2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Planning Using Belief Summaries for Goal-Directed Manipulation of Articulated Objects with Force and Proprioception

Thavishi Illandara[1], Michael Hagenow[2], and Julie A. Shah[1]

***Abstract*— Enabling robots to manipulate articulated objects is essential for their successful integration into human-centric environments. Such manipulation is often part of a larger multistep task, where achieving a specific joint configuration is necessary for subsequent actions—for example, in a cluttered environment, a cabinet door must be rotated to a precise angle that creates just enough clearance to retrieve an object, beyond which it would collide with surrounding obstacles. In this work, we present an approach to learning goal-directed policies for articulated object manipulation using force and proprioceptive feedback. We formulate the manipulation problem as a Partially Observable Markov Decision Process (POMDP) with a continuous state space and a set of low-level control actions. Due to the limitations of standard POMDP solvers in this setting, we introduce Planning using Belief Summaries (PuBS), which approximates the POMDP as a Markov Decision Process (MDP) over compact particle-filter belief summaries encoding estimated state and uncertainty. This approximate MDP is then solved using reinforcement learning techniques to learn goal-directed policies that enable safe exploration while efficiently guiding the object toward the goal. We evaluate our approach through simulation and real-world robotic experiments, demonstrating reliable goal-reaching performance.**

I. INTRODUCTION

Articulated objects—objects that consist of one or more rigid parts or "links" connected by joints—are ubiquitous in human-centric environments such as homes, workplaces, and factories; therefore, enabling robots to manipulate these objects is critical for performing everyday tasks [1]. However, such manipulation presents significant challenges. For instance, generating relative motion between an object's rigid parts requires control strategies that satisfy the constraints imposed by its limited degrees of freedom. These control strategies should also be adaptive and safe, avoiding damaging contact forces at joint limits while exerting sufficient effort to overcome unique features like magnetic catches or stiff detents. Moreover, manipulating articulated objects is often only a single step within a multistep task, requiring robots to achieve specific joint state configurations as a prerequisite for subsequent actions. For example, the retrieval of an object from a drawer is feasible only if the robot achieves a sufficient opening distance during manipulation. These factors, ranging from satisfying joint constraints to achieving goal-directed behavior, contribute to the complexity robots face when interacting with articulated objects. Within the broader context of articulated object manipulation, our work focuses on the subproblem of goal-directed manipulation, particularly in the post-grasp phase. Here, goal-directed manipulation refers to the task of moving an articulated object to a desired configuration of its joints.

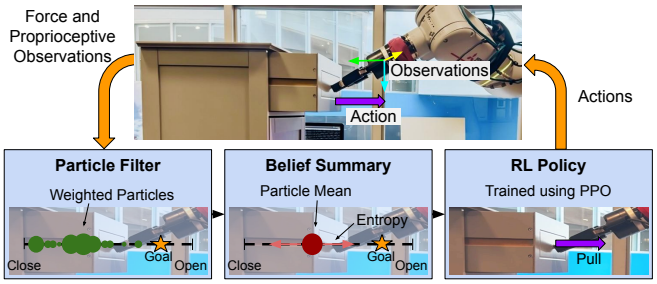

Fig. 1. Overview of the proposed approach for goal-directed manipulation. A particle filter estimates the joint state, which is summarized into a compact belief representation used by a learned policy to reach the goal.

Existing methods enabling efficient interaction with articulated objects have leveraged various sensory modalities, both visual and non-visual, to understand unique and standard features of an object's underlying kinematic model. While current research increasingly relies on vision to understand an object's joint dynamics and constraints [2]–[6], our work utilizes force and proprioceptive data. This choice is motivated by several factors, including the well-known challenges in vision, such as occlusion, poor lighting, and perception noise. Additionally, the visual perception of articulated objects can often be ambiguous, as objects with different internal constraints may exhibit similar external appearances [3], [7]. In contrast, non-visual modalities can accurately capture an object's articulation in contact-rich settings. Furthermore, interaction forces provide reliable cues to the joint state, revealing proximity to limits and features such as detents or locks. However, they cannot directly observe an object's current joint configuration, introducing the challenge of partial observability. Thus, relying on force and proprioception requires the robot to interleave goal-directed actions with informative actions that refine its state estimate. Moreover, these actions must dynamically respond to interaction forces, ensuring safe exploration near joint limits while allowing sufficient force exertion to overcome potential locking mechanisms.

To this end, we formulate goal-directed manipulation as a Partially Observable Markov Decision Process (POMDP), using force and proprioceptive sensing as observation modalities. Solving this continuous-state POMDP, however,

*This material is based in part upon work supported by the National Science Foundation under Grant No. 2330040, the Engineering Research Center (ERC) for Human AugmentatioN via Dexterity (HAND).

[1]Thavishi Illandara and Julie A. Shah are with MIT CSAIL, USA. {thavishi, julie_a_shah}@csail.mit.edu

[2]Michael Hagenow is with the Department of Computer Sciences, University of Wisconsin–Madison, USA. hagenow@cs.wisc.edu

presents a significant challenge for real-time robotic control. Traditional online solvers, while capable of handling continuous spaces, often have planning times unsuitable for the low-level control required for our manipulation problem. Conversely, offline solvers that offer fast execution times are typically computationally infeasible for such domains. To bridge this gap, we introduce the **Planning using Belief Summaries (PuBS)** framework that approximates the POMDP as an approximate Markov Decision Process (MDP) over low-dimensional belief summaries, enabling the use of reinforcement learning (RL) techniques for policy training (Figure 1). We instantiate this framework in a closed-loop modular system that uses a particle filter for belief estimation and an RL-trained policy for action selection.

The primary contributions of this work are threefold. First, we propose the PuBS framework, a novel offline, model-based approach to solving continuous POMDPs while supporting real-time robotic control. Second, we apply this framework to the problem of goal-directed articulated object manipulation, developing a system that includes a specialized discrete action space to support both compliant and high-force interaction modes, as well as a reward function that encourages safe and efficient behavior. Finally, we validate our approach through comprehensive experiments, demonstrating high success rates across the Light Dark POMDP benchmark and several real-world articulated object manipulation tasks.

