Estimating Value of Assistance for Online POMDP Robotic Agents

Yuval Goshen and Sarah Keren
The Collaborative AI and Robotics (CLAIR) Lab
The Taub Faculty of Computer Science
Technion - Israel Institute of Technology

Abstract: Robotic agents operating in dynamic, partially observable environments often benefit from teammate assistance. For robots using online POMDP planning, evaluating assistance actions requires computationally intensive policy evaluations, limiting real-time decision-making capabilities. We formulate *Value of Assistance* (VOA) for POMDP agents and develop efficient heuristics that approximate VOA without requiring complete evaluations. Our empirical results on both a standard POMDP benchmark and a multi-robot manipulation task demonstrate that our suggested heuristics enable real-time computation while maintaining sufficient accuracy for effective helping action selection.

Keywords: Partially Observable Markov Decision Processes (POMDPs), Online Planning, Multi-Agent Systems, Multi-Robot Systems

1 Introduction

Robotic agents are required to operate in increasingly dynamic and unpredictable environments, in which they lack complete information about the world due to sensor noise, limited field of view, or incomplete models of the environment. In this work, we support multi-robot settings in which agents can assist each other in different ways. For example, a team member can move obstacles or rearrange objects to facilitate manipulation or to improve observability. These assistance options create opportunities for meaningful collaboration in multi-robot systems but introduce several decision-making challenges: selecting the most effective helping action from alternatives, evaluating whether pausing a current task to assist others is worthwhile, and determining which agent would benefit most from assistance when multiple team members require it.

To address these challenges, our work focuses on formulating ways to estimate *Value of Assistance* (VOA) for partially informed robotic agents and on providing principled approaches for quantifying and comparing the potential benefits of different assistive actions.

Example: *Consider the two-agent setting depicted in Figure 1* [1]*. Agent1 is equipped with a parallel gripper and camera, and is tasked with placing cups on a table and pouring soda into them. This requires planning high-level actions while moving to positions that enable manipulation and sensing. In this setting, Agent2 can assist by moving obstacles (which is impossible for Agent1) using its vacuum gripper. The figure shows two assistance options: moving Obstacle #1 reveals the occluded can, improving visibility, while moving Obstacle #2 clears a path to the blue cup. We aim to quantify Value of Assistance (VOA) as the expected long-term benefit of each assistive action, enabling comparison against the cost of pausing Agent2's own task.*

Beyond this illustrative example, assessing VOA is relevant to a broader class of applications such as automated manufacturing, environmental monitoring, and construction in which agents can assist each other but need to consider their own resources and objectives. In such settings, deciding how to assist other agents requires considering the long-term effects of actions. For example, will clearing a path benefit only the current navigation segment or enable access to multiple future goals?

[1] A video demonstrating this scenario: https://youtu.be/wf1qMjeNgnU

9th Conference on Robot Learning (CoRL 2025), Seoul, Korea.

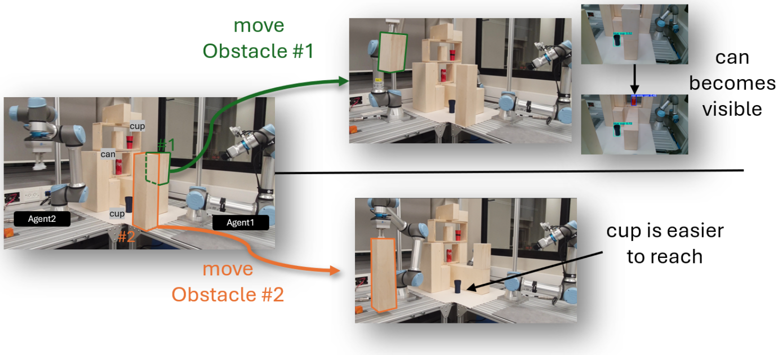

Figure 1: A collaborative multi-robot setting.

As is typical in such settings, we model the agent's task using a Partially Observable Markov Decision Process (POMDPs) [1]. POMDPs provide a mathematical framework for decision-making under uncertainty, balancing exploration for information gathering with exploitation of current knowledge. They capture the complexities of noisy sensors, imperfect actuation, and partial observability inherent in real-world robotic tasks. However, exact POