Predicting Motion Plans for Articulating Everyday Objects

Anonymous Author(s)
Affiliation
Address
email

Abstract

Mobile manipulation tasks such as opening a door, pulling open a drawer, or lifting a toilet seat require constrained motion of the end-effector under environmental and task constraints. This, coupled with partial information in novel environments, makes it challenging to employ classical motion planning approaches at test time. Our key insight is to cast it as a learning problem to leverage past experience of solving similar planning problems to directly predict motion plans for mobile manipulation tasks in novel situations at test time. To enable this, we develop a simulator, ArtObjSim, that simulates articulated objects placed in real scenes. We then introduce IIK+θ_0, a fast and flexible representation for motion plans. Finally, we learn models that use IIK+θ_0 to quickly predict motion plans for articulating novel objects at test time. Experimental evaluation shows improved speed and accuracy at generating motion plans than pure search-based methods.

1 Introduction

As humans, when faced with everyday articulated objects as shown in Figure 1, we draw upon our vast past experience to successfully articulate them. We know to stand on the side as we pull open a oven, and where to lean on a door to push it open. Very rarely do we pull open a door onto our feet, or bump into the toilet while lifting a toilet seat. In this paper, we develop techniques that enable robots to similarly use past experience to *mine* and quickly *predict* strategies for articulating everyday objects in cluttered real environments.

Current work on articulating objects casts it as a motion planning problem: given a full scan of the environment, find a robot joint trajectory that leads the end-effector to track the trajectory that the grasp-point on the object should follow. This suffers from both a high-sensing cost and a high-planning cost. Building a full articulable 3D reconstruction of the environment for collision checking and planning is expensive and time consuming. At the same time, finding paths that conform to tight constraints on the end-effector trajectory while not colliding with self or surrounding obstacles or the articulating object is computationally hard. States that adhere to the given constraint form a measure zero set among the set of all states. This creates issues for sampling-based motion planners which can fail to sample states that satisfy the constraint, or must incur computation cost to project states to the constraint manifold [19, 4].

Rather than re-solving, from scratch, how to open a door every time we encounter one, our proposal is to build a repertoire of strategies based on past experience. This replaces the search in the high-dimensional motion plan space with the much simpler problem of selecting from a small family of good strategies, leading to gains in efficiency. Furthermore, this simpler search can be driven by whatever observation is readily available from on-board sensors through the use of machine learning. Our experiments demonstrate the effectiveness of casting this as a learning problem. Given a single RGB-D observation of an articulated object in cluttered real world scenes and associated end-effector

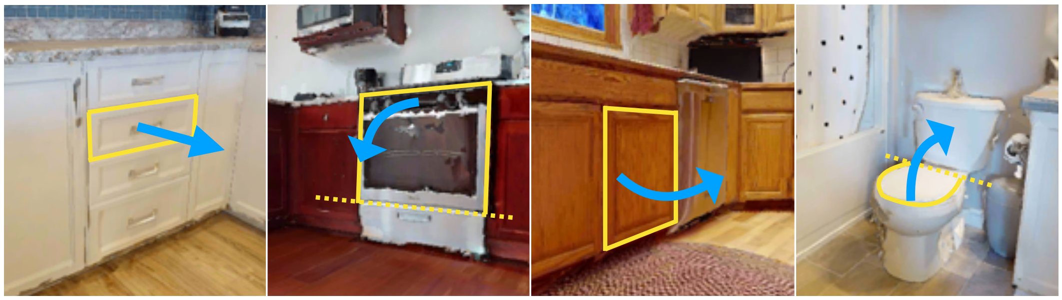


Figure 1: Household robots need to articulate everyday objects (*e.g.* pull open drawers, swing open cupboards, lift toilet seats). Such articulation involves applying forces onto the environment while maintaining relevant contact, such as with the drawer handle as we pull it open. This requires reasoning about the feasibility of the entire trajectory (*i.e.* points along the trajectory should not just be reachable, but it must be possible to continuously go from one point to the next). This paper develops datasets and techniques for learning models that can predict motion plans for such constrained motion planning problems with low sensing and planning costs.

pose trajectory to track, we can output motion plans that track the end-effector trajectory to within 1 cm error with just a few inverse kinematic calls. This, by far, outperforms the constrained motion planning implementation for the projected state space method from the OMPL library [19, 42] which fails to find any motion plans with less than 1cm tracking error even when given 15 minutes of planning time. Our impressive performance is enabled by the following three key innovations.

First, we construct, ArtObjSim, a lightweight kinematic simulator for everyday articulated objects placed in real scenes. Crucially, this simulator is derived from scans of *real-world* environments (from HM3D dataset [35]). This retains the appearance and the cluttered environmental context of the articulated objects. The simulator not only provides the experience to build the repertoire of strategies, but also serves as the first of its kind benchmark for generating plans for articulating objects in real environments. Our dataset consists of 2914 articulated object instances across 4 articulation types (prismatic *e.g.* drawers, vertical hinge *e.g.* cabinets, horizontal up-hinge *e.g.* toilet lids, horizontal down-hinge *e.g.* dishwashers) across 10 object categories in 97 scenes.

Second, rather than predicting a motion plans, that must conform to tight task constraints and are hence hard to directly predict, we instead predict a *strategy* that can be efficiently decoded into a motion plan using the articulation geometry. Our decoding process consists of *synchronously* solving inverse kinematics (IK) problems for end-effector waypoints sampled along the given end-effector trajectory. This synchronization is done by warm starting IK for the t^{th} time-step using solution from the $(t-1)^{\text{th}}$ time-step. We call this decoding process Incremental Inverse Kinematics or IIK. By directly optimizing to reduce end-effector pose error, IIK leads to low tracking errors. The initialization for the first time step, θ_0, serves as the strategy. Changing θ_0 changes the strategy and generates a different motion plan. We find that this representation, IIK+θ_0, is fast (motion plans can be quickly decoded) and flexible (with the right θ_0 it can produce high-quality motion plans for diverse objects in diverse situations).

Not all initializations would work well for all situations. Some might not be able to track the end-effector accurately enough, some may lead to collisions, and others yet might violate the task constraint when joint angles are interpolated for smooth execution. Thus, we need to find good initializations for IIK+θ_0 at test time. Our third innovation, the use of a convolutional neural network to predict good initializations for IIK+θ_0 (or equivalently, good strategies) from RGB image observations, speeds up test time inference. We train this model on a dataset of object images labeled with good initializations, as generated using our proposed ArtObjSim simulator. We are able to find good solutions with only a few IK calls. This is much faster than sampling-based planning at test time which would make tens of thousands of IK calls to project sampled states to the constraint set. We also show that our method can work with predicted end-effector waypoints. Collected dataset, ArtObjSim, and code will be made publicly available upon acceptance.

2 Related Work

Motion planning under constraints [19, 4] has been used to tackle object articulation problems, *e.g.* [36, 7, 34, 5, 27, 6, 44] among numerous other works. Researchers have tackled many aspects: design of task-space regions for expressing constraints on end-effectors [5], planning for base and arm motion separately [27], considering whole-body manipulation [6], reasoning about good locations to position the base through inverse reachability maps [44], and even casting it instead as a trajectory optimization or optimal control problem [9, 33, 40, 28]. All these approaches solve a new object articulation problem, from scratch, every time they encounter one. Consequently, they incur a high sensing and planning costs. Different from these works, our interest is in techniques to leverage experience with similar articulation problems in the past to quickly predict motion plans with low sensing and planning cost. Online system identification approaches [18, 16, 32, 31] that adapt plans using feedback have also been studied.

Perception of articulated objects. A body of work [47, 25, 17, 50, 29, 46, 45, 37, 1, 30, 21, 3, 2] has tackled the perception of *articulation* geometry for articulated objects. Given raw sensory input (RGB images, RGB-D images, depth images, point clouds, or meshes) the goal is to predict articulation parameters: *e.g.* articulation type (prismatic *vs.* hinge), segmentation of parts that independently articulate, axis of rotation / translation, points of interaction. Researchers have a) investigated the use of different input modalities [38, 29, 17, 25], b) built datasets for training models [30], c) designed unified output parameterizations [17], and d) designed novel neural architectures and representation [25, 50]. Researchers have also studied directly predicting sites for interaction [29] and trajectories that the robot end-effector should follow [47] to articulate the object. Our work is complementary, and focuses on converting articulation geometry, possibly predicted from any of these past models, into motion plans.

Simulators for studying object articulation have been challenging to build. Most past efforts use manually created *synthetic* scenes: AI2-THOR [22], Sapeins [49], ManipulaTHOR [8], ThreeD-World [10]. Habitat 2.0 [43] and iGibson [39] improve realism by manually aligning 3D models to real scenes, but are small in size (92 objects in 1 home and 500 objects in 15 homes respectively). Our proposed ArtObjSim simulator is unique in its focus, studying prediction of motion plans for everyday articulated objects, and scale, having 2900 articulated objects spread across 97 unique real world scenes. To our knowledge, ArtObjSim is the largest dataset, to date, for the study of motion planning performance for articulating everyday objects in everyday scenes.

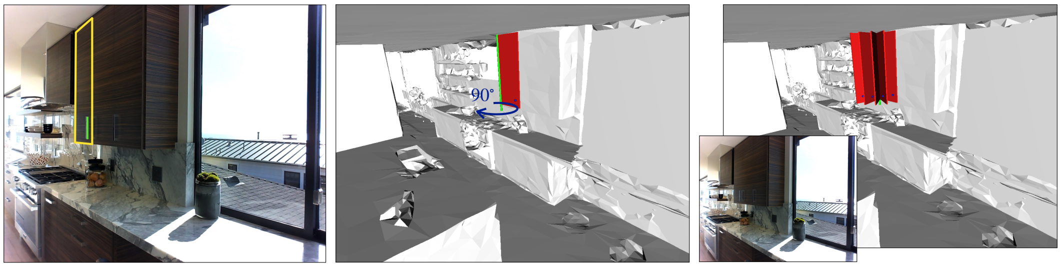
End-to-end RL approaches can also be used to leverage prior experience for fast execution under partial information at test time [24, 48, 13]. However, the large sample complexity of learning policies through RL and the small number of environments available for training has prevented past works to show generalization results in novel environments. By leveraging classical components and scaling up learning, we are able to learn models that generalize to novel objects.