II. RELATED WORK

A. Articulated Object Manipulation

Articulated object manipulation is inherently complex, with prior work developing end-to-end systems [4]–[6], [8] or addressing specific subproblems such as kinematic model learning [2]–[4], [9]–[11] or generalizable manipulation strategies [4], [5], [12]–[17]. With advancements in vision-based machine learning and the emergence of large-scale datasets, such as the PartNet-Mobility dataset [18], recent literature has increasingly relied on vision for articulated object manipulation, either by inferring explicit articulation parameters for planning [2]–[4] or by directly mapping visual observations to actions via imitation [19], [20] or reinforcement learning [5], [14], [21]. While some studies develop end-to-end systems designed for reaching goal configurations [4], [5], [8], these methods often require additional mechanisms to ensure sufficient visual observations to verify task success. For instance, one study requires the robot to return to its initial viewpoint for image-based verification [5], whereas Xie et al. [8] engineer the reward function to maintain visual contact with the moving part. To address these limitations, we utilize force and proprioceptive data, signals inherently available during contact, to enable goal-directed manipulation through joint state estimation.

While non-visual sensor modalities have been incorporated into vision-based methods to mitigate challenges such as visual ambiguity [4], [11], [17], researchers have also explored using force, proprioception, and tactile sensing alone to perform post-grasp articulated object manipulation. One avenue within this research leverages these sensors' capabilities to infer features unique to a given object, such as joint parameters [9], [10] or environmental constraints [15]. Another research direction learns generalizable manipulation strategies by leveraging sensor feedback during manipulation [7], [12], [13], [16]. However, these methods generally do not support goal-directed manipulation. Instead, task termination is achieved using heuristic strategies, such as issuing an external stop command [12], reaching a joint limit [15], or manipulating to a predefined distance [7]. To address this gap, we explore the use of force and proprioceptive feedback to enable goal-directed behavior, eliminating the need for vision or heuristic-based stopping conditions.

B. POMDP Solvers in Manipulation

The POMDP is a mathematical framework widely used in manipulation research to handle decision-making under uncertainty resulting from partially observable states. Existing approaches to solving POMDPs are traditionally classified as offline or online solvers [22], while more recent advances have introduced model-free reinforcement learning methods as an alternative approach.

Traditional offline POMDP solvers that precompute policies enable fast action lookups during execution, making them well-suited for low-level motion control. However, both exact [23] and approximate algorithms [24], [25] are typically designed for finite-dimensional belief spaces, limiting their applicability in domains with continuous state spaces. While discretization is a common strategy to adapt these solvers to continuous spaces, designing an effective discretization scheme involves a critical tradeoff between computational feasibility and solution accuracy [22]. On the other hand, online POMDP solvers, such as [26]–[28], are commonly used in robotics due to their ability to handle large continuous state spaces. However, despite algorithmic modifications to improve efficiency, planning times continue to be significant, limiting their suitability for low-level control.

Recent work has explored the use of reinforcement learning (RL) to solve POMDP problems in robotics. While RL can be applied online via on-policy methods, it is more commonly utilized as a model-free offline solver by leveraging simulation environments to compute policies before execution. For instance, Xie et al. [8] utilizes RL in simulation to learn policies that enable articulated object manipulation in real-world settings, given visual and proprioceptive observations. As most RL algorithms are formulated under the MDP assumption, their application to POMDPs, especially those requiring informative actions to reduce uncertainty, demands careful design decisions, such as employing recurrent neural networks (RNN) as function approximators [29].

III. METHODOLOGY

Our work focuses on enabling a robot to manipulate an articulated object from an unknown initial configuration to reach a goal configuration, given its kinematic model and initial direction of motion. We frame this problem as a POMDP using force and proprioceptive feedback as observations. In this section, we first review the POMDP

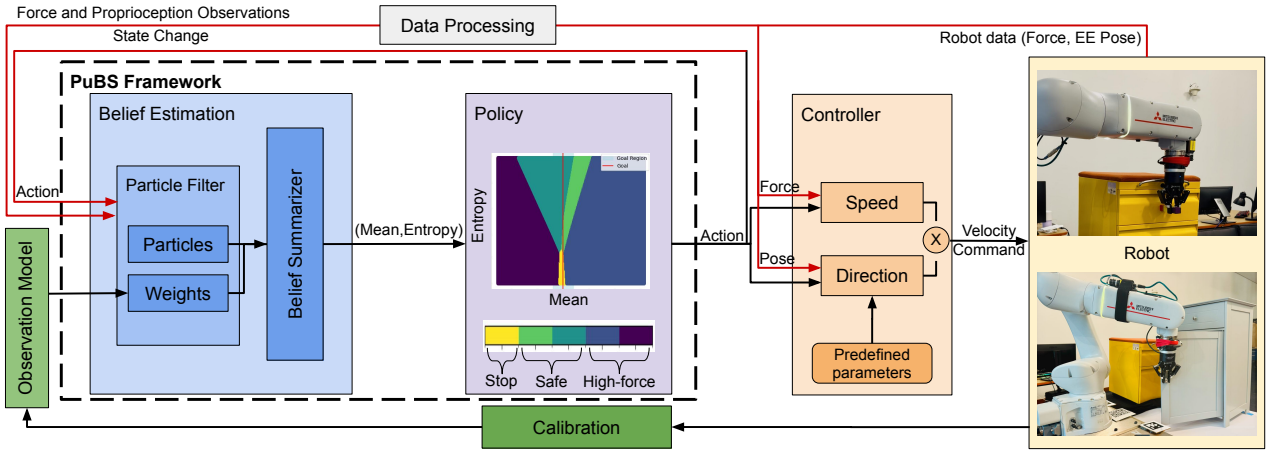

Fig. 2. Modular closed-loop PuBS system for goal-directed articulated manipulation. Force and proprioceptive observations drive actions that become robot velocity commands. A calibration-learned observation model supports belief estimation. Red arrows indicate feedback signals.

formulation of the manipulation task. We then introduce the PuBS framework and describe how belief summaries are constructed and used for policy learning. Finally, we detail the system implementation that realizes this framework in a real robotic setting, as illustrated in Figure 2.

A. Goal-Directed Manipulation as a POMDP

Articulated object manipulation encompasses several subproblems, such as kinematic model inference, motion inference, grasping, and achieving a goal joint state configuration. To limit the scope of this work, we focus on post-grasp manipulation of an articulated object with a predefined kinematic model (joint type, joint limits, and radius if revolute) and a known initial direction of motion. We formulate this goal-directed manipulation problem as an infinite-horizon POMDP, **M**, defined by the 7-tuple $(S, A, O, T, Z, R, \gamma)$.

1) State space: We focus on objects with a single revolute or prismatic joint having a single degree of freedom. Therefore, the state space $S = [s_{min}, s_{max}] \subseteq \mathbb{R}$ is a one-dimensional continuous state interval defined by the joint limits s_{min} and s_{max}.

2) Action space: Effective manipulation of articulated objects requires control strategies that can adapt to diverse physical interaction regimes. For instance, a robot must be compliant and exhibit high force sensitivity to safely identify hard joint limits, yet be capable of exerting significant forces to overcome mechanical locking mechanisms or detents. To address this dichotomy, we design a discrete action space that enables the policy to switch between different interaction modes for adaptive and safe exploration. Each mode runs at its maximal safe velocity, so continuous speed control would only introduce slower variants without performance gains.

Our discrete action space A consists of five actions: (1) Stop (a_0), (2) Safe Mode, move in opening direction (a_1), (3) Safe Mode, move in closing direction (a_2), (4) High-force mode, move in opening direction (a_3), and (5) High-force mode, move in closing direction (a_4). Each action is a tuple specifying an interaction mode (safe or high-force) and a direction (open or close). In the safe mode, the robot stops if the sensed force exceeds a safety threshold, preventing unsafe interactions. Conversely, the high-force mode disables this safety constraint, allowing motion through resistance when needed. A separate stop action sets the velocity to zero, signaling task completion.

3) Observation space: An observation vector is defined as $o = [f_d, f_z, f_n, m_p]^T \in \mathbb{R}^3 \times \{0, 1\}$. Here, f_d, f_z, and f_n are forces measured along the direction of motion, vertical axis aligned with gravity, and the axis normal to f_d and f_z. The proprioceptive observation is $m_{p,t} = \mathbb{1}[d_t - d_{t-1} \neq 0]$, where $\mathbb{1}[\cdot]$ denotes the indicator function and d represents the end-effector pose. Throughout this paper, variables indexed by t refer to their values at discrete timesteps t.

4) Transition and observation distributions: We perform a calibration phase prior to policy learning, during which the robot interacts with the object to collect observation-state pairs, (o, s), as described in Section III-C.1. These data, along with the known dynamics of the constrained manipulation controller, are used to define a model $s', o \longleftarrow G(s, a)$ that can be sampled to estimate the conditional probability distributions $T(s'|s, a)$ and $Z(o|a, s')$ that govern state transitions and observation emissions, respectively.

5) Reward specification: To encourage the learning of a safe and time-efficient goal-directed manipulation policy, we define the reward function as,

$$R(a, s') = R_g(s', a) + R_f(s', a) + R_s(s, a) + R_t, \quad (1)$$

where R_g, R_f, R_s, and R_t denote the goal-proximity reward, failed-stop penalty, safety-violation penalty, and time penalty terms, respectively. The goal-proximity reward incentivizes reaching the precise goal joint state, s_g, by assigning a positive reward proportional to the proximity of the joint state to the goal state at termination ($a = a_0$). We define a goal region, $G = \{s \mid |s - s_g| \leq \epsilon_g\}$, where ϵ_g is the tolerance, and compute the reward as,

$$R_g(s') = \mathbb{1}[a = a_0] \cdot \max(0, 1 - |s' - s_g|/\epsilon_g). \quad (2)$$

The failed-stop penalty term, $R_f(s') = -\mathbb{1}[a = a_0] \cdot \mathbb{1}[s' \notin G]$, penalizes the agent for stopping outside the goal region.

To enforce safe exploration, we define the safety-violation penalty as $R_s(s) = -\mathbb{1}[a \in \{a_3, a_4\}] \cdot \mathbb{1}[(s + \Delta s) \notin S]$ to penalize the robot for taking high-force actions near joint limits. Here, Δs denotes the expected joint displacement due to action a. The time penalty, R_t, is applied at every timestep to encourage timely task completion.

The objective in this formulation is to learn a policy $\pi : O \to A$ that maps observations to actions in a manner that is both safe and time-efficient. Specifically, the policy should guide the articulated object to a goal joint state configuration $s_g \in S$ while overcoming locking mechanisms and avoiding unsafe interactions near joint limits. Formally, the goal is to maximize the expected cumulative reward:

$$\pi^* = \arg\max_{\pi} \ \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \mid a_t = \pi(o_t)\right]. \tag{3}$$

In our setting, this involves leveraging force and proprioceptive feedback to infer the latent joint state and to adaptively switch between safe and high-force control strategies to achieve goal-directed manipulation of articulated objects.

B. Asynchronous Belief and Policy Framework

Solving the goal-directed manipulation POMDP described in Section III-A presents two challenges. First, both the state and observation spaces are continuous, rendering traditional offline POMDP solvers infeasible. Second, robotic control requires low-latency action selection during deployment, which limits the feasibility of online POMDP solvers. To address these challenges, we propose the **Planning using Belief Summaries (PuBS)** framework: a modular, offline model-based approach for solving continuous POMDPs by approximating them as MDPs over low-dimensional belief summaries. PuBS decouples the problem into two asynchronous modules (Figure 2):(1) a belief estimation module that uses a particle filter to maintain a distribution over the latent state and summarizes it using particle-weighted statistics, and (2) a policy module that selects actions using a reinforcement learning policy trained on these compact belief summaries. This separation allows each module to operate at its own frequency, improving its robustness to noisy observations. The remainder of this section describes each module and how they interact to form a tractable solution to a continuous POMDP.

Due to the Markovian property of a POMDP, optimal decisions can be inferred from the conditional probability distribution of the state given the history,

$$b_t(s) \equiv \mathbb{P}(s_t = s | h_t), \tag{4}$$

as given in [30]. This distribution is called the belief of the POMDP, where the history h_t refers to the history of actions and observations,

$$h_t \equiv (b_0, a_0, o_1, a_1, \ldots, 0_{t-1}, a_{t-1}, o_t). \tag{5}$$

Therefore, a POMDP is inherently an MDP on its belief space B, the set of all possible belief states in the POMDP.

Since goal-directed manipulation is framed as a discrete-time POMDP with a continuous state space, the corresponding belief space is infinite-dimensional, making direct computation intractable. To address this, we approximate the belief state using a finite set of weighted particles via a particle filter. Specifically, we employ a sequential importance sampling filter (SIS) with a transition prior proposal to estimate the belief distribution $b_t(s)$.

Let $\{y_{i,t}, w_{i,t}\}_{i=1}^{N}$ be the set of particles and weights representing the estimated belief $\tilde{b}_t(s)$ over the hidden state s. Upon taking action $a_t \in A$ and receiving an observation $o_{t+1} \in O$, the belief estimate is updated using the SIS filter, which consists of a prediction and weight update steps. The prediction step propagates each particle $y_{i,t}$ to a new state $y_{i,t+1}$ by sampling from the transition distribution,

$$y_{i,t+1} \sim T(s' | s = y_{i,t}, a_t). \tag{6}$$

The weight update step first reweighs the particle according to the likelihood of the new observation o_{t+1} to obtain the unnormalized weight,

$$\tilde{w}_{i,t+1} = w_{i,t} \cdot Z(o_{t+1} | s = y_{i,t+1}, a_t). \tag{7}$$

Next, the weights are normalized to ensure they form a valid probability distribution,

$$w_{i,t+1} = \frac{\tilde{w}_{i,t+1}}{\sum_{j=1}^{N} \tilde{w}_{j,t+1}}. \tag{8}$$

Beliefs estimated using particles and weights have enabled approaches like Particle Belief MDPs (PB-MDP), solvable using sampling-based MDP algorithms such as Sparse Sampling - ω [30]. However, accurately approximating the true belief state $b_t(s)$ with a particle filter generally requires using a large number of particles, resulting in high-dimensional spaces. As we employ neural networks to approximately solve an MDP, the high-dimensional input space created by particle-based belief representations makes neural network training computationally challenging and data-intensive [31]. Furthermore, standard neural network architectures are not inherently permutation invariant. They require specialized architectures to process set-based inputs [32]. Therefore, we summarize the belief estimate $\tilde{b}_t(s)$ to obtain a fixed-size, structured representation that abstracts away the specifics of the SIS filter implementation, such as the exact number N or the arbitrary ordering of particles. Specifically, we summarize the belief estimate using the particle mean,

$$\mu_t = \sum_{i=1}^{N} w_{i,t} \cdot y_{i,t}, \tag{9}$$

as a point estimate of the belief, and the Shannon entropy of the normalized particle weights,

$$H_t = -\sum_{i=1}^{N} w_{i,t} \log(w_{i,t}), \tag{10}$$

to quantify the belief's uncertainty.

To learn a policy for the underlying POMDP, we define a surrogate MDP, $\mathbf{M}' = (X, A, T', R', \gamma)$, which can be

solved via Proximal Policy Optimization (PPO) [33], a well-established RL algorithm for approximately solving MDPs. The state space X consists of the low-dimensional belief summaries, $x_t = (\mu_t, H_t) \in \mathbb{R}^2$, while the action space A remains the same as the original action space. The transition probability $\tilde{T}(x_{t+1}|x_t, a_t)$ is not explicitly modeled. Instead, the dynamics over x_t are implicitly defined through the steps of the particle filter, which propagates the particles according to the original POMDP's state transitions T and updates the weights using the observation emissions Z. Defining a reward function $\tilde{R}(x_t, a_t)$ would require manual shaping of the rewards based on the belief summary, such as specifying confidence thresholds for stopping. Instead, we leverage the RL algorithm's access to the true state, s_t, during training to compute rewards directly from the original reward function $R(s, a, s')$. This ensures the agent implicitly learns to balance uncertainty reduction and task performance.

C. System Implementation

We instantiate the PuBS framework for goal-directed manipulation of articulated objects that leverages force and proprioceptive feedback to enable closed-loop decision-making under uncertainty. As illustrated in Figure 2, the architecture consists of three main components: a particle filter for belief estimation, a policy trained via RL that selects actions conditioned on the current belief, and a low-level controller that translates these actions into velocity commands. The system also relies on an observation model learned during a one-time offline calibration phase. This design facilitates both modularity and asynchronous operation. Modularity permits independent updates to components such as the observation model or controller parameters without retraining the policy. Asynchronous execution enables modules to run at independent frequencies. The controller and particle filter operate faster for smooth control and robust belief tracking, while the policy acts on the refined belief summary at a lower rate. The following sections will detail the calibration process and the controller design.

1) Calibration and Observation Model: To learn the observation model for state estimation and policy learning, we run an offline calibration in which an automated script guides the robot to open and close the object between known joint limits repeatedly. It executes the four movement actions with five demonstrations each, while recording force and end-effector poses. Offline, the collected force vectors are decomposed into three orthogonal components f_d, f_z, and f_n. Next, we apply changepoint detection [34] to segment each component signal into regions of distinct variance. Within each segment, we compute variance and obtain a continuous function of the mean by applying the Savitzky-Golay filter [35] and interpolation. The proprioceptive observation m_p is modeled separately by binarizing the magnitude of the force data relative to the safety threshold. Finally, we obtain the observation likelihood function as a product of independent Gaussian distributions:

$$Z(o|s,a) = \prod_{c \in f_d, f_n, f_z, m_p} \mathcal{N}(\mu_c(s,a), \sigma_c^2(s,a)). \quad (11)$$

Here, the mean μ_c and variance σ_c^2 for each force component (f_d, f_n, and f_z) are functions derived from the segment-wise interpolation and variance estimates, respectively. Particles are deterministically propagated by mapping measured end-effector pose changes to the state space and then weighted using the observation likelihood function.

2) Controller Design: As described in Section III-A, we design an action space A that consists of 5 actions that specify an interaction mode (safe or high-force) and a direction (open or close). Each discrete action maps to a 6D velocity command $\xi \in \mathbb{R}^6$ composed of a speed magnitude $v_s \in \mathbb{R}$ and a normalized direction vector $\xi_{dir} \in \mathbb{R}^6$:

$$\xi(a, x_{ee}, F_{ee}) = v_s(a, F_{ee}) \cdot \xi_{dir}(a, x_{ee}, \theta). \quad (12)$$

The speed magnitude depends on the action a and force feedback F_{ee},

$$v_s(a, F_{ee}) = \begin{cases} 0, & \text{if } a = a_0 \\ 0, & \text{if } a \in \{a_1, a_2\} \text{ and } F_{ee} \cdot \xi_{dir} > F_t \\ v_{safe}, & \text{if } a \in \{a_1, a_2\} \text{ and } F_{ee} \cdot \xi_{dir} \leq F_t \\ v_{high}, & \text{if } a \in \{a_3, a_4\}. \end{cases} \quad (13)$$

Here, the dot product $F_{ee} \cdot \xi_{dir}$ measures the force along motion direction and is utilized in safe mode to halt motion if it exceeds the safety threshold F_t. The values v_{safe} and v_{high}, where $v_{safe} \leq v_{high}$, are predefined speeds for safe and high-force modes, respectively.

The direction vector ξ_{dir} is computed using the action-defined direction, the end-effector pose x_{ee}, and object-specific parameters θ (the initial direction of motion vector and the center of rotation for revolute objects). For prismatic joints, the controller generates linear motion based on the initial direction. In the case of revolute joints, it produces arc motion using the initial direction and the center of rotation.

IV. EVALUATION

To evaluate the performance of our proposed framework, we seek to answer the following research questions: (1) How effective is our particle-filter-based policy compared to RL baselines that use raw observations? (2) How does our method compare against state-of-the-art offline POMDP solvers? (3) How critical is uncertainty representation for the policy? (4) Can the learned policy be safely and reliably transferred to a physical robot? In the remainder of this section, we refer to our method as *PuBS*.

A. Experimental Setup and Evaluation Metrics

1) Tasks: We evaluate our framework on six tasks: four using real-world objects (two drawers and two doors, with and without magnetic locks as shown in Figure 3), a simplified articulated object (*Sim Object*), and the 1D Light Dark POMDP benchmark (*Light Dark*) as defined in [30]. To enable comparison with offline POMDP solvers, we design the *Sim Object* task with a discrete state space $S \in [1, 100]$ and a 1D force signal that is high at the joint limits and low elsewhere, creating a worst-case setting with sparse localized information. Goals for the articulated objects included fully

Fig. 3. Real-world articulated objects used in our experiments: (a) *Drawer 1* without a lock, (b) *Drawer 2* with a magnetic lock, (c) *Door 1* without a lock, and (d) *Door 2* with a magnetic lock.

open (*Open*), fully closed (*Closed*), and an intermediate configuration (*Middle*), while the *Light Dark* goal is state 0.

2) Robot Platform: Real-world experiments are conducted on a 6-DoF Mitsubishi MELFA RV-5AS-D Assista robot arm, equipped with a Mitsubishi Electric 1F force-torque sensor. The belief estimation, policy, and controller modules run at 30 Hz, 10 Hz, and 100 Hz, respectively.

3) Simulation Environment and Hyperparameters: We train and simulate in a custom Gym environment with learned observation models and controller-specific dynamics. Policies use PPO (Stable-Baselines3 [36]) with a 3-layer MLP of 64 units, a linear learning rate starting at 1×10^{-4}, and default hyperparameters otherwise. State estimation uses an SIS particle filter without resampling, with 300 particles for real-world tasks and 100 for *Sim Object* and *Light Dark*. Real-world policies are trained for 5×10^5 timesteps, and *Sim Object* and *Light Dark* for 10^6. To stabilize learning on articulated objects, episodes start at a joint limit until 50 successful terminations are observed. Training episodes end on a safety violation or a "stop" action. Evaluation uses deterministic actions with the same terminations and an additional 250-step cap. For each task, we train five seeds, evaluate each over 10,000 randomized trials, report the best seed, and deploy it on hardware. Rewards for articulated-object tasks are 1, -1, -1, and -0.001 for goal proximity, failed stop, safety violation, and time per step.

B. Experiment: Comparison with RL-based Baselines

We compare our method (*PuBS*) on the four real-world articulated tasks under three goal conditions against three baselines: (1) *PPO* removes the particle filter and feeds raw observations into the same policy architecture, (2) *LSTM1* augments *PPO* with a single 256-node LSTM layer to model temporal context, and (3) *LSTM2* adds two LSTM layers of 256 nodes each. These recurrent architectures were the best-performing variants from a broader search that we conducted. We report the average discounted reward ($\gamma = 0.99$) with standard errors (SE), as the time penalty is not fully reflected in the average reward. Additionally, we present the number of safety violations per goal, summed across all tasks.

PuBS is decisively best in the *Middle* goal condition and no baseline falls within its 95% confidence interval (CI) as presented in Table I. In *Open* and *Close*, where joint limit contact signals make raw observations highly

TABLE I

SECTION IV-B REPORTS MEAN DISCOUNTED REWARD ($\pm$SE), AND ENTRIES WITHIN THE BEST METHOD'S 95% CI ARE IN BOLD.

