28, 29], imitation learning [30, 31, 32, 33, 34] and motion planning [35, 36, 37]. Similar to Ze et al. [34], the input of our diffusion model is a compact 3D point cloud which requires the model to have 3D awareness. Different from prior methods, we show how 3D diffusion models can be used for articulated object manipulation.

3 Background

3D Articulation Flow (3DAF): Suppose that an articulated object is described by a point cloud $P = \{p_0, \ldots, p_{N-1}\}$ for each 3D point $p_i \in \mathbb{R}^3$ on the object visible surface. We assume that each point p_i belongs to some part on the object, and that the parts are connected in a kinematic tree. We further assume that there are no closed-loop kinematic chains on the object that we are manipulating and also that each joint has only one degree of freedom. Following Eisner et al. [1], for each point p_i, we define the 3D Articulation Flow (3DAF) $f_i \in \mathbb{R}^3$ as the motion that point would undergo if there were an infinitesimal positive displacement $\delta\theta$ of the joint between that part and its parent. Eisner et al. [1] showed that, given a suction gripper, we can most efficiently open the articulated object using the predicted 3DAF by attaching the gripper to the point with the largest predicted flow norm $\arg\max_i \|f_i\|$ and then moving the gripper along its predicted flow direction f_i. The set of 3D Articulation Flows $F = \{f_0, \ldots, f_{N-1}\}$ for all points in the point cloud is hierarchy-free (unlike a kinematic tree), and invariant under object translation and scaling, which allows for effective learning over a large set of objects of different categories and effective transfer from simulation to the real world.

Denoising Diffusion Probabilistic Models (DDPMs): DDPMs [38] are a class of generative models that learn to progressively denoise towards the target data distribution starting from pure Gaussian noise. During the training process, for each ground-truth example, we iteratively add noise over multiple noising steps; we can compute the noised input in closed form as $x_t \sim$

$\mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I})$ where $\bar{\alpha}_t$ is a noise parameter. The task objective is to train a noise predictor $\epsilon_\theta(x_t, t)$ that inverts the added noise. We minimize the learning objective formulated as: $\mathbb{E}[||\epsilon_t - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t, t)||_2{}^2]$. With its iterative noise addition and denoising process, diffusion has demonstrated the ability to model complicated and multi-modal distributions.

4 Problem Statement and Assumptions

The task we propose to tackle is opening fully-closed objects from a single-view depth-sensor. The inputs to our model are a history of 3D point cloud observations of the articulated object being manipulated. As mentioned above, we assume that the articulated objects do not consist of a closed-loop kinematic chain and that each joint has only a single degree of freedom.

5 Method

As stated above, our goal is to open fully-closed articulated objects. To achieve this, we aim to predict how each articulated object might open, and we will use this prediction to determine how to manipulate the object. In making this prediction, we consider two types of ambiguity: multi-modality in how the object might open, and occlusion ambiguity, in which there is ambiguity from certain viewpoints because of object self-occlusions. The ambiguities will cause typical failure modes in existing methods[1]: the multi-modal ambiguity causes deterministic models to fail to open an object from a fully closed state; the occlusion ambiguity obscures models from making reasonable predictions from certain viewpoints, causing the robot to get stuck when the object passes through those viewpoints (see Figure 2). We aim to propose a robust model able to overcome the two failure modes and open articulated objects regardless of these ambiguities. To achieve this, we propose a history-aware diffusion model that handles ambiguity caused by both multi-modality and occlusion, together with a system for manipulating articulated objects based on our model's predictions.

5.1 Diffusion for Multi-Modality

Due to the natural ambiguity of closed objects, there might be instances of objects with similar appearance but entirely different opening directions, such as a closed door that might open via push or pull. These examples cause multi-modality in both training datasets and in evaluation settings. To resolve these ambiguities, we take advantage of the ability of diffusion models to capture multi-modal distributions.

To condition the diffusion on a 3D point cloud input $\mathbf{P}_{obs}$, we propose a novel 3D-aware diffusion structure. Our 3D-aware denoiser consists of a PointNet++ [39] encoder $\mathbf{E}_{cur}$ that encodes the current inputs and a DiffusionTransformer (DiT) [40] denoiser $\mathbf{D}$. During training, we first add noise to the ground truth flow F^* (see Section 3) to obtain a noisy flow F_ϵ, following the DDPM [38] diffusion formulation described in Section 3. Then we pair each point and its corresponding noisy flow as the current state $\mathbf{S}_{cur} = \{\mathbf{P}_{obs,i}, F_{\epsilon,i}\}, i \in [0, \mathbf{M})$, where $\mathbf{M}$ is the total point count. We input the current state $\mathbf{S}_{cur}$ through the point cloud encoder to obtain 3D geometry-aware point-wise features: $\mathbf{E}_{cur}(\mathbf{S}_{cur})$. Following the standard diffusion training method [38], we train the denoiser $\mathbf{D}$ conditioned on the current state encoding with the following learning objective:

$$\mathbb{E}[||\epsilon_t - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t, t, \mathbf{E}_{cur}(\mathbf{S}_{cur}))||_2{}^2].$$

5.2 Aggregating History for Disambiguation and Occlusion Handling

While the diffusion training method alone is capable of modeling multi-modal distributions, it lacks the ability to disambiguate based on past interactions. For example, during occlusions, if the robot succeeded in the past timesteps to partially open the door with the pull action, it should output a distribution that puts more probability on continuing to pull the door open in the future.

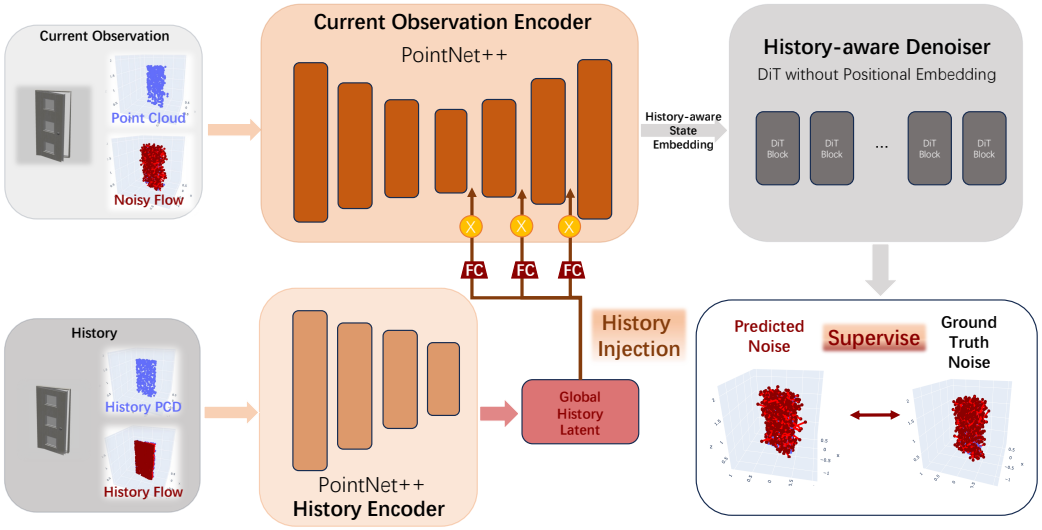


Figure 1: **FlowBotHD** structure: the history observation and the history flow are input into a history encoder to obtain a global history latent. This history latent is injected into the current observation encoder through fully-connected projection layers and element-wise multiplication. The current observation encoder outputs a history-aware point-wise embedding. The history-aware embedding is then input to a DiT-based denoiser to predict the flow [1], i.e. the predicted future motion of the object if it were to be opened a small amount.

Therefore, we propose to integrate the past history with a history encoder. We define a compact history state representation as the previous time step's point cloud observation $\mathbf{P}_{his}$, along with the previous step's flow F_{his}. During training, we generate a synthetic history by sampling a random small angle to perturb the object to generate the "history" point cloud observation, and we obtain the corresponding ground truth articulation flow as the history flow. At inference time, we use the model's previous point cloud observations and its previous flow predictions as history.

To aggregate history into the 3D-aware diffusion pipeline, we propose to fuse an embedding of the history state into the current state encoding process. Specifically, we first pair the history point cloud $\mathbf{P}_{his}$ and the history flow F_{his} into the history state $\mathbf{S}_{his} = \{\mathbf{P}_{his,i}, f_{his,i}\}, i \in [0, \mathbf{M})$, where $\mathbf{M}$ is the total point count of the history step. Then we input $\mathbf{S}_{his}$ to a PointNet++ [39] based history state encoder $\mathbf{E}_{his}$ to obtain a global history latent embedding for the history point cloud: $\mathbf{L}_{his} = \mathbf{E}_{his}(\mathbf{S}_{his}) \in \mathbb{R}^{d_h}$, where d_h is the dimension of the history latent.

To inject the history information into the current state encoder, we use a similar two-stream architecture to that of Shridhar et al. [41]; see Figure 1. In their case, the two streams are based on "what" and "where" whereas our two streams encode the current observation and the history. Consider a subset of layers in the encoder $\mathbf{E}_{cur}$ from layer k_1 to layer k_2 as: $\{e_{cur,k_1}, \ldots, e_{cur,k_2}\}$, each of dimensionality d_k. We wish to element-wise multiply each of these layers by the history encoding $\mathbf{L}_{his}$. In order to make the dimensionality match up, we define a series of linear projection layers $\mathbf{W}_k \in \mathbb{R}^{d_k \cdot d_h}$. We then element-wise multiply (Hadmard product) each of the encoder layers with the projected history encoding: $e'_{cur,k} = e_{cur,k} \odot (\mathbf{W}_k \cdot \mathbf{L}_{his})$.