Learning for motion planning has been used to reduce the runtime of motion planning algorithms: [14, 41, 15]. Strudel *et al.* [41] learn obstacles representations for motion planning, while Ichter *et al.* [15, 14] use learning to bias sampling of states for motion planners. Our use of learning is similarly motivated, but we learn to predict low-dimensional strategies (that can be decoded into full motion plans) for constrained motion planning problems from visual input.

3 ArtObjSim: A Simulator for Everyday Articulated Objects in Real Scenes

We introduce ArtObjSim, a lightweight kinematic simulator for articulated objects placed in real scenes. ArtObjSim is built upon the HM3D dataset [35]. HM3D consists of 3D scans of real world environments. It offers both, realistic image renderings from real scenes, and access to the underlying 3D scene geometry. ArtObjSim is made possible through 2D annotations of articulation geometry on images, which are then lifted to 3D to allow for a kinematic simulation of the articulated objects. To our knowledge, ArtObjSim is the first simulator that enables a systematic large-scale study of articulation of everyday objects in real world environments. We describe the steps involved in the construction of ArtObjSim.

Annotating Articulation Geometry on Images. The first step is to annotate 2D articulation geometry on images. 2D articulation geometry includes marking the extent, axis of articulation, articulation type, and interaction locations (handles). We collect annotations in two phases.



a) RGB Image with annotation for cupboard face and handle.

b) 3D scan associated with the RGB image, with annotation for articulations overlaid.

c) Lightweight kinematic simulator with real world object placement, clutter & appearance.

Figure 2: Simulator development. (a) We annotate RGB images inside 3D scans with 2D articulation geometry. (b) This is lifted to 3D using the underlying 3D geometry. (c) As a result we get simulators that can simulate articulated objects in realistic scenes.

In the first phase, we manually walk through the HM3D scenes to find kitchens and bathrooms and identify locations that show articulation objects. We render out images from different viewpoints from these locations for labeling.

In the second phase, we use an annotation service to obtain the necessary 2D labels. We obtain annotations for the segmentation mask for the front face, handle locations, and articulation type (prismatic *vs.* left hinge *vs.* right hinge t’op hinge *vs.* bottom hinge). See Figure 2(a) for an example annotation. For most rectangular objects (*e.g.* drawers, cupboards, refrigerators) these three together with the underlying 3D information from the mesh are sufficient to deduce the axis of articulation. This doesn’t work for toilets and we get additional labels for the axis of rotations (location where the lid is attached). Toilet lids also don’t have handles, we annotate and use the lid tip as the interaction point.

We manually verify the annotation quality after each phase and fix or reject bad annotations. The annotation procedure is fast and cost effective ($0.5 per object instance).

Extracting 3D Articulation Geometry from 2D Annotations. We use the collected 2D annotations, combined with the 3D scene geometry, to obtain a 3D simulation for each articulated object. For each object, we fit a plane to the points within the segmentation mask on the depth image corresponding to the RGB image. This gives us a 3D representation (a 3D rectangle) for the object face that will undergo articulation. We project the 2D handle location onto this 3D plane to obtain the 3D handle location. Articulation parameters are obtained from this 3D representation. We assume that the prismatic objects pull out perpendicular to the face, and the hinged objects rotate about the corresponding edge (top, bottom, left or right) of the 3D rectangle. As noted, toilet lids can’t be approximated as rectangles. We project the annotated 2D axis to the 3D plane. All annotations are converted into the mesh coordinate frame using the transformations for the camera used to render out the image. This defines all that we need to simulate the articulating object in 3D, see Figure 2(c).

ArtObjSim Simulator. As a result of the above two steps, we obtain kinematic simulations for thousands of unique object instances placed in real 3D scenes. Not only can we can simulate the object (*i.e.* how the collision geometry will change as the object articulates or how will the end-effector need to move), we also have a sense of the surrounding 3D geometry of the scene (*i.e.* the counter below the cabinet), and can render out the RGB appearance of the object from multiple different views.

Table 1 shows dataset statistics. The dataset is diverse with close to 3000 object instances from across 97 scenes across 10 object categories and 4 articulation types. The dataset also includes a large geometric variety *e.g.* cabinets high up above the counter and oven drawers very close to the ground. This diversity, along with the fact that these objects are immersed in real scenes makes up problem instances which have not been tackled extensively in the literature.

In Section 4, we will use ArtObjSim to design, train, and evaluate models for predicting motion plans for articulating everyday objects. However, we anticipate ArtObjSim will be useful for many other tasks. For example, predicting articulation parameters or end-effector waypoints from RGB images, or for mining statistics about placement of articulated objects in kitchens to build generative models for scene layout, or for building policies for mobile manipulation.

