Predicting Motion Plans for Articulating Everyday Objects

Arjun Gupta Max E. Shepherd Saurabh Gupta
University of Illinois at Urbana-Champaign

1. Introduction

Mobile manipulation tasks such as opening a door, pulling open a drawer, or lifting a toilet lid 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.

2. Method

2.1. 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 [6]. 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. ArtObjSim is diverse with 3758 object instances from across 97 scenes across 10 object categories and 4 articulation types. The dataset contains kinematic simulations for each 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. To our knowledge, ArtObjSim is the first simulator that enables a systematic large-scale study of articulation of everyday objects in real world environments.

2.2. Representing and Decoding Motion Plans

Next, we introduce SeqIK+θ_0, a fast and flexible representation for motion plans. Our motion plan representation builds upon numerical inverse kinematics methods [5]. 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 *sequentially* 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, Sequential Inverse Kinematics or SeqIK+θ_0 (see Figure 1).

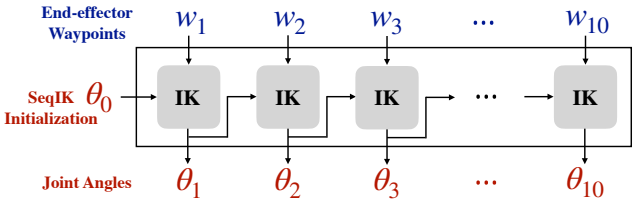

Figure 1. **Sequential Inverse Kinematics (SeqIK+θ_0).** Given an initial joint configuration (θ_0), and a sequence of end-effector pose waypoints, SeqIK+θ_0 uses inverse kinematics (IK) to generate configurations that achieve the given end-effector waypoints. IK for subsequent steps is warm-started with IK solutions from the previous time step.

2.3. Predicting Motion Plans from Images

Finally, we learn a model to predict good initializations θ_0s for SeqIK+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 SeqIK+θ_0 to achieve end-effector waypoints $[\ldots, w_t, \ldots]$ without collisions. The 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. The function f is realized through a CNN with an ImageNet pre-trained ResNet-34 backbone [1]. Training labels are generated by decoding each candidate θ_0 into motion plans using SeqIK+θ_0, and testing them for end-effector pose deviation, self-collision,

collision with the static environment, and collision with the articulating object in ArtObjSim. We then render multiple views for each articulated object to generate 40K images to train the function f.

Our full method, *Motion Plans to Articulate 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 SeqIK+θ_0, 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).

3. Experiments

3.1. 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. SeqIK+θ_0 is able find successful, collision-free motion plans which adhere to the task constraints 99.1%, 63.3%, 71.8%, and 44.7% of the time for prismatic, vertical hinge, horizontal down-hinge, horizontal up-hinge objects respectively.

We also compared SeqIK+θ_0 to two other class of methods: 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 [4] 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. For **constrained motion planning**, we used the projected state space method from the OMPL library [2, 3, 7]. It would find motion plans that conformed to the task constraint to some extent. However, the minimum translation deviation was 0.02 m, much more than the tolerance level needed in our tasks, resulting again in a 0% success rate. We experimented with many hyper-parameter settings. Some worked better than others, but none were able to return any plans with less than 0.02 m translation deviation. In summary, SeqIK+θ_0 is effective at producing joint angles that conform to a given end-effector trajectory.

3.2. 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 RGBD images. More specifically, given an RGBD image along with an end-effector trajectory, we measure the success rate of predicting motion plans as a function of planning time.

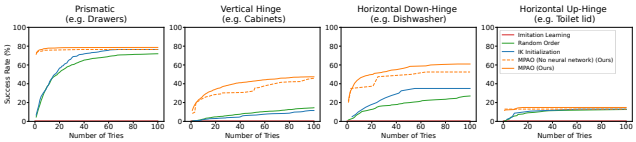

Figure 2. **Motion plan prediction success rate and speed.** We show success rate as a function of number of tries for the different articulation types. Our method MPAO, achieves a higher success rate and generates solutions faster than pure search or pure learning methods. We use 0.01 m translational and 0.01 rad rotational tolerance on the end-effector pose to determine success.

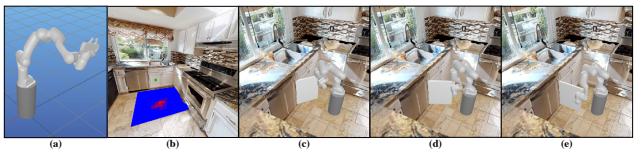

Figure 3. **(a)** One of the ten arm joint configurations from Θ used for initialization. **(b)** Example of an object from the dataset (indicated by the green marker), along with predictions for the configuration shown in (a) overlaid onto the image (warmer colors mean higher score). **(c, d, e)** Visualizations of a successful execution.

We compare against other search schemes for finding good θ_0 for SeqIK+θ_0. These baseline schemes employ the same overall structure as our method (SeqIK+θ_0 decoding followed by filtering based on feasibility), but don't use any past experience (learned model) to rank initializations. **Random Order** uses a random order to sort the set of initializations Θ for each object rather than using our learned function f. **IK initialization** uses IK to find not only the joint angles but also the base location for the first waypoint. After this point, SeqIK+θ_0 is used to obtain a trajectory with a fixed base position, just as for our method. **MPAO (No neural network) (Ours)** ranks initializations in Θ by their success rate on the training set. Though this doesn't use the neural network, it is still *data-driven* in that it leverages experience with past constrained motion planning problems to output plans. We also compare to **Imitation Learning**, a pure machine learning approach that uses imitation learning to directly predict motion plans.

Results. Figure 2 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. For all categories, we are able to match baseline performance with $10\times$ fewer tries, and obtain more than 25% absolute improvement in success rate for vertical and horizontal down hinges. Our full method, MPAO, boosts performance further and is able to effectively leverage the RGB observation to improve the ranking among solutions. These experiments together establish the effectiveness of our method at predicting good motion plans. Figure 3 visualizes a sample θ_0, a heatmap of predictions by our model f, and an example motion plan by MPAO. [1]

[1] Full paper appearing at ICRA 2023: https://arjung128.github.io/mpao/

References

[1] 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. 1

[2] 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. 2

[3] Zachary Kingston, Mark Moll, and Lydia E. Kavraki. Exploring implicit spaces for constrained sampling-based planning. 38(10–11):1151–1178, Sept. 2019. 2

[4] James J Kuffner and Steven M LaValle. RRT-connect: An efficient approach to single-query path planning. 2000. 2

[5] Kevin M Lynch and Frank C Park. *Modern robotics*. Cambridge University Press, 2017. 1

[6] 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. 1

[7] 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. 2