To be compatible with situations at inference time where we don't always have history available (such as the first timestep), we also include samples without providing history during training. To handle the absence of history, we use a learnable feature token $\mathbf{L}_0$ as the history latent embedding for history injection. In summary, the history-aware encoder module $\mathbf{E}_{ha}(\mathbf{S}_{cur}, \mathbf{L}_{his})$ produces a history-aware encoding combining the current state observation and the history latent embedding:

$$\mathbf{F}_{ha} = \begin{cases} \mathbf{E}_{ha}(\mathbf{S}_{cur}, \mathbf{L}_{his} = \mathbf{E}_{his}(\mathbf{S}_{his})) & \text{if there is history,} \\ \mathbf{E}_{ha}(\mathbf{S}_{cur}, \mathbf{L}_{his} = \mathbf{L}_0) & \text{otherwise.} \end{cases}$$

Then we input the history-aware feature into the DiT-based denoiser $\mathbf{D}$ to output the predicted noise with learning objective: $\mathbb{E}[||\epsilon_t - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t, t, \mathbf{F}_{ha})||_2{}^2]$. From the predicted noise, we are able to estimate the predicted flow $\hat{f}$ following the equations in Section 3.

5.3 Articulation Manipulation Policy

Along with the history-aware diffusion model, we propose a novel articulation manipulation policy that applies the flow predicting model to real manipulation tasks taking into consideration the multi-modality of diffusion and the incorporation of object history.

Switch Grasp Point (SG): Following Eisner et al. [1], to achieve the best manipulation efficiency with a suction gripper, we want to grasp at the point with the largest leverage; thus the optimal action given the predicted flow is to grasp the point that has the maximum flow norm [1]. Due to the multi-modality of diffusion, the prediction at different steps will vary; each prediction might imply a different point that the robot should optimally grasp. In order to correct for past potential grasping mistakes as well as to maintain a reasonable re-grasping cost, we propose to switch the grasp point whenever the leverage - defined as the current largest flow norm - of the newly predicted grasp point is larger than that of the current grasp point by a threshold: $||\hat{F}||_{max} - ||\hat{F}||_g > \delta l$, where $\hat{F}$ is the current step's flow prediction for all points in the point cloud, g is the index of the point that is closest to the current grasp point, and δl is the threshold which we set to 0.2 in practice.

Consistency Check (CC): Being able to model multi-modality, some samples from the diffusion model will likely be correct, whereas other samples will be incorrect. With the expectation that diffusion will produce the correct prediction given enough samples, we design a heuristic consistency check to filter the actions that are not consistent with the previously executed motion. Specifically, we define the consistency direction g_{his} as the gripper movement of the previous execution (we only keep the previous execution if its motion was greater than 0). To perform a consistent following step, the predicted action's direction shouldn't deviate too much from that of the previous step; thus we set an angle threshold $\delta\theta_{cc}$ for the consistency check: $\arccos \frac{\hat{f} \cdot g_{his}}{||\hat{f}|| \cdot ||g_{his}||} < \delta\theta_{cc}$, where $\hat{f}$ is the current sampled flow direction. We will sample actions from the diffusion model until we find one that passes the consistency check, before executing the sampled action.

History Filter (HF): Since we have trained our model to use the history flow prediction to guide the current step's predictions, we don't want previous mistakes to mislead the model. Therefore, we include a history filter that only updates the history point cloud $\mathbf{P}_{his}$ and history flow F_{his} if the previous action achieves a good movement in execution. Specifically, after an execution, we only update the history if two criteria are satisfied:

1. the delta gripper motion is consistent with past executions: $\arccos \frac{\delta g_{cur} \cdot g_{his}}{||\delta g_{cur}|| \cdot ||g_{his}||} < 90°$ where g_{cur} is the current gripper movement and g_{his} is the gripper movement from the previous execution.
2. the gripper motion is larger than a threshold $||\delta g_{cur}|| > \delta m$ where g_{cur} is the current gripper movement and δm is the threshold

The pseudocode for our method, FlowBotHD (History and Diffusion) can be found in Algorithm 1.

6 Experiments and Analysis

We use the PartNet-Mobility dataset [7] to train and evaluate the models. We split a subset of this dataset containing 21 categories of objects (the same categories as in prior work [13, 1]) into a train set and test set in which the test set consists of unseen samples of the training categories (707 training objects and 166 testing objects). To enable the models to open objects from a fully closed state, we create a mixed dataset which contains both randomly opened samples (as in prior work [1]) and training samples that are fully closed. The fully closed examples are the most ambiguous, since it might be unclear which direction these objects open.

Algorithm 1: The FlowBotHD articulation manipulation policy

Require: $\theta \leftarrow$ parameters of a trained FlowBotHD network D
  while `NotFullyOpened()` **do**
    $\mathbf{P}_{obs} \leftarrow$ Observation
    $\hat{F} \sim D_\theta(\mathbf{P}_{obs}, \mathbf{S}_{his})$ Sample the current flow direction from the diffusion model
    $\hat{f} \leftarrow$ `GetMaxFlow`$(\hat{F})$
    if `Inconsistent`$(\hat{f}, g_{his})$ and not `ExceedMaxTrialPerStep()` **then**
      continue ; `// If the sampled action fails the Consistency Check`
    end if; `// (Sec. 5.3) then resample`
    $p_{grasp} \leftarrow$ `SelectGraspPoint`$(\hat{F})$
    if `NeedSwitchGraspPoint`$(p_{grasp,his}, p_{grasp})$ **then**
      $p_{grasp,his} = p_{grasp}$, ; `// If the Switch Grasp Point check is passed`
      `Grasp`(P_{grasp}) ; `// (Sec. 5.3) then change the grasp point`
    end if
    Apply a force to the gripper in the direction of $\hat{f}$ for small duration δt.
    $g_{cur} \leftarrow$ `GetDeltaGripper()`
    if `Consistent`(g_{cur}, g_{his}) **then**
      $g_{his} \leftarrow g_{cur}$
      if $||g_{cur}|| > \delta m$ **then**
        $\mathbf{S}_{his} \leftarrow (\mathbf{P}_{obs}, \hat{F})$; `// Apply the History Filter (Sec. 5.3)`
      end if
    end if
  end while

6.1 Simulation results

To evaluate each model's ability to open objects, we build a simulation environment with PyBullet [42] that simulates operating a suction gripper to operate objects modeled in PartNet-Mobility dataset. We initialize every object as fully closed and filtered out instances that don't open from the fully closed state even with the ground truth action sequence. Following prior work [1], we compute 2 evaluation metrics: **Success Rate**: the percent of objects the model succeeds to open $> 90\%$ of the maximum opening amount, and **Normalized Distance** [13]: the mean distance between the end state of this trial and the fully opened state, normalized with its range of motion $d_{goal} = \frac{||j_{end} - j_{closed}||}{||j_{opened} - j_{closed}||}$ where j_{end} is the joint angle at the end of the policy rollout, j_{closed} and j_{opened} are the fully closed angle and the fully opened angle respectively. We evaluate each test object for 5 times.

Baselines: To compare our model with the original FlowBot3D [1] setting, we trained FlowBot3D on a dataset with only randomly opened ("RO") samples which aligns with the original training strategy of Eisner et al. [1]. We also retrain Flowbot on the mixed dataset ("MD") that we use to train our model in which we also include examples in which the object is fully closed. Further, for a fair comparison to our method, we implement a baseline in which we allow FlowBot to switch its grasp point ("SG") using the same criteria as for our method.

Comparing the $Baselines$ models with $Ours$ in Tables 1 and 2 (see Appendix for normalized distance metrics), we can see that our full method FlowBotHD with the complete set of policy improvements (switch grasp point + consistency check + history filter) achieves the best overall performance. Specifically, compared to the original FlowBot3D [1] model, it increases the success rate averaged across all samples from 87% to 95%, or a 61% reduction in error rate (from 13% to 5%). We see an especially large increase in success rate in the door category (from 23% to 77%), which is the most ambiguous category when fully closed. These quantitative results demonstrate our model's ability to overall improve the performance and more importantly, better handle multi-modal cases.

	SG	CC	HF	AVG_c	AVG_s																				
Baselines																									
FlowBot (RO)	×	×	×	0.800	0.865	**0.85**	0.50	0.75	0.71	**1.00**	0.30	0.91	0.80	0.75	0.69	0.89	0.90	**0.89**	**0.96**	**1.00**	**1.00**	0.23	**1.00**	0.86	**1.00**
FlowBot (RO)	✓	×	×	0.793	0.903	**0.85**	0.50	0.75	0.82	**1.00**	0.60	0.87	0.80	0.58	0.88	0.96	0.94	0.82	**0.96**	0.73	0.60	0.53	**1.00**	0.86	0.80
FlowBot (MD)	×	×	×	0.800	0.875	0.60	0.80	**1.00**	0.82	**1.00**	**1.00**	0.89	0.76	0.33	0.72	0.91	0.91	0.78	0.92	0.80	0.40	0.37	**1.00**	**1.00**	**1.00**
FlowBot (MD)	✓	×	×	0.842	0.926	0.75	**1.00**	**1.00**	0.96	**1.00**	**1.00**	0.95	0.80	0.33	**0.99**	0.95	0.95	**0.89**	**0.96**	0.73	0.40	0.40	**1.00**	0.89	0.90
Ablations																									
Ours- No History	×	×	×	0.692	0.806	0.70	0.00	0.75	0.84	0.84	0.60	0.82	0.72	0.44	0.59	0.80	0.86	0.76	0.72	0.93	0.20	0.47	**1.00**	0.86	0.95
Ours- No History	✓	×	×	0.740	0.864	0.70	0.40	0.85	0.93	**1.00**	1.00	0.76	0.80	0.33	0.69	0.91	0.90	**0.89**	0.80	0.73	0.00	0.53	**1.00**	0.86	0.70
Ours- No History	✓	✓	×	0.743	0.876	0.70	0.30	0.75	0.95	0.92	1.00	0.73	0.80	0.11	0.84	0.94	0.91	0.84	0.92	0.87	0.20	0.53	**1.00**	0.86	0.70
Ours- No Diffusion	✓	×	×	0.673	0.720	0.55	0.30	0.45	0.89	**1.00**	0.40	0.85	0.80	0.28	0.56	0.82	0.70	0.58	0.16	0.87	**1.00**	0.40	**1.00**	0.86	**1.00**
Ours- No Diffusion	✓	×	✓	0.683	0.735	0.50	0.30	0.55	0.89	**1.00**	0.10	0.82	0.80	**0.97**	0.56	0.86	0.72	0.58	0.08	0.60	**1.00**	0.53	**1.00**	0.80	**1.00**
Ours																									
FlowBotHD	✓	×	×	0.735	0.865	0.35	0.10	0.90	**1.00**	0.96	0.50	0.89	0.80	0.36	**0.99**	0.85	0.92	0.49	0.92	0.47	**1.00**	0.43	**1.00**	0.83	0.95
FlowBotHD	✓	✓	×	**0.884**	0.937	0.65	0.70	0.95	**1.00**	**1.00**	**1.00**	**0.96**	**0.96**	0.82	0.97	0.94	0.96	0.64	**0.96**	0.73	0.80	0.70	**1.00**	0.97	0.95
FlowBotHD	✓	×	✓	0.758	0.904	0.30	0.00	0.85	**1.00**	0.96	0.90	**0.96**	0.80	0.45	**0.99**	0.95	0.95	0.64	0.92	0.53	0.80	0.43	**1.00**	0.86	0.85
FlowBotHD	✓	✓	✓	0.878	**0.952**	0.75	0.70	0.95	0.98	**1.00**	0.70	**0.96**	0.84	0.58	**0.99**	**0.98**	**0.98**	0.73	**0.96**	0.87	**1.00**	**0.77**	**1.00**	0.83	**1.00**

Table 1: Success Rate Metric Results (↑): Amount of success trials (normalized distance 0.1) of different objects' categories after a full rollout across different methods. The higher the better. $\textbf{AVG}_c$ and $\textbf{AVG}_s$ refers to metric averaged over categories or samples. For the policy column, SG refers to switch grasp point, CC refers to consistency check and HF refers to history filter. We only add history filter in history-aware methods, and add consistency check with diffusion-based methods because inconsistent predictions in deterministic methods can't be refined even with re-predictions.

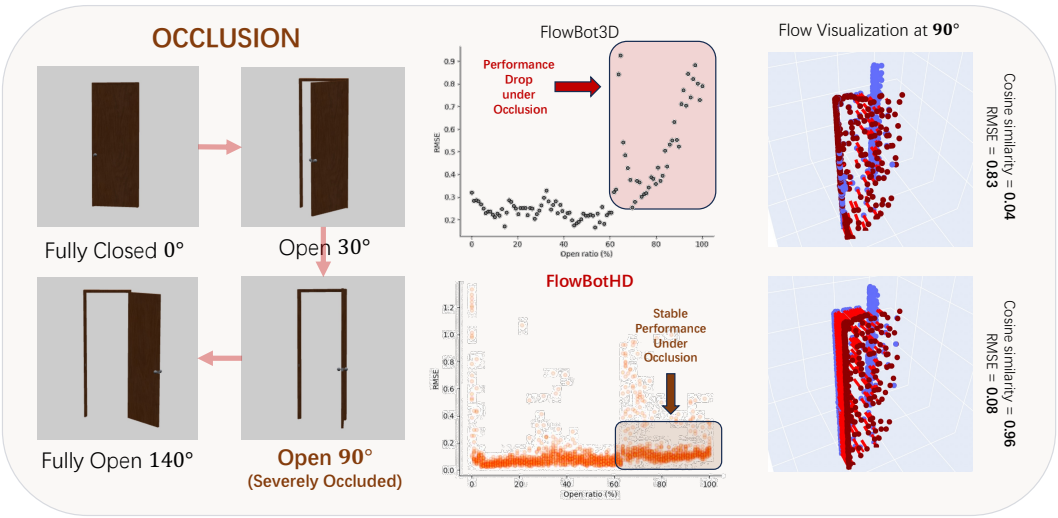


Figure 2: Occlusion analysis: We quantitatively demonstrate our model's performance improvements over severely occluded case. We open the object to different angles, make predictions and plot the RMSE metric (↓) against the open ratio. We also include flow visualizations to intuitively show the quality comparison of the predictions.

6.2 Ablations

We carried out ablations in which we remove various components of the model: No history, no diffusion, and varying whether we use switch grasping ("SG" column), consistency check ("CC" column), and history filtering ("HF" column). We observe large performance improvements with adding history awareness to the diffusion model (comparing Ours to "Ours- No History"). This improvement demonstrates that the extra guidance of history is beneficial for diffusion to reduce the multi-modality in cases of past success in which the opening direction is now disambiguated.

These tables show that our policy improvements of switch grasp point, consistency check, and history filter all contribute to the model performance. The importance of the consistency check potentially means that one challenge for using the current diffusion method is its randomness and, as a

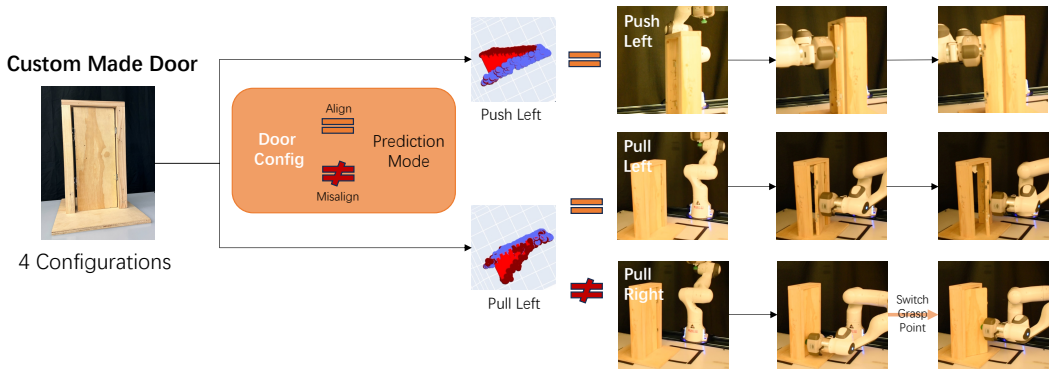


Figure 3: Demonstration of FlowBotHD opening custom made door in different configurations with or without the need to switch grasp point. The first two rows show examples of our model's successful trials with the first prediction aligned with the current door configuration. The last row shows switch grasp point when the first prediction falls into misaligned modes.

result, inconsistency. By heuristically filtering out inconsistent predictions, our model can produce an improved quality of predictions.

6.3 Qualitative Analysis

To understand the ability of our model to handle occlusions which can not be intuitively shown in the quantitative results, in Figure 2, we show the model predictions for different opening angles. At certain angles, the observed point cloud is severely occluded to become almost a line. We can see from the RMSE plot that the FlowBot3D prediction suddenly fails at a certain angles for which there are severe occlusions. The FlowBotHD distribution looks much better where it still has high probability of predicting the correct direction even at the point of occlusion. The flow visualization shows a clear improvement between our model and the baseline for this ambiguous case.

6.4 Real-World Experiments

We performed real-world experiments on general articulated objects and specifically our custom made multi-modal door to demonstrate FlowBotHD's ability to function in real-world scenarios. For the multi-modal door, we demonstrate our model's ability to open the door in all 4 possible modes, and also the ability to adjust to better grasp points with the switch grasp point policy. Static visualizations of the real-world roll-outs on custom made door and general articulated objects are presented in Figure 3 and Appendix Figure 14, and full videos can be found on our project website. Details of the environmental set-ups including the design of custom made door can be found in Appendix F.

7 Conclusion

In this work, we propose a history-aware diffusion model **FlowBotHD** to solve ambiguities in articulated object manipulation, along with a carefully designed heuristic manipulation policy. Both quantitative experiments and qualitative analysis have shown the overall effectiveness of our model and specific improvements over multi-modal or severely occluded cases.

Limitations: While our model greatly improves the ability of the policy to handle ambiguities, we have observed trade-offs in accuracy and consistency when adopting a diffusion pipeline. Our diffusion model still sometimes preserve multi-modality even when the object is opened enough to indicate a deterministic opening direction. While we currently solve this issue by adding a consistency check in the policy, an ideal model should learn to disambiguate by itself based on history trials. Therefore, we hope to further explore how to exert better control over the diffusion's randomness for better consistency and accuracy.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No.IIS-2046491 and Toyota Research Institute.

References

[1] B. Eisner, H. Zhang, and D. Held. Flowbot3d: Learning 3d articulation flow to manipulate articulated objects. *arXiv preprint arXiv:2205.04382*, 2022.

[2] R. Martín-Martín, S. Höfer, and O. 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.

[3] K. Desingh, S. Lu, A. Opipari, and O. C. Jenkins. Factored pose estimation of articulated objects using efficient nonparametric belief propagation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7221–7227. IEEE, 2019.

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

[5] R. Staszak, M. Molska, K. Młodzikowski, J. Ataman, and D. Belter. Kinematic structures estimation on the rgb-d images. In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, volume 1, pages 675–681. IEEE, 2020.

[6] A. Jain, R. Lioutikov, C. Chuck, and S. 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.

[7] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. 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.

[8] K. Mo, S. Zhu, A. X. Chang, L. Yi, S. Tripathi, L. J. Guibas, and H. 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.

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

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

[11] X. Wang, B. Zhou, Y. Shi, X. Chen, Q. Zhao, and K. 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.

[12] R. Hu, W. Li, O. Van Kaick, A. Shamir, H. Zhang, and H. Huang. Learning to predict part mobility from a single static snapshot. *ACM Transactions on Graphics (TOG)*, 36(6):1–13, 2017.

[13] Z. Xu, H. Zhanpeng, and S. Song. Umpnet: Universal manipulation policy network for articulated objects. *IEEE Robotics and Automation Letters*, 2022.

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

[15] H. Zhang, B. Eisner, and D. Held. Flowbot++: Learning generalized articulated objects manipulation via articulation projection. *arXiv preprint arXiv:2306.12893*, 2023.

[16] H. F. Durrant-Whyte. Uncertain geometry in robotics. *IEEE Journal on Robotics and Automation*, 4(1):23–31, 1988.

[17] S. Thrun. Probabilistic algorithms in robotics. *Ai Magazine*, 21(4):93–93, 2000.

[18] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In *Machine intelligence and pattern recognition*, volume 5, pages 435–461. Elsevier, 1988.

[19] M. Mohammadzadeh Babr, M. Faghihabdolahi, D. Ristić-Durrant, and K. Michels. Deep learning-based occlusion handling of overlapped plants for robotic grasping. *Applied Sciences*, 12(7):3655, 2022.

[20] H. Chandel and S. Vatta. Occlusion detection and handling: a review. *International Journal of Computer Applications*, 120(10), 2015.

[21] P. Wunsch and G. Hirzinger. Real-time visual tracking of 3d objects with dynamic handling of occlusion. In *Proceedings of International Conference on Robotics and Automation*, volume 4, pages 2868–2873. IEEE, 1997.

[22] A. Lazanas and J.-C. Latombe. Motion planning with uncertainty: a landmark approach. *Artificial intelligence*, 76(1-2):287–317, 1995.

[23] G. Kahn, P. Sujan, S. Patil, S. Bopardikar, J. Ryde, K. Goldberg, and P. Abbeel. Active exploration using trajectory optimization for robotic grasping in the presence of occlusions. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4783–4790. IEEE, 2015.

[24] L. Zhang and J. C. Trinkle. The application of particle filtering to grasping acquisition with visual occlusion and tactile sensing. In *2012 IEEE International Conference on Robotics and Automation*, pages 3805–3812. IEEE, 2012.

[25] S. Dasgupta, A. Gupta, S. Tuli, and R. Paul. Uncertainty-aware active learning of nerf-based object models for robot manipulators using visual and re-orientation actions. *arXiv preprint arXiv:2404.01812*, 2024.

[26] J. Choi, F. Castaneda, C. J. Tomlin, and K. Sreenath. Reinforcement learning for safety-critical control under model uncertainty, using control lyapunov functions and control barrier functions. *arXiv preprint arXiv:2004.07584*, 2020.

[27] A. Jain, S. Giguere, R. Lioutikov, and S. Niekum. Distributional depth-based estimation of object articulation models. In *Conference on Robot Learning*, pages 1611–1621. PMLR, 2022.

[28] Z. Wang, J. J. Hunt, and M. Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.

[29] A. Ajay, Y. Du, A. Gupta, J. Tenenbaum, T. Jaakkola, and P. Agrawal. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022.

[30] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

[31] T. Pearce, T. Rashid, A. Kanervisto, D. Bignell, M. Sun, R. Georgescu, S. V. Macua, S. Z. Tan, I. Momennejad, K. Hofmann, et al. Imitating human behaviour with diffusion models. *arXiv preprint arXiv:2301.10677*, 2023.

[32] M. Reuss, M. Li, X. Jia, and R. Lioutikov. Goal-conditioned imitation learning using score-based diffusion policies. *arXiv preprint arXiv:2304.02532*, 2023.

[33] Z. Xian, N. Gkanatsios, T. Gervet, T.-W. Ke, and K. Fragkiadaki. Chaineddiffuser: Unifying trajectory diffusion and keypose prediction for robotic manipulation. In *7th Annual Conference on Robot Learning*, 2023.

[34] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. In *ICRA 2024 Workshop on 3D Visual Representations for Robot Manipulation*, 2024.

[35] K. Saha, V. Mandadi, J. Reddy, A. Srikanth, A. Agarwal, B. Sen, A. Singh, and M. Krishna. Edmp: Ensemble-of-costs-guided diffusion for motion planning. *arXiv preprint arXiv:2309.11414*, 2023.

[36] M. Janner, Y. Du, J. B. Tenenbaum, and S. Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

[37] A. Simeonov, A. Goyal, L. Manuelli, L. Yen-Chen, A. Sarmiento, A. Rodriguez, P. Agrawal, and D. Fox. Shelving, stacking, hanging: Relational pose diffusion for multi-modal rearrangement. *arXiv preprint arXiv:2307.04751*, 2023.

[38] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

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

[40] W. Peebles and S. Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.

[41] M. Shridhar, L. Manuelli, and D. Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on robot learning*, pages 894–906. PMLR, 2022.

[42] E. Coumans and Y. Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. 2016.

Appendix

Table of Contents

A Training Details **12**

B Simulation Details **13**

C Multi-modality Analysis **14**
- C.1 Multi-modality of Dataset 14
- C.2 Model Performance on Multi-modal Cases 14

D Occlusion Analysis **17**

E Policy Analysis **17**
- E.1 History Filter Analysis 17
- E.2 Switch Grasp Point Analysis 19
- E.3 Step Number Analysis 20

F Real-world Equipment **21**
- F.1 Hardware 21
- F.2 Workspace 21
- F.3 Foreground Segmentation 21
- F.4 Robot Control Paradigm 22
- F.5 Custom Multi-modal Door 22

A Training Details

Model Structure: FlowBotHD consists of a current state encoder, a history state encoder and a denoiser. Both current and history state encoder are based on PointNet++ [39], where the history encoder only includes the PointNet++ encoding module (i.e. no decoder) that outputs a global latent of 128 dimensions. We kept the same model architecture and hyperparameter set as the original PointNet++ paper except for using a regression head for the history encoder output instead of the original segmentation head. We inject the history latent into every layer of the PointNet++ framework in the decoder (see Figure 1). This is performed through a hadamard product of the current point cloud encoding with the latent encodings at every decoding layer to produce the final output of the model. We base the denoiser on a DiT with 5 layers, 4 heads, and a hidden size of 128.

Dataset Details: Similar to the original FlowBot3D paper [1], we create a randomly opened dataset (RO) in which we randomly sample the view perspective, the target joint, and the open ratio. To open objects from the fully closed state, we create another mixed dataset (MD) in which we half of the objects are fully closed and the other half randomly opened. To enable history-aware training, we generate a dataset with 1/3 of the objects fully closed, 1/3 randomly opened but without history and the final third randomly opened and with history at random interval.

Training Process: We train the model for 450 epochs using AdamW optimizer with a weight decay of 1e-5. We use a learning rate of 1e-4, and a batch size of 128. We evaluate the model every 20 epochs, and save the checkpoint with the best winner-take-all RMSE metric. The Winner-take-all metric means that we repeatedly make predictions for each sample 20 times and record the best RMSE.

B Simulation Details

We implement a simulation environment with PyBullet that simulates operating a suction gripper to operate PartNet-Mobility objects. We first roll out the simulation with ground truth flow and filter out samples that can't open with the ground truth motion, due to errors in the simulator. We repeat each sample for 5 times during evaluation. The final test object categories we evaluated on after filtering and repeating are listed in Table 2.

Icon	Name	Count	Icon	Name	Count	Icon	Name	Count	Icon	Name	Count
	Oven	20		Microwave	10		Table	245		Phone	5
	Bucket	10		Window	55		Furniture	765		Door	30
	Box	20		Kettle	25		Dishwasher	45		Laptop	40
	Toilet	55		Safe	12		WashMachine	25		TrashCan	35
	KitchenPot	25		Refrigerator	75		FoldingChair	15		Stapler	20

Table 2: Simulation Objects and Sample Counts

To manipulate the objects, we first attach the suction gripper to the object. This is implemented by first teleporting the gripper to a position close to the target grasp point and then gradually moving towards the target grasp point until contact is detected. After contact, we apply a physical force constraint between the gripper and the object. The move action is then implemented as applying a certain velocity along the predicted direction.

Hyperparameter in Policy: There are 3 hyperparameters to set in Algorithm 1: the threshold for switch grasp point δl, the threshold for consisteny check $\delta\theta_{cc}$ and the threshold for good movement δm. We set $\delta l = 0.2$, $\delta\theta_{cc} = 30°$ and δm =1e-2 in simulation. Table 3 demonstrates the quantitative simulation results for the normalized distance metric, computed in the same way as in Eisner et al. [1].

	SG	CC	HF	$\underline{\text{AVG}_c}$	$\underline{\text{AVG}_s}$																				
Baselines																									
FlowBot (RO)	×	×	×	0.250	0.166	0.23	0.54	0.29	0.29	**0.00**	0.90	0.10	0.23	0.35	0.31	0.10	0.15	0.18	0.10	0.10	0.48	0.39	**0.07**	0.11	0.08
FlowBot (RO)	✓	×	×	0.220	0.117	**0.16**	0.53	0.29	0.22	**0.00**	0.43	0.10	0.15	0.45	0.14	0.05	0.09	0.18	0.09	0.27	0.31	0.46	**0.07**	0.15	0.26
FlowBot (MD)	×	×	×	0.183	0.143	0.31	**0.19**	0.06	0.22	**0.00**	**0.06**	0.14	0.21	0.47	0.30	0.09	0.12	0.23	0.12	0.09	0.24	0.63	**0.07**	**0.05**	0.08
FlowBot (MD)	✓	×	×	0.159	0.098	0.26	0.08	0.07	0.09	**0.00**	**0.06**	0.08	0.14	0.56	**0.07**	0.05	0.09	0.16	0.09	**0.07**	0.35	0.60	**0.07**	0.13	0.16
Ablations																									
Ours- No History	×	×	×	0.255	0.176	0.29	0.57	0.18	0.19	0.16	0.36	0.14	0.23	0.46	0.28	0.18	0.14	0.20	0.31	**0.07**	0.53	0.54	**0.07**	0.09	0.10
Ours- No History	✓	×	×	0.178	0.113	0.26	0.36	0.09	0.09	**0.00**	**0.06**	0.13	0.10	0.38	0.17	0.08	0.09	0.12	0.21	0.31	0.29	0.45	**0.07**	0.10	0.20
Ours- No History	✓	✓	×	0.176	0.110	0.28	0.28	0.19	0.10	0.08	**0.06**	0.10	0.12	0.75	0.11	0.07	0.10	**0.10**	0.08	0.10	0.11	0.45	**0.07**	0.11	0.27
Ours- No Diffusion	✓	×	×	0.265	0.213	0.35	0.67	0.55	0.15	**0.00**	0.51	0.14	0.19	0.73	0.31	0.15	0.20	0.31	0.37	0.11	**0.08**	0.55	**0.07**	0.12	0.08
Ours- No Diffusion	✓	×	✓	0.194	0.183	0.39	0.48	0.40	0.11	**0.00**	0.45	0.18	0.17	**0.04**	0.30	0.11	0.18	0.28	0.35	0.24	**0.08**	0.48	**0.07**	0.15	**0.07**
Ours																									
FlowBotHD	✓	×	×	0.181	0.139	0.55	**0.50**	0.13	0.05	0.04	0.39	0.12	0.15	0.33	0.08	0.15	0.10	0.40	0.11	0.23	0.09	0.52	**0.07**	0.10	0.09
FlowBotHD	✓	✓	×	0.103	0.086	0.33	0.35	0.11	**0.05**	**0.00**	**0.06**	**0.06**	**0.02**	0.15	0.08	0.07	0.08	0.25	0.09	0.08	0.10	0.27	**0.07**	0.07	0.08
FlowBotHD	✓	×	✓	0.110	0.096	0.53	0.52	0.13	**0.05**	0.04	0.10	**0.06**	0.15	0.24	**0.07**	0.05	0.07	0.21	0.13	0.23	0.11	0.49	**0.07**	0.12	0.10
FlowBotHD	✓	✓	✓	**0.096**	**0.072**	0.25	0.26	**0.05**	**0.05**	**0.00**	0.20	**0.06**	0.11	0.38	**0.07**	**0.04**	**0.06**	0.18	**0.07**	**0.07**	0.09	**0.23**	**0.07**	0.09	**0.07**

Table 3: Normalized Distance Metric Results (↓): Normalized distances to the goal articulation joint angle after a full rollout of the methods. The lower the better.

We demonstrate the simulation process in Fig 4. We visualize the flow predictions (the first row), the simulation process (the last row, x-axis is the step number, the y-axis is the open ratio), and the trajectory plot along with policy signals (the middle plot). In these visuals, we can see our model and policy's pattern of opening an object: make trials at the beginning, once the door is opened a bit, make consistent predictions based on past history and consistency check.

The first example is quite smooth, our model succeeds to move the door at the first step. Then it continues to make consistent and history-aware predictions until the door is fully opened. Consis-

tency check didn't have to filter out much predictions due to the good prediction quality, and history is always updated at each step, meaning that we're always using the previous step's history for the current step's prediction. In the second example, our model makes several trials at the fully closed state. History is not updated during these steps and remains none. Once the door is opened, the history information and the consistency check enable the model to execute consistent actions. We can see from the background bar plot that for step 11, the consistency check filters out 12 inconsistent predictions, demonstrating the effectiveness of applying consistency check. Also, we can see that at step 6, the door is opened a bit, but the history is not updated due to the small movement (indicating the prediction is not good enough) which demonstrates how the history filter works.

C Multi-modality Analysis

C.1 Multi-modality of Dataset

Analyzing how many of the objects in the dataset are multi-modal or typical multi-modal paradigms provides valuable insights of the dataset. Take door category as an example, doors without handle typically have 4 possible modes when fully-closed (push left, push right, pull left and pull right); handles can indeed disambiguate which side of the door opens but still leaves ambiguity on whether the door opens by push or pull. Among the 28 training door joints and 7 test door joints, most of them (20 out 28 for training, 5 out of 7 for test) are with handles (2 possible modes - push and pull), and the other samples are without handles (4 possible modes - push left, pull left, push right and pull right).

For other object categories, there is no obvious automatic way to determine whether an object is ambiguous, it is hard to manually count ambiguous samples for all the categories (and subjective), therefore we analyze conceptually where ambiguities typically happen instead. There are three typical multi-modal objects (See Figure 5) : 1) Joints without handles, 2) Joints with small handles that might be missed when downsampling, and 3) Joints with handles that are still possible to open in different ways: for example a dishwasher/oven with a handle on the top of its front surface, although the handle eliminates many modes, it is unclear whether it opens by pulling out like a drawer, or pulling from the top to the bottom like a lid or door.

Similar to counting modes of general articulated objects, there is no precise way to count multi-modality in predictions without including manual judging and counting. To approximately have an idea about the multi-modal predictions of the model, we make 50 predictions for each object in the test set and calculate their RMSE with respect to the ground truth flow. We record how many of the objects include both RMSE < 0.2 (presumably a correct prediction) and RMSE $>$ 0.6 (presumably a different mode than the ground truth mode). We can see from the bar plot in Figure 6 that the model predicts multi-modality mostly in conceptually multi-modal categories like Door, WashingMachine, and Refrigerators. However it also makes multi-modal predictions with conceptually inambiguous cases, which aligns with limitations of this work we addressed in 7 about the difficult balance between precision and multi-modality.

C.2 Model Performance on Multi-modal Cases

To specifically analyze our model's improvements on handling multi-modality, we evaluated baseline FlowBot3D trained on randomly opened dataset (RO) / mixed dataset (MD) and our model FlowBotHD on doors. We open each test door from 0% to 100%. We average the metrics within 10% open to compute the performance for closed doors, and we average the metrics above 10% open to compute the performance for open doors. As we can see from Table 4, our model outperforms the baselines by a large margin across all metrics. The observation that FlowBot3D trained on randomly opened dataset outperforms the one trained on the mixed dataset shows that multi-modal training data can confuse the regression models' training process.

Fig 7 visualizes the denoising process that produces different modes based on the same fully closed door, demonstrating our model's ability of preserving multi-modality for ambiguous examples.

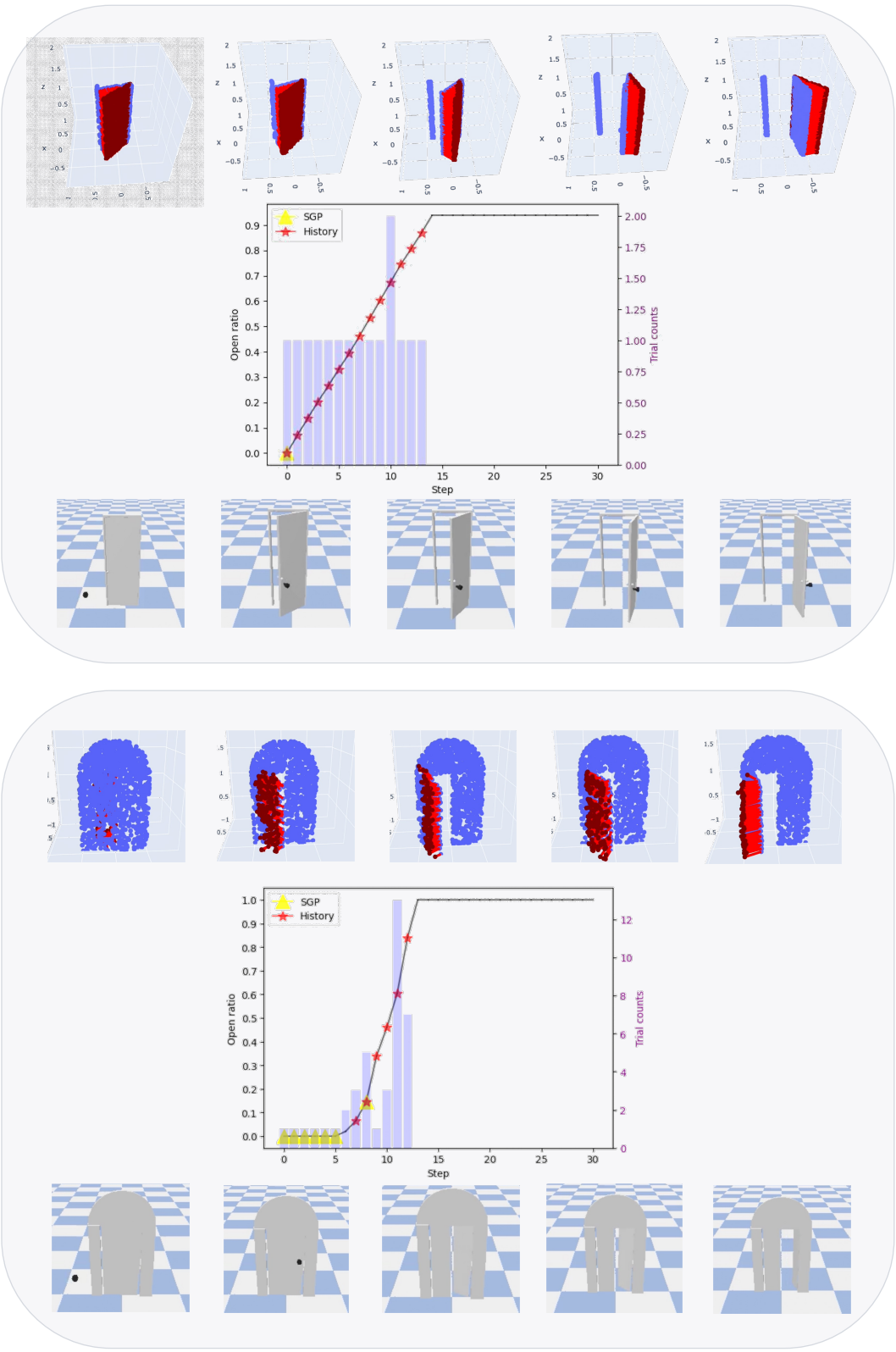


Figure 4: Simulation Visualizations: The plot in the middle is a simulation trajectory plot with x-axis as step number, and left y axis as the open ratio. We visualize the history update signal with red polygons, step with red polygon means updating this step's prediction as the latest history. Yellow triangle represents the switch grasp point (SGP) signal, meaning that this step requires a new grasp point. The bar plot on the background corresponds to number of trials we take to generate a prediction that satisfies the consistency check trial. The axis for the bar plot is the right y-axis.

Figure 5: Ambiguous Showcases: Conceptually, ambiguities typically happen with 1) Joint without handles, 2) Joints with small handles that might be missed when downsampling, and 3) Joints with handles which are still possible to open in different ways.

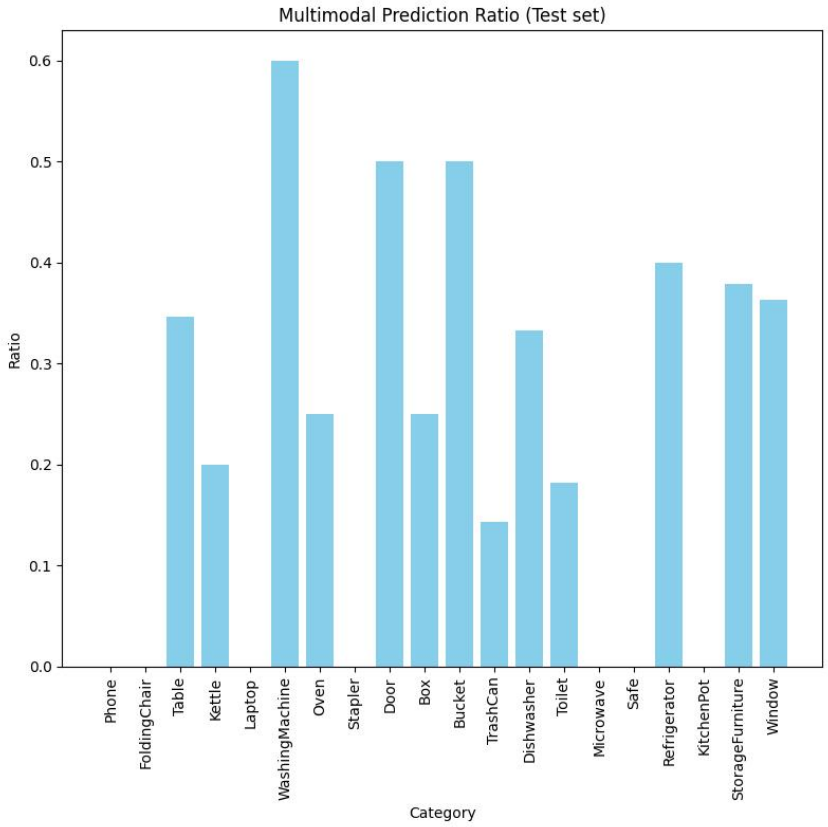


Figure 6: Multi-modal Prediction Ratio for Different Categories: We make 50 predictions for each object in the test set, calculate their RMSE with the ground truth flow, and record how many of the objects include both correct prediction and different modes.

	Cosine (↑)			RMSE(↓)			MAG(↓)		
Model	FlowBot3D (RO)	FlowBot3D (MD)	**FlowBotHD**	FlowBot3D (RO)	FlowBot3D (MD)	**FlowBotHD**	FlowBot3D (RO)	FlowBot3D (MD)	**FlowBotHD**
Closed (<10% open)	0.2033	0.3129	**0.8046**	0.4426	0.4069	**0.1833**	0.2468	0.2150	**0.1047**
Randomly Open (>10% open)	0.7351	0.2720	**0.9089**	0.2218	0.4617	**0.1246**	0.1782	0.2215	**0.0919**

Table 4: Multi-Modality Analysis: We compare our model with baselines on doors. The cosine metric means the cosine similarity between the predicted flow and the ground truth flow, the higher the better. RMSE metric means the root mean square error between prediction and ground truth, the lower the better. MAG metric means the flow magnitude error, the lower the better.

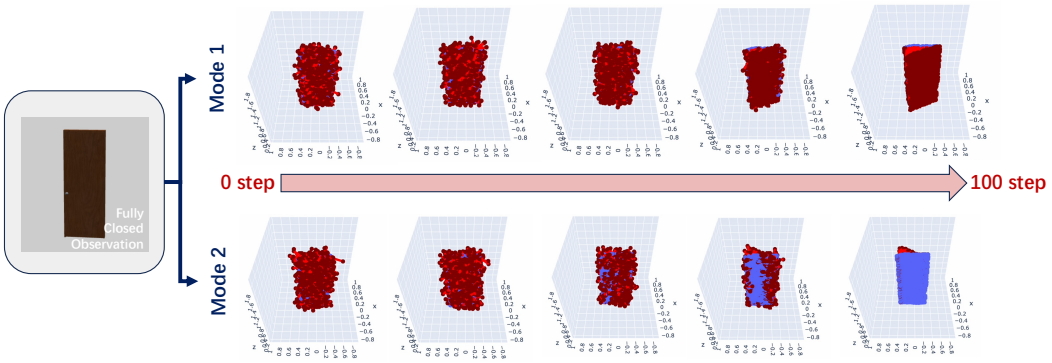


Figure 7: Multi-Model Diffusion Process Visualization: We visualize the denoising process (100 steps). We can see that the flow starts from pure random noises and gradually converges to different modes: Pull and Push.

D Occlusion Analysis

We analyze our model's performance under occlusions on different object categories. A door example is included in the Fig 2 of the main paper, and we include occluded fridge and furniture examples in Fig 8. We can see from the visualizations that with history, FlowBotHD is able to produce stable and robust predictions regardless of the occlusions while FlowBot3D makes less consistent predictions when severely occluded.

E Policy Analysis

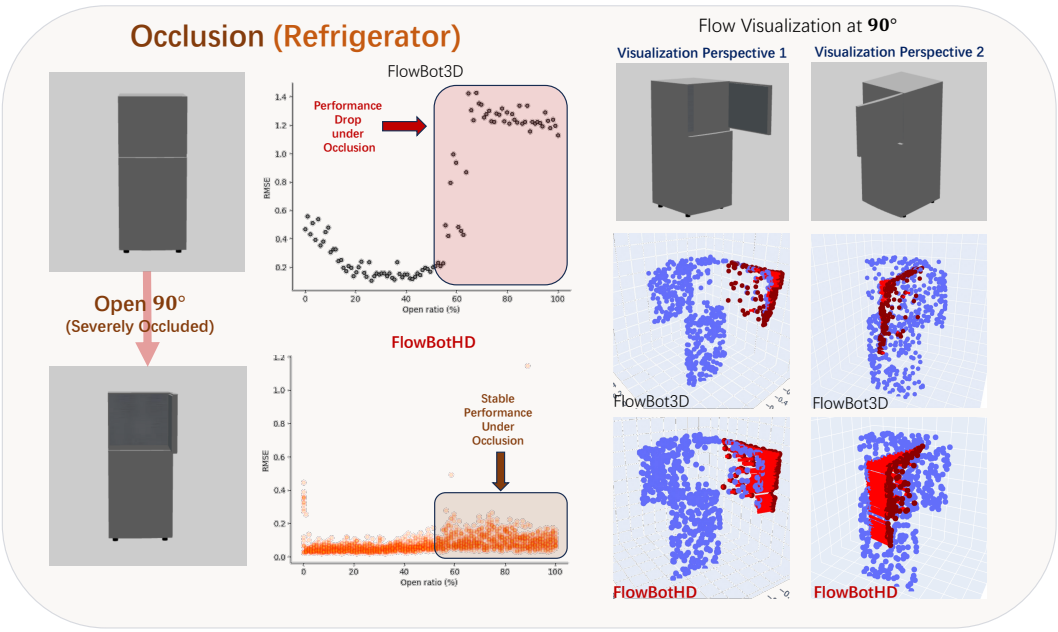
We conduct further analysis on components of our roll-out policies, including the occasional negative effect of history filter, the efficiency of switch grasp point policy and the effect of step number limitations.

E.1 History Filter Analysis

Since we train the model with only ground truth data, we propose to use a history filter to prevent incorrect predictions in the history from disturbing the history distribution; our history filter removes previous predictions that fail to further open the object. However, occasionally we find that including incorrect predictions in the history can lead to better performance; below we analyze when this might occur.

Consider the below dishwasher (which opens by pulling the upper side) as an example, we experimented with 4 types of history input: No history (which can occur either at the beginning or due to the history filter removing incorrect predictions), history from ground-truth predictions, history from predictions with cosine similarity to the ground-truth > 90 degrees, history from predictions with cosine similarity to the ground-truth < 30 degrees..

We can see from the box plot of RMSE distribution (the lower, the better) of 50 samples in Figure 13, that the history from the ground-truth predictions leads to the best performance on average, and the history from predictions with cosine similarity to the ground-truth > 90 degrees leads to worse



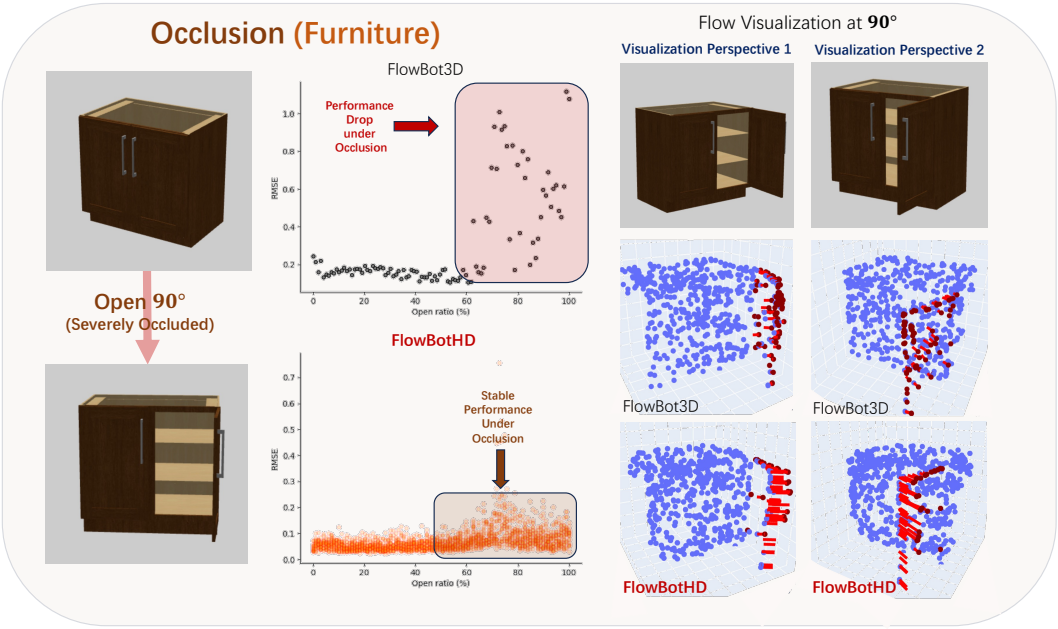


Figure 8: Occlusion analysis: We open the object to different angles, make predictions and plot the Cosine Similarity metric (↑) and RMSE metric (↓) against the open ratio. We also include flow visualizations from different viewing perspective to intuitively show the quality comparison of the predictions. We can see from the visualization that FlowBotHD can make good and consistent predictions even under severe occlusions.

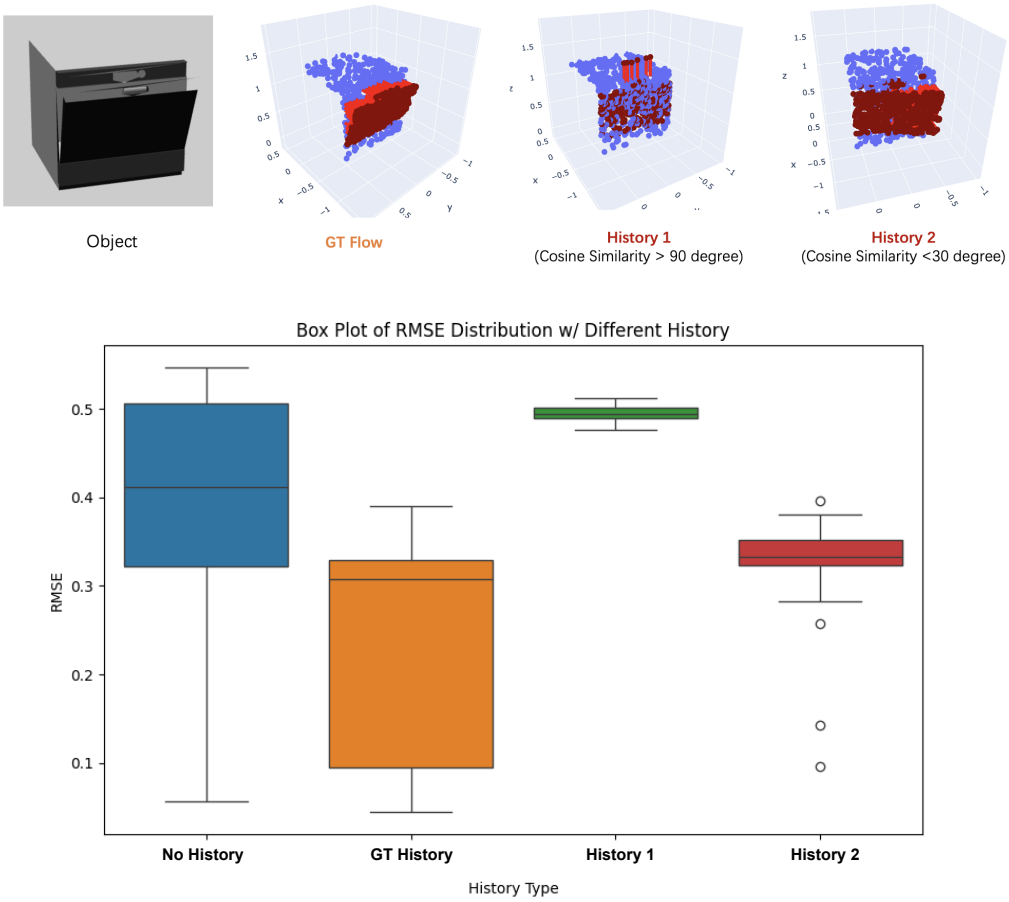


Figure 9: History Filter Analysis: Above shows the visualization of three types of history, the box plot below visualizes the effect of different histories on the next step's prediction quality.

performance than no history. However, interestingly, we also see that including history from predictions with cosine similarity to the ground truth < 90 degrees sometimes leads to better performance than including no history at all.

This experiment shows that including history from incorrect predictions can sometimes lead to better performance and sometimes worse depending on the accuracy of the previous predictions in the history. This implies that the history filter can sometimes inadvertently filter out helpful predictions in the history that might be close to the ground truth. Nonetheless, our results show that using the history filter on average leads to better performance than not using it.

E.2 Switch Grasp Point Analysis

Switching grasp point costs extra time, therefore we should only allow necessary switches to happen. In our method, the stablility of grasp selection is achieved via the leverage change threshold δl as proposed in Section 5.3. By adjusting this threshold, we can control how frequently the policy switches the grasp point.

We performed an additional analysis to calculate the number of grasp actions each object needs. By average, our method needs 1.76 times grasp actions including the initial grasp, so on average each object requires 0.76 switch-grasps. The results show that our method does not require a lot of switching and still leads to improved performance over the baseline.