Table 1: **Statistics for objects and scenes in Simulator for Everyday Articulated Objects in Context (ArtObjSim).**

	Train	Val	Test	Total
# Scenes	70	17	10	97
# Unique Object Instances	2137	459	318	2914
# Prismatic (*e.g.* Drawer)	719	137	107	963
# Vertical Hinge (*e.g.* Cabinet)	1255	282	188	1725
# Horizontal Down-hinge (*e.g.* Oven)	163	40	23	226
# Horizontal Up-hinge (*e.g.* Toilet lid)	70	12	14	96

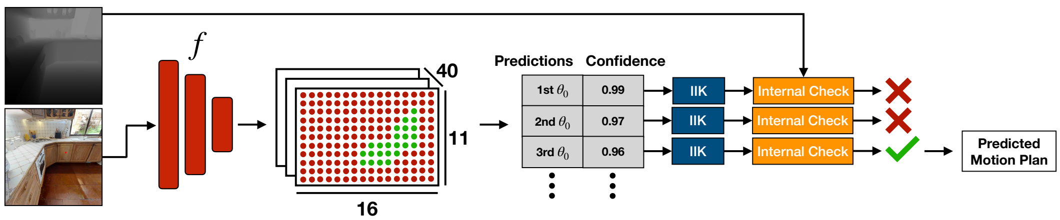


Figure 3: Overview of MPAO (Motion Plans for Articulating Objects). Given an RGB-D image of the object to be articulated (denoted with a red marker), we use a CNN to predicts good initializations for incremental inverse kinematics (IIK). IIK uses end-effector trajectories to generate motion plans corresponding to each returned high-scoring initializations. Generated plans are tested for deviations from the intended trajectory, and collisions using the depth image. The first plan that succeeds these internal checks is returned.

4 Representing and Predicting Motion Plans

Given a single RGB-D image pair $[I, D]$ of an articulated object, and a sequence of end-effector poses necessary to articulate the object $[\ldots, w_t, \ldots]$, our next goal is to predict a motion plan, *i.e.* sequence of joint angles $[\ldots, \theta_t, \ldots]$ that bring the end-effector in the necessary pose to conduct the desired articulation. Rather than re-solving each new problem instance from scratch using motion planning under partial information, we pursue a machine learning approach that leverages past experience to directly predict motion plans. A straight-forward application of machine learning doesn't work as the predicted plans need to satisfy tight task constraints. Instead, we use machine learning to predict a *strategy* which is decoded into a complete motion plan that adheres to the task constraints at hand. We first describe what strategies are and how they are decoded in Section 4.1 and then describe how we use them to predict motion plans from RGB images in Section 4.2.

4.1 Representing and Decoding Motion Plans

We represent motion plans as the initialization of a deterministic gradient-based solver that optimizes joint angles to get the end-effector in the desired pose.

Our motion plan representation builds upon numerical inverse kinematics methods [26]. Inverse kinematics (IK) is the process of obtaining joint angles that get the end-effector to a given desired pose. Starting from some initial joint angles, a numerical IK solver iteratively updates the joint angles using the Jacobian of the forward kinematics till a solution is found. As we are interested in not one but a sequence of joint angles that track the given end-effector trajectory, we *incrementally*

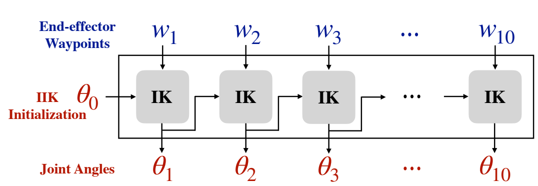


Figure 4: Incremental Inverse Kinematics (IIK). Given an initial configuration (θ_0), and a sequence of end-effector pose waypoints, IIK uses inverse kinematics (IK) to generate configurations that achieve the given end-effector waypoints. IK for subsequent steps is warm-started with IK solutions from the previous time step.

solve a sequence of inverse kinematic problems by initializing the inverse kinematic solver for the t^{th} time-step with the solution from the $(t-1)^{\text{th}}$ time-step. We call this process, Incremental Inverse Kinematics or IIK, and show a block diagram in Figure 4.

Thus, IIK can be viewed as a *deterministic* process that converts a sequence of end-effector waypoints and an initial joint configuration θ_0 into a sequence of joint angles that realize the end-effector poses. θ_0 can be thought of as *knob* that controls the motion plans that IIK generates. Varying θ_0 varies the motion plan generated. We use (IIK, θ_0) as our representation for *strategies* that generate motion plans. Our experiments demonstrate that it is a flexible and efficient way to generate motion plans for articulating everyday objects, and outperforms both unconstrained and constrained motion planning approaches.

Note that IIK+θ_0, shorthand for (IIK, θ_0), may not generate feasible motion plans for all inputs θ_0. Initializing from some θ_0 may not get the end-effector to where we want it to be, others might cause the end-effector to deviate too much from the desired trajectory when interpolating between waypoints, yet others might cause collisions with self or with the environment. We address this issue by *predicting* good θ_0's from the RGB image showing the articulated object as we describe in Section 4.2.

4.2 Predicting Motion Plans from Images

Our next step is to predict good initializations θ_0's for IIK+θ_0 from RGB images. As there can be more than one good θ_0 for each image, we adopt a classification approach. We work with a set of initializations Θ. We train a function $f(I, \theta_0)$ that classifies whether or not the use of θ_0 serves as a good initialization for IIK to achieve end-effector waypoints $\mathbf{w}$ without collisions. We provide details about the initialization set Θ, function f, training data, and loss function to train f.

Initialization set Θ comes from the Cartesian product of a set of robot base positions in $\mathbb{R}^3$ and a set of 10 arm configurations. We use 704 base positions (sampled in a uniform $1m \times 1.5m$ 2D grid of base positions at a 10cm resolution at 4 different heights) and 10 arm configurations, resulting in Θ having 7040 elements. To acquire the 10 arm configurations, we sample 20 random configurations which satisfy the joint limits, and then select the 10 which give us the most successes across the dataset.

Function f is realized through a CNN with an ImageNet pre-trained ResNet-34 backbone [12]. We add 2 fully connected layers on the conv5 output to produce a 7040 dimensional representation. This is reshaped into an 80-dimensional spatial output of size 11×16. This is processed through another 3 convolutional layers to produce a $(10 \cdot 4) \times 11 \times 16$ tensor containing 11×16 spatial output logits for each of the 10 arm configurations at each of the 4 heights.

Training labels are generated by decoding each candidate θ_0 into motion plans using IIK, and testing them for end-effector pose deviation, self-collision, collision with the static environment, and collision with the articulating object in our simulator from Section 3. Note that while testing the decoded motion plans, we interpolate between consecutive states to simulate how the plan will be executed in practice. This process generates a binary success label for each of the 7040 candidates in Θ. This is used to supervise the logits predicted by f via a binary cross-entropy loss.

Training details. Each articulated object instance in ArtObjSim comes with waypoints and ground truth labels as described above. We render multiple views for each articulated object to generate 30K images to train the function f.

Our full method, Motion Plans for Articulating Objects (MPAO), uses the learned function f to rank candidate initialization in Θ. We go down the ranked list, decode them into motion plans using IIK, and return the first *feasible* plan (feasible meaning: accurately tracks the given waypoints and also doesn't collide with self or with the geometry visible in the depth image). See Figure 3 for an overview.

5 Experiments

Our experiments evaluate two aspects: a) the flexibility and decoding efficiency of our proposed motion plan representation from Section 4, and b) how effectively can we leverage RGB images to quickly convert end-effector poses to motion plans. For the former, we make comparisons to motion

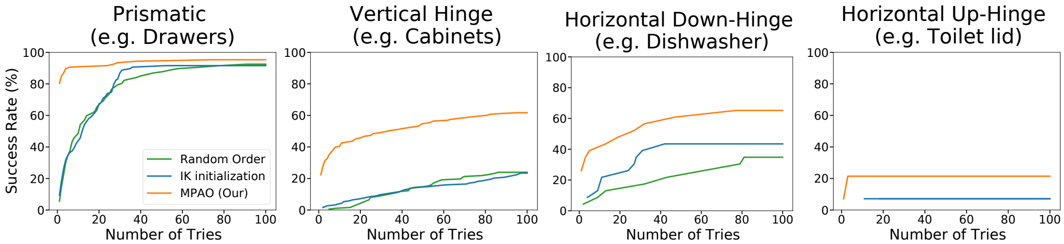
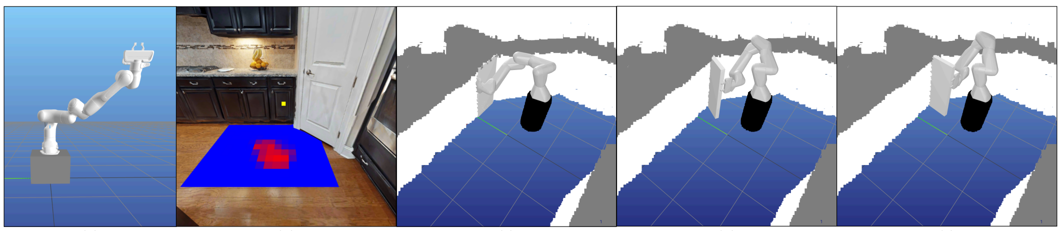


Figure 5: Result plots. We show success rate as a function of number of tries for the different articulation types. Our method, MPAO, that predicts good strategies based on visual input, achieves a higher success rate and generates solutions faster than pure search methods.



(a) (b) (c) (d) (e)

Figure 6: **(a)** One of the ten arm joint configurations from Θ used for initialization. **(b)** Example of a cabinet from the dataset (indicated by the yellow marker), along with predictions for the configuration shown in (a) overlaid onto the image (warmer colors mean higher score). **(c, d, e)** Visualizations of a successful execution from one of the high-scoring locations.

planning, and for the latter we compare against variations that don't use the RGB image. We also evaluate how our method works with predicted end-effector waypoints.

Experimental Setup. We leverage the geometry and appearance of articulated objects in real scenes in our proposed ArtObjSim simulator for evaluation. We adopt the train, val, and test splits as noted in Table 1. All instances from the same scene are in the same set. This allows us to measure how well our models perform on novel held-out object instances. We work with the 7DOF Franka Emika Panda robot. We assume that it can take one of 4 discrete heights (0.25m, 0.5m, 1.0m and 1.5m). While we reason about where the base should be to conduct the motion, we assume that the base remains fixed during execution. Leveraging base motion to better articulate objects is left to future work.