	Drawer1	Drawer2	Door1	Door2
		Middle		
PuBS	$\mathbf{0.898 \pm 0.001}$	$\mathbf{0.808 \pm 0.001}$	$\mathbf{0.718 \pm 0.001}$	$\mathbf{0.717 \pm 0.001}$
PPO	0.833 ± 0.001	0.482 ± 0.003	0.187 ± 0.005	0.705 ± 0.002
LSTM1	0.837 ± 0.001	-0.478 ± 0.001	0.519 ± 0.002	-0.436 ± 0.001
LSTM2	0.850 ± 0.001	-0.441 ± 0.001	0.048 ± 0.006	-0.433 ± 0.000
		Open		
PuBS	$\mathbf{0.805 \pm 0.001}$	$\mathbf{0.748 \pm 0.001}$	$\mathbf{0.458 \pm 0.003}$	$\mathbf{0.477 \pm 0.003}$
PPO	0.726 ± 0.001	0.708 ± 0.001	$\mathbf{0.459 \pm 0.003}$	$\mathbf{0.471 \pm 0.003}$
LSTM1	$\mathbf{0.807 \pm 0.002}$	-0.434 ± 0.000	0.223 ± 0.005	-0.478 ± 0.007
LSTM2	0.661 ± 0.002	0.604 ± 0.002	-0.281 ± 0.004	0.431 ± 0.004
		Close		
PuBS	$\mathbf{0.798 \pm 0.001}$	0.710 ± 0.001	$\mathbf{0.482 \pm 0.003}$	$\mathbf{0.467 \pm 0.003}$
PPO	0.752 ± 0.007	0.712 ± 0.002	$\mathbf{0.480 \pm 0.003}$	0.335 ± 0.004
LSTM1	0.772 ± 0.004	0.738 ± 0.002	0.392 ± 0.003	0.423 ± 0.004
LSTM2	0.665 ± 0.002	$\mathbf{0.745 \pm 0.001}$	-0.544 ± 0.006	-0.160 ± 0.007

TABLE II

SECTION IV-C REPORTS MEAN REWARD ($\pm$SE) AND SUCCESS RATES WITH ENTRIES WITHIN THE BEST METHOD'S 95% CI IN BOLD.

	Light-Dark		Sim Object	
	Reward	Success	Reward	Success
PuBS	$\mathbf{88.7 \pm 0.13}$	99.58%	$\mathbf{0.575 \pm 0.002}$	100.0%
SARSOP	88.1 ± 0.16	99.36%	$\mathbf{0.578 \pm 0.002}$	100.0%
QMDP	-33.2 ± 0.46	13.65%	-0.009 ± 0.00	0.0%

informative, *PuBS* remains top on most tasks with narrower margins, with exceptions at *Door 1-Open* and *Drawer 2-Close*. However, relying on raw observations can increase the risk of safety violations if the policy fails to detect goal completion promptly, as illustrated in Figure 4, where *PuBS* shows markedly fewer safety violations across all goals. In the only challenging cases, *PuBS* records 1, 1, and 10 violations for *Drawer 1-Open*, *Drawer 2-Open*, and *Door 2-Close*, while baselines with positive rewards exceed 100 in the same settings. The lower reward for *PuBS* on *Drawer 2-Close* is due to a conservative near-limit strategy that avoids collisions yet can time out in rare trials, prioritizing safety over speed. To further validate this observation, future work could systematically vary the safety-violation and time-penalty terms in the reward function. We omit *Light Dark* and *Sim Object* here since baselines produce uninformative negative rewards.

C. Experiment: Comparison with Offline POMDP Solvers

We compare *PuBS* with the offline POMDP solvers *SARSOP* [25] and *QMDP* [24] on *Light Dark* and *Sim Object*, omitting real-world tasks as their continuous spaces make *SARSOP* computationally infeasible. Metrics are mean reward with standard error and success over 10,000 trials, with undiscounted rewards reported for *Light Dark* per prior work [30]. While *PuBS* surpasses *QMDP* in both tasks, it outperforms *SARSOP* on *Light Dark* and is within its 95% CI on *Sim Object* (Table II). The small reward gap despite identical success rates on *Sim Object* arises from minor differences in the average number of timesteps. In *Light Dark*,

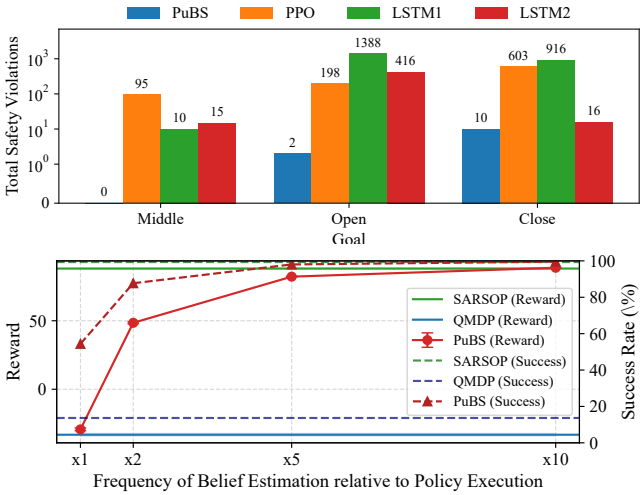

Fig. 4. (Top) Total safety violations per goal, summed across all tasks, for PuBS and RL baselines. PuBS shows fewer violations. (Bottom) Rewards and success rates vs offline solvers as belief-update frequency increases. Faster updates improve PuBS, exceeding QMDP and approaching SARSOP.

TABLE III

SECTION IV-D REPORTS MEAN DISCOUNTED REWARD (±SE), AND ENTRIES WITHIN THE BEST METHOD'S 95% CI IN BOLD.

Task	PuBS	Random H	No H
Drawer 1	$\mathbf{0.898 \pm 0.001}$	0.844 ± 0.001	0.862 ± 0.001
Drawer 2	$\mathbf{0.808 \pm 0.001}$	0.785 ± 0.001	0.782 ± 0.001
Door 1	$\mathbf{0.718 \pm 0.001}$	0.695 ± 0.002	0.696 ± 0.002
Door 2	$\mathbf{0.717 \pm 0.001}$	0.706 ± 0.002	0.695 ± 0.002
Sim Object	$\mathbf{0.575 \pm 0.002}$	0.419 ± 0.004	0.452 ± 0.003
Light-Dark	$\mathbf{88.7 \pm 0.131}$	-225 ± 0	-214.6 ± 0.516

running the particle filter ten times faster than the policy for large moves (action = ± 10) closes the remaining gap to *SARSOP*'s performance, whereas at equal frequencies *PuBS* is lower yet still exceeds *QMDP*, as shown in Figure 4. While similar performance improvements could be expected for articulated object tasks, we found that even with synchronous execution, *PuBS* performs adequately for those scenarios. These results demonstrate that *PuBS* combines computational feasibility with competitive performance, while remaining applicable to continuous domains where classical solvers fail.

D. Experiment: The Critical Role of Uncertainty

We ablate uncertainty inputs by comparing *PuBS*, which provides the policy with belief mean μ_t and entropy H_t, against *Random* H, which replaces entropy with a random value uniformly sampled from $[0, 1]$, and *No* H, which omits entropy. We report average discounted rewards with standard errors, undiscounted for *Light Dark*, and show a representative set of policy behavior heatmaps for *PuBS* to analyze how uncertainty influences the learned behaviors qualitatively. As summarized in Table III, *PuBS* outperforms both *Random* H and *No* H, particularly in the *Light Dark* task where the ablations yield negative rewards. The policy behavior heatmaps, illustrated in Figure 5, show a consistent principle learned across all tasks is that the policy only

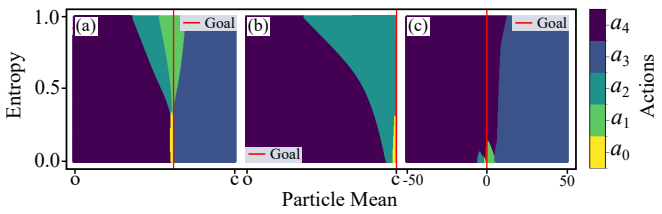

Fig. 5. Policy behavior heatmaps for (a) *Door 2-Middle*, (b) *Drawer 2-Close*, and (c) *Light Dark*. For articulated tasks, action definitions are in Section III-A.2. For Light Dark, actions are $\{0, -1, 1, -10, 10\}$ in order.

TABLE IV

SECTION IV-E REPORTS SUCCESS RATES, GOAL ERROR (±SE), AND TIME EFFICIENCY (±SE) FOR 20 TRIALS PER TASK.

	Success	Goal Error	Time Eff.
Drawer 1	95%	1.77 ± 0.36	0.884 ± 0.066
Drawer 2	95%	1.52 ± 0.28	0.982 ± 0.098
Door 1	90%	3.60 ± 2.02	0.166 ± 0.012
Door 2	80%	7.31 ± 1.73	1.867 ± 1.142

selects the stop action when the entropy falls below a task-specific, learned threshold to improve goal accuracy. For instance, in the *Light Dark* task (Figure 5(c)), the policy first oscillates about the "light" region ($s = 10$) to reduce uncertainty, then switches to small, precise actions once entropy is sufficiently low to reach the goal. Additionally, the policy uses entropy to balance the trade-off between speed and safety. It begins with slow, safe actions under high uncertainty, switching to faster, high-force actions as entropy drops, and reverting to safe actions near joint limits. These emergent behaviors demonstrate that uncertainty is a key signal for both safety and efficiency in policy execution.

E. Experiment: Real-World Policy Transfer

We evaluate sim to real transfer on four objects at the *Middle* goal with 20 trials per object from randomized starts and report success rate where a trial fails if it ends with a safety violation, goal error as the final distance to the goal for successful trials (centimeters for drawers, degrees for doors), and time per unit as total task time divided by start-to-goal distance. As shown in Table IV, policies for *Drawer 1*, *Drawer 2*, and *Door 1* transferred successfully with high success rates and small goal errors relative to their total range of motion (25 – 28 cm for drawers and 125° for doors). The few failures observed were attributed to minor calibration drift during the experiment. Importantly, once the observation model was recalibrated, these trials succeeded without requiring retraining of the policy (the new trials have not been included in the reported results). The weaker results of *Door 2* were due to the structural instability of the setup; the strong magnetic latch on the door would cause the entire wooden cabinet to shift when opening or closing the door at the joint limit. This behavior, which a human operator would intuitively counteract by stabilizing the cabinet with one hand and opening while pulling with the other, introduced significant discrepancies between the

learned observation model, predefined controller parameters, and the real-world dynamics.

V. CONCLUSION

We formulate goal-directed manipulation of articulated objects as a POMDP and introduce PuBS, which summarizes the estimated belief to define an approximate MDP solvable with reinforcement learning. Using force and proprioceptive feedback, PuBS estimates the latent state and executes policies that reach goals safely and efficiently. Simulation and real-world experiments show strong performance across diverse articulated objects, including those with complex force profiles, such as locking mechanisms. A key strength of our framework is its uncertainty-aware behavior, enabling the policy to balance safety and time efficiency during execution. A limitation is sensitivity to shifts from calibration, as seen with unstable, moving fixtures. Future work will enable online updates to the observation model to improve robustness and adaptability through repeated interaction. Additionally, PuBS will be extended to multidimensional state spaces to infer kinematic model parameters, enabling manipulation of previously unseen articulated objects.

REFERENCES

[1] O. Kroemer, S. Niekum, and G. Konidaris, "A review of robot learning for manipulation: challenges, representations, and algorithms," *Int. J. Comput. Vis.*, vol. 22, no. 1, Jan. 2021.

[2] B. Abbatematteo, S. Tellex, and G. Konidaris, "Learning to generalize kinematic models to novel objects," in *Proc. Conf. Robot Learn. (CoRL)*, ser. Proc. Mach. Learn. Res. (PMLR), vol. 100. PMLR, Oct. 30 – Nov. 1, 2020, pp. 1289–1299.

[3] V. Zeng, T. E. Lee, J. Liang, and O. Kroemer, "Visual identification of articulated object parts," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2021, pp. 2443–2450.

[4] J. Lv, Q. Yu, L. Shao, W. Liu, W. Xu, and C. Lu, "Sagci-system: Towards sample-efficient, generalizable, compositional, and incremental robot learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2022, pp. 98–105.

[5] H. Xiong, R. Mendonca, K. Shaw, and D. Pathak, "Adaptive mobile manipulation for articulated objects in the open world," *arXiv*, vol. abs/2401.14403, 2024.

[6] Y. Wang, Z. Wang, M. Nakura, P. Bhowal, C.-L. Kuo, Y.-T. Chen, Z. Erickson, and D. Held, "Articubot: Learning universal articulated object manipulation policy via large scale simulation," in *Proc. Robotics: Science and Systems (RSS)*, June 2025.

[7] Z. Zhao, Y. Li, W. Li, Z. Qi, L. Ruan, Y. Zhu, and K. Althoefer, "Tac-man: Tactile-informed prior-free manipulation of articulated objects," *IEEE Trans. Robot.*, vol. 41, pp. 538–557, 2025.

[8] P. Xie, R. Chen, S. Chen, Y. Qin, F. Xiang, T. Sun, J. Xu, G. Wang, and H. Su, "Part-guided 3d rl for sim2real articulated object manipulation," *IEEE Robot. Autom. Lett.*, vol. 8, no. 11, pp. 7178–7185, 2023.

[9] J. Sturm, C. Stachniss, and W. Burgard, "A probabilistic framework for learning kinematic models of articulated objects," *J. Artif. Intell. Res.*, vol. 41, no. 2, p. 477–526, May 2011.

[10] S. Niekum, S. Osentoski, C. G. Atkeson, and A. G. Barto, "Online bayesian changepoint detection for articulated motion models," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2015, pp. 1468–1475.

[11] R. Martín-Martín and O. Brock, "Coupled recursive estimation for online interactive perception of articulated objects," *Int. J. Robot. Res.*, vol. 41, no. 8, pp. 741–777, 2022.

[12] Y. Karayiannidis, C. Smith, F. E. Viña, P. Ogren, and D. Kragic, ""open sesame!" adaptive force/velocity control for opening unknown doors," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2012, pp. 4040–4047.

[13] Y. Karayiannidis, C. Smith, F. E. V. Barrientos, P. Ögren, and D. Kragic, "An adaptive control approach for opening doors and drawers under uncertainties," *IEEE Trans. Robot.*, vol. 32, no. 1, pp. 161–175, 2016.

[14] Z. Xu, Z. He, and S. Song, "Universal manipulation policy network for articulated objects," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 2447–2454, 2022.

[15] X. Li and O. Brock, "Learning from demonstration based on environmental constraints," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 10 938–10 945, 2022.

[16] T. Lips and F. Wyffels, "Revisiting proprioceptive sensing for articulated object manipulation," *arXiv*, vol. abs/2305.09584, 2023.

[17] R. Buchanan, A. Röfer, J. Moura, A. Valada, and S. Vijayakumar, "Online estimation of articulated objects with factor graphs using vision and proprioceptive sensing," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2024, pp. 16 111–16 117.

[18] F. Xiang, *et al.*, "Sapien: A simulated part-based interactive environment," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 11 094–11 104.

[19] J. Wong, A. Tung, A. Kurenkov, A. Mandlekar, L. Fei-Fei, S. Savarese, and R. Martín-Martín, "Error-aware imitation learning from teleoperation data for mobile manipulation," in *Proc. Conf. Robot Learn. (CoRL)*, ser. Proc. Mach. Learn. Res. (PMLR), vol. 164. PMLR, Nov. 8–11, 2022, pp. 1367–1378.

[20] R. Gong, *et al.*, "Arnold: A benchmark for language-grounded task learning with continuous states in realistic 3d scenes," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2023, pp. 20 426–20 438.

[21] Y. Urakami, A. Hodgkinson, C. Carlin, R. Leu, L. Rigazio, and P. Abbeel, "Doorgym: A scalable door opening environment and baseline agent," *arXiv*, vol. abs/1908.01887, 2019.

[22] M. Lauri, D. Hsu, and J. Pajarinen, "Partially observable markov decision processes in robotics: A survey," *IEEE Trans. Robot.*, vol. 39, no. 1, pp. 21–40, 2023.

[23] A. Cassandra, M. L. Littman, and N. L. Zhang, "Incremental pruning: a simple, fast, exact method for partially observable markov decision processes," in *Proc. Conf. Uncertainty in Artif. Intell. (UAI)*, ser. UAI'97. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997, p. 54–61.

[24] M. L. Littman, A. R. Cassandra, and L. P. Kaelbling, "Learning policies for partially observable environments: scaling up," in *Proc. Int. Conf. Mach. Learn. (ICML)*, ser. ICML'95. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1995, p. 362–370.

[25] W. S. L. Hanna Kurniawati, David Hsu, "SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces," in *Proc. Robotics: Science and Systems (RSS)*, Zurich, Switzerland, June 2008.

[26] D. Silver and J. Veness, "Monte-carlo planning in large pomdps," in *Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 23. Curran Associates, Inc., 2010.

[27] N. Ye, A. Somani, D. Hsu, and W. S. Lee, "Despot: online pomdp planning with regularization," *J. Artif. Intell. Res.*, vol. 58, no. 1, p. 231–266, Jan. 2017.

[28] H. Kurniawati and V. Yadav, *An Online POMDP Solver for Uncertainty Planning in Dynamic Environment*. Cham: Springer International Publishing, 2016, pp. 611–629.

[29] T. Ni, B. Eysenbach, and R. Salakhutdinov, "Recurrent model-free rl can be a strong baseline for many pomdps," in *Proc. Int. Conf. Mach. Learn. (ICML)*. PMLR, 2022, pp. 16 691–16 723.

[30] M. H. Lim, T. J. Becker, M. J. Kochenderfer, C. J. Tomlin, and Z. N. Sunberg, "Optimality guarantees for particle belief approximation of pomdps," *J. Artif. Intell. Res.*, vol. 77, pp. 1591–1636, 2023.

[31] R. S. Sutton and A. G. Barto, *Reinforcement learning - an introduction, 2nd Edition*. MIT Press, 2018.

[32] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola, "Deep sets," in *Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 30. Curran Associates, Inc., 2017.

[33] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv*, vol. abs/1707.06347, 2017.

[34] P. Fearnhead, "Exact and efficient bayesian inference for multiple changepoint problems," *Stat. Comput.*, vol. 16, no. 2, p. 203–213, June 2006.

[35] A. Savitzky and M. J. Golay, "Smoothing and differentiation of data by simplified least squares procedures." *Anal. Chem.*, vol. 36, no. 8, pp. 1627–1639, 1964.

[36] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *J. Mach. Learn. Res.*, vol. 22, no. 268, pp. 1–8, 2021.