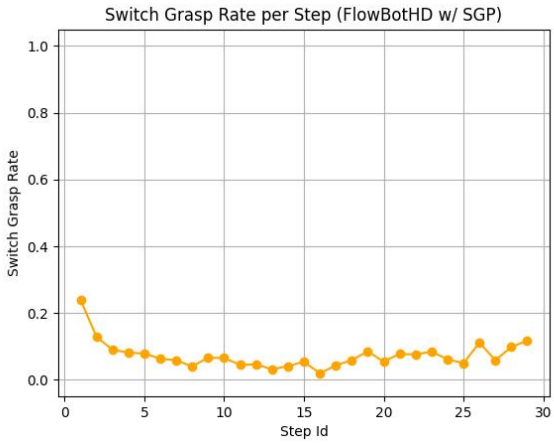


Figure 10: Switch Grasp Rate per Step Plot: As seen in the plot, the switch grasp point rate gets lower among the first few steps, and increases slightly at the later stages.

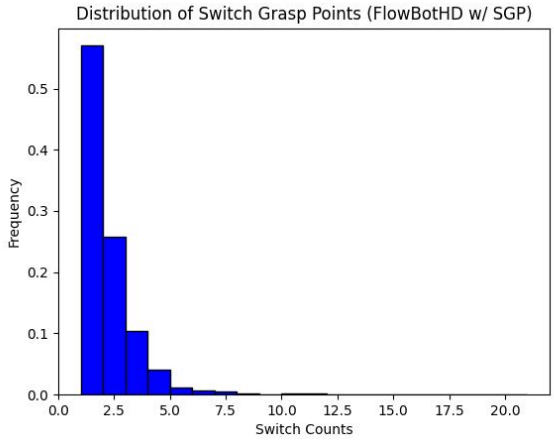


Figure 11: Switch Grasp Count Histogram: we show the distribution of the number of switch grasps across the dataset. About 60% of objects only require one grasp action (the first step). For the other 40 %, most of them require only < 5 extra switches of the grasp point.

We made 2 supplementary plots to visualize the frequency of switching grasp points in our simulation experiments. Figure 10 shows the switch grasp point rate per step. We can see that the switch grasp point rate gets lower among the first few steps - the consistency increases with multi-modality decreased. But it increases slightly at the later stages - possibly meaning that occlusions could happen and instablize the grasp point predictions. Figure 11 shows the distribution of the number of switch grasps across the dataset. We can see that about 60% of objects only require one grasp action (the first step). And for the other 40%, most of them require only <5 extra switches of the grasp point.

E.3 Step Number Analysis

During a roll-out with limited number of steps (in our case, 30 steps), failures can happen either because of bad prediction quality that will never succeed, or a less efficient strategy or grasp point that might eventually succeed given enough steps. To further analyze the time dependency of the success rate we measured, we made a plot of the opening success rate at each time step.

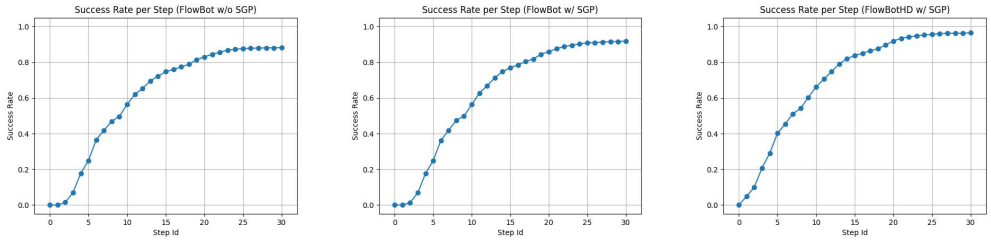


Figure 12: Success Rate over Timestep: We can see that both baseline with and without switch grasp policy (SGP) and FlowBotHD (ours) have nearly reached a plateau after 25 steps. We infer that the performance is not much limited by the 30 step timesteps constraint.

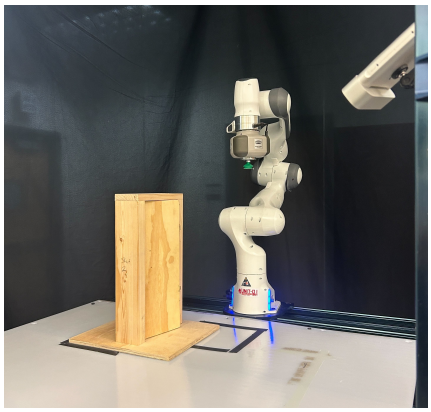


Figure 13: Real World Workspace: This picture depicts the ambiguous door, Franka robot, and Azure Kinect camera.

We can see in Figure 12 that both FlowBot3D (baseline) with and without switch grasp policy (SGP) and FlowBotHD (ours) have nearly reached a plateau after 25 steps, indicating that further timesteps after 25 would likely have limited performance increase. In our paper, we allow each method 30 timesteps to open the object before we compute the success rate for our analysis. Therefore we infer that most of the remaining failures are the result of incorrect predictions and the performance is not much limited by the 30 timestep constraint.

F Real-world Equipment

F.1 Hardware

In the real-world demos, we deploy our model on a Franka Emika Panda Robot. We obtain point cloud input data from an Azure Kinect Depth Camera. The robot's end effector is a Schmalz Cobot Pump with a suction cup 3 cm wide in diameter.

F.2 Workspace

We constructed our workspace in a 1.3 m by 1.3 m by 1.1 m space. The Azure Kinect Camera was placed so that it pointed toward the center of the workspace at an angle that allowed the door to be seen clearly. To avoid collisions with the table when motion planning, we added a box representing the table top using MoveIt's collision box construction tool.

F.3 Foreground Segmentation

To isolate the objects in the point cloud, we had to programmatically segment the table, background, and robot points.

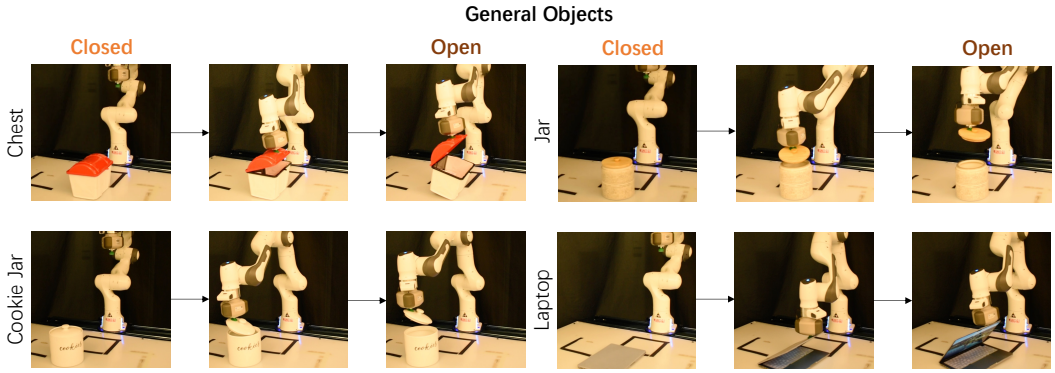


Figure 14: Demonstration of FlowBotHD opening general articulated objects in real-world.

To segment out the table and background, we threshold the x, y, and z values of the points to effectively crop all points outside of the box where the object would be. For example, all points with a z value below 0.02 were removed, as that marks the top of the table.

The robot points are filtered out in real-time using a ROS package called Robot Body Filter that takes in the robot's URDF file. The filter assumes perfect calibration of the camera, which leads to some trailing points from the robot occasionally remaining in the scene. We filter out those points using outlier removal since they are sparser than the object's points.

F.4 Robot Control Paradigm

The robot is controlled using position control by inputting the desired end-effector pose to MoveIt, which would then motion plan and execute the path.

To make contact, the robot aligns itself with the chosen grasp point and estimated flow direction, then moves to a point 10 cm in the flow direction away. Then, a function is called to start the pump, and the robot moves in 2 cm increments towards the contact point. Another function detects when a vacuum has successfully been established, meaning the robot can progress to opening the object.

For each step, the robot moves 5 cm in the direction of the respective predicted flow vector. After the robot attempts to execute the initial step, we use the switch grasp policy to determine if it should attempt to remake contact. To remake contact, the robot moves back to its initial starting pose, then repeats the process described above. The robot can remake contact up to 5 times.

F.5 Custom Multi-modal Door

We custom built a multi-modal door out of plywood, with four pairs of hinges. This allows for it to be configured to open in all four of the possibilities of a standard door (forward to the right, forward to the left, backward to the right, and backward to the left). The door's current configuration is determined by two thin allen key-shaped metal rods, which connect a pair of hinges. Those hinges are then effectively in use and determine how the door opens. The door's frame is 25 cm by 8.5 cm by 43.8 cm, with a 31.75 cm by 36 cm by 1.25 cm stabilizing base. The door itself is 16.83 cm by 1.75 cm by 43.5 cm, with nothing on its front or back face to differentiate which way it opens. To ensure that it fits within our workspace, we built it to be much smaller than a normal door, then scaled the point cloud up by three before passing it into the model.

Given the custom made door, our model has the ability to make multi-modal predictions and open the door in all 4 different ways: pull left, pull right, push left and push right. It also has the ability to actively switch grasp point when the initial prompts fail to align with the current configuration. In all the experiments, the action of 'push' is executed as pulling from the back.