5.1 Motion Plan Representation

We evaluate the flexibility and decoding efficiency of our proposed motion planning representation. More specifically, given a 10 time-step end-effector trajectory and complete collision geometry of the situation, this evaluation measures the quality of the joint angle trajectory produced by our method. We search for a good initialization $\theta_0 \in \Theta$ for IIK and spits out the first solution that doesn't have collisions (to self, surrounding environment, or the articulating object) and conforms to the given tolerance in end-effector pose.

Metric. A predicted trajectory is considered successful if: a) it conveys the end-effector to the goal pose within 1 cm and 0.01 radian, b) the resulting end-effector trajectory violates the task constraint by less than 1 cm in translation and 0.01 radians in rotation for each time step, and c) it doesn't cause collisions with self, the static environment, or the object as it articulates. Before measuring deviations and collision-checking, we linearly interpolate the joint angle trajectories to bring all joint angle changes to ≤ 0.1 radians.

Results. We report the success rate and time taken by our method for different articulation types in Table 2. Prismatic drawers are easy: we can find solutions for 98.5% of the instances to within 0.01 cm, in as little as 2s of compute while only needing to try a median of 15 initializations. Vertical hinged and horizontal down-hinged objects are harder: we are only able to solve 75% instances while also needing to sample many more initializations, taking around 100s. Toilets are by far the hardest because of the tight space in bathrooms.

Comparison with other methods. We also compared IIK+θ_0 to two other class of methods:

Table 2: Motion planning for articulating objects under full information. We measure the success rate and quality of successful plans generated by the different motion planning methods we considered. We note that IIK+θ_0 is able to successfully generate plans quickly. Motion planning, both unconstrained and constrained, obtained a 0% success rate, and hence are omitted from the table, see Section 5.1 for more details.

Articulation Type	Performance		Speed	
	Success %	Deviation (cm)	#inits.	Time (s)
Prismatic (*e.g.* Drawers)	98.5	0.01	15	2.48
Vertical Hinge (*e.g.* Cabinets)	73.8	0.16	306	111.09
Horizontal Down-Hinge (*e.g.* Dishwasher)	75.0	0.28	171	91.32
Horizontal Up-Hinge (*e.g.* Toilet lid)	50.0	0.13	255	432.31

unconstrained and constrained motion planning, neither of which were able to find any successful solutions in a tractable amount of time. For **unconstrained motion planning**, we used RRT-connect [23] to find a path between a start and end joint configuration obtained using inverse kinematics. While this always found a path, without any constraint on the intervening end-effector poses, the path would always violate the 1-DOF constraint imposed by articulated object. This is not surprising as the two poses are quite far from one another. To our surprise, even when these poses are brought close to one another, by sampling 10 way-points along the trajectory, unconstrained motion planning would still only return solutions that would wildly swing the end-effector around. For **constrained motion planning**, we used the projected state space method from the OMPL library [19, 42, 20]. It would find motion plans that conformed to the task constraint to some extent. However, the minimum deviation was 2 cm, much more than the tolerance level needed for our tasks, resulting again in a 0% success rate. We experimented with many different hyper-parameter settings. Some worked better than others, but none were able to return any plans with lower than 2 cm deviations.

In summary, IIK+θ_0 is effective at producing joint angles that conform to a given end-effector trajectory. Finding a solution is still computationally expensive as it requires testing many initializations. We address this using the prediction network f. We evaluate it next.

5.2 Motion Plan Prediction with Known Waypoints

Our next evaluation seeks to measure how quickly and accurately, we can predict motion plans for articulated objects places in novel contexts as observed through RGB-D images. More specifically, given an RGB-D image along with an end-effector trajectory, we measure the success rate of predicting motion plans as a function of planning time. As in Section 5.1, we call a predicted motion plan successful if it reaches the goal while violating the task constraint by less than 1 cm, 0.01 radians and not colliding with self, the environment, and the articulating object. While the metric is the same, the focus of this evaluation is to assess how well methods can cope with partial information from RGB-D observations and their speed of generating solutions.
Comparisons. We compare against other search schemes for finding good θ_0 for IIK. These baseline schemes employ the same overall structure as our method (IIK decoding followed by filtering based on feasibility), but don't use any past experience (learned model) to rank initializations. We consider two variants. *Random Order* uses the same set of initializations Θ as our method, but evaluates them in a random order. *IK initialization* conducts IK with 100 different initializations to generate arm joint angles and base locations that reach the first end-effector waypoint.

Results. Figure 5 presents the success rate for different methods as a function of total number of solutions tried for novel object instances in the test set. Across all articulation types, our method dominates pure search baselines in success rate and speed. We are able to match baseline performance for prismatic joints with $3\times$ fewer tries, and obtain $2.58\times$ the success rate of the baselines for vertical hinges. This establishes the effectiveness of the learned model at predicting good initializations from just RGB image observations. Figure 6 shows an example visualization. We also experimented with a pure imitation learning approach that directly predicts the entire motion plan but weren't able to train a model that generalized to novel instances in preliminary experiments.

5.3 Motion Plan Prediction with Unknown Waypoints

As a proof-of-concept, we have also integrated MPAO into an overall pipeline that doesn't require known waypoints. We experimented with drawers. We adapt Mask RCNN [11] to detect and predict drawer faces (segmentation mask) and handle locations (keypoints) using annotations from ArtObjSim. We converted them into end-effector waypoints using the depth image. This by itself gave an median error of 1.6cm. When using MPAO to track these predicted waypoints, we are able to predict plans that solve 39% drawers to within 1 cm error and 70% to within 5cm error.

6 Conclusion

We pursued a learning approach that uses past experience to quickly predict motion plans for articulating objects. We collected ArtObjSim, a large dataset that enables a kinematic simulation of everyday objects placed in real scenes. We designed IIK+θ_0, a fast and flexible way to represent motion plans under end-effector constraints, and trained neural network models that leverage IIK+θ_0 to quickly predict plans for articulating novel objects.

References

[1] Ben Abbatematteo, Stefanie Tellex, and George Konidaris. Learning to generalize kinematic models to novel objects. In *CoRL*, pages 1289–1299, 2019.

[2] JKT Alexander Andreopoulos and John K Tsotsos. A framework for door localization and door opening using a robotic wheelchair for people living with mobility impairments. In *Robotics: Science and systems, Workshop: Robot manipulation: Sensing and adapting to the real world, Atlanta*, 2007.

[3] Dragomir Anguelov, Daphne Koller, Evan Parker, and Sebastian Thrun. Detecting and modeling doors with mobile robots. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, volume 4, pages 3777–3784. IEEE, 2004.

[4] Dmitry Berenson. *Obeying Constraints During Motion Planning*, pages 1–32. Springer Netherlands, 2018.

[5] Dmitry Berenson, Siddhartha Srinivasa, and James Kuffner. Task space regions: A framework for pose-constrained manipulation planning. *The International Journal of Robotics Research*, 30(12):1435–1460, 2011.

[6] Felix Burget, Armin Hornung, and Maren Bennewitz. Whole-body motion planning for manipulation of articulated objects. In *ICRA*, pages 1656–1662, 2013.

[7] Sachin Chitta, Benjamin Cohen, and Maxim Likhachev. Planning for autonomous door opening with a mobile manipulator. In *2010 IEEE International Conference on Robotics and Automation*, pages 1799–1806. IEEE, 2010.

[8] Kiana Ehsani, Winson Han, Alvaro Herrasti, Eli VanderBilt, Luca Weihs, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. Manipulathor: A framework for visual object manipulation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4497–4506, 2021.

[9] Farbod Farshidian, Edo Jelavic, Asutosh Satapathy, Markus Giftthaler, and Jonas Buchli. Real-time motion planning of legged robots: A model predictive control approach. In *ICHR*, pages 577–584, 2017.

[10] Chuang Gan, Jeremy Schwartz, Seth Alter, Martin Schrimpf, James Traer, Julian De Freitas, Jonas Kubilius, Abhishek Bhandwaldar, Nick Haber, Megumi Sano, et al. Threedworld: A platform for interactive multi-modal physical simulation. *arXiv preprint arXiv:2007.04954*, 2020.

[11] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, pages 2961–2969, 2017.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[13] Daniel Honerkamp, Tim Welschehold, and Abhinav Valada. Learning kinematic feasibility for mobile manipulation through deep reinforcement learning. *IEEE RA-L*, pages 6289–6296, 2021.

[14] Brian Ichter, James Harrison, and Marco Pavone. Learning sampling distributions for robot motion planning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7087–7094. IEEE, 2018.

[15] Brian Ichter, Edward Schmerling, Tsang-Wei Edward Lee, and Aleksandra Faust. Learned critical probabilistic roadmaps for robotic motion planning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9535–9541. IEEE, 2020.

[16] Advait Jain and Charles C Kemp. Behaviors for robust door opening and doorway traversal with a force-sensing mobile manipulator. Georgia Institute of Technology, 2008.

[17] Ajinkya Jain, Rudolf Lioutikov, Caleb Chuck, and Scott Niekum. Screwnet: Category-independent articulation model estimation from depth images using screw theory. In *ICRA*, pages 13670–13677, 2021.

[18] Yiannis Karayiannidis, Christian Smith, Francisco Eli Vina Barrientos, Petter Ögren, and Danica Kragic. An adaptive control approach for opening doors and drawers under uncertainties. *IEEE Transactions on Robotics*, 32(1):161–175, 2016.

[19] Zachary Kingston, Mark Moll, and Lydia E Kavraki. Sampling-based methods for motion planning with constraints. *Annual review of control, robotics, and autonomous systems*, 1:159–185, 2018.

[20] Zachary Kingston, Mark Moll, and Lydia E. Kavraki. Exploring implicit spaces for constrained sampling-based planning. 38(10–11):1151–1178, September 2019.

[21] Ellen Klingbeil, Ashutosh Saxena, and Andrew Y Ng. Learning to open new doors. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2751–2757. IEEE, 2010.

[22] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. AI2-THOR: An Interactive 3D Environment for Visual AI. *arXiv*, 2017.

[23] James J Kuffner and Steven M LaValle. RRT-connect: An efficient approach to single-query path planning. 2000.

[24] Chengshu Li, Fei Xia, Roberto Martín-Martín, and Silvio Savarese. Hrl4in: Hierarchical reinforcement learning for interactive navigation with mobile manipulators. In *CoRL*, pages 603–616, 2020.

[25] Xiaolong Li, He Wang, Li Yi, Leonidas J Guibas, A Lynn Abbott, and Shuran Song. Category-level articulated object pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3706–3715, 2020.

[26] Kevin M Lynch and Frank C Park. *Modern robotics*. Cambridge University Press, 2017.

[27] Wim Meeussen, Melonee Wise, Stuart Glaser, Sachin Chitta, Conor McGann, Patrick Mihelich, Eitan Marder-Eppstein, Marius Muja, Victor Eruhimov, Tully Foote, John Hsu, Radu Bogdan Rusu, Bhaskara Marthi, Gary Bradski, Kurt Konolige, Brian Gerkey, and Eric Berger. Autonomous door opening and plugging in with a personal robot. In *2010 IEEE International Conference on Robotics and Automation*, pages 729–736. IEEE, 2010.

[28] Mayank Mittal, David Hoeller, Farbod Farshidian, Marco Hutter, and Animesh Garg. Articulated object interaction in unknown scenes with whole-body mobile manipulation. *arXiv preprint arXiv:2103.10534*, 2021.

[29] Kaichun Mo, Leonidas Guibas, Mustafa Mukadam, Abhinav Gupta, and Shubham Tulsiani. Where2act: From pixels to actions for articulated 3d objects. *arXiv preprint arXiv:2101.02692*, 2021.

[30] Kaichun Mo, Shilin Zhu, Angel X. Chang, Li Yi, Subarna Tripathi, Leonidas J. Guibas, and Hao Su. PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding. In *CVPR*, 2019.

[31] Keiji Nagatani and SI Yuta. An experiment on opening-door-behavior by an autonomous mobile robot with a manipulator. In *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, volume 2, pages 45–50. IEEE, 1995.

[32] Günter Niemeyer and J-JE Slotine. A simple strategy for opening an unknown door. In *Proceedings of International Conference on Robotics and Automation*, volume 2, pages 1448–1453. IEEE, 1997.

[33] Johannes Pankert and Marco Hutter. Perceptive model predictive control for continuous mobile manipulation. *IEEE RA-L*, pages 6177–6184, 2020.

[34] L Peterson, David Austin, and Danica Kragic. High-level control of a mobile manipulator for door opening. In *Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000)(Cat. No. 00CH37113)*, volume 3, pages 2333–2338. IEEE, 2000.

[35] Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-matterport 3d dataset (HM3d): 1000 large-scale 3d environments for embodied AI. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[36] T. Ruhr, J. Sturm, D. Pangercic, M. Beetz, and D. Cremers. A generalized framework for opening doors and drawers in kitchen environments. In *ICRA*, pages 3852–3858, 2012.

[37] R. B. Rusu, W. Meeussen, S. Chitta, and M. Beetz. Laser-based perception for door and handle identification. In *International Conference on Advanced Robotics*, pages 1–8, 2009.

[38] Radu Bogdan Rusu, Wim Meeussen, Sachin Chitta, and Michael Beetz. Laser-based perception for door and handle identification. In *2009 International Conference on Advanced Robotics*, pages 1–8. IEEE, 2009.

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

[40] Jean-Pierre Sleiman, Farbod Farshidian, Maria Vittoria Minniti, and Marco Hutter. A unified mpc framework for whole-body dynamic locomotion and manipulation. *IEEE RA-L*, pages 4688–4695, 2021.

[41] Robin Strudel, Ricardo Garcia, Justin Carpentier, Jean-Paul Laumond, Ivan Laptev, and Cordelia Schmid. Learning obstacle representations for neural motion planning. *arXiv preprint arXiv:2008.11174*, 2020.

[42] Ioan A. Şucan, Mark Moll, and Lydia E. Kavraki. The Open Motion Planning Library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, December 2012. `https://ompl.kavrakilab.org`.

[43] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. *arXiv preprint arXiv:2106.14405*, 2021.

[44] N. Vahrenkamp, T. Asfour, and R. Dillmann. Robot placement based on reachability inversion. In *ICRA*, pages 1970–1975, 2013.

[45] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *CVPR*, pages 2642–2651, 2019.

[46] Xiaogang Wang, Bin Zhou, Yahao Shi, Xiaowu Chen, Qinping Zhao, and Kai Xu. Shape2motion: Joint analysis of motion parts and attributes from 3d shapes. In *CVPR*, 2019.

[47] Ruihai Wu, Yan Zhao, Kaichun Mo, Zizheng Guo, Yian Wang, Tianhao Wu, Qingnan Fan, Xuelin Chen, Leonidas Guibas, and Hao Dong. VAT-Mart: Learning visual action trajectory proposals for manipulating 3d articulated objects. *arXiv preprint arXiv:2106.14440*, 2021.

[48] Fei Xia, Chengshu Li, Roberto Martín-Martín, Or Litany, Alexander Toshev, and Silvio Savarese. Relmogen: Leveraging motion generation in reinforcement learning for mobile manipulation. *ICRA*, 2021.

[49] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X. Chang, Leonidas J. Guibas, and Hao Su. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11097–11107, 2020.

[50] Vicky Zeng, Timothy E Lee, Jacky Liang, and Oliver Kroemer. Visual identification of articulated object parts. In *IROS*, pages 2443–2450, 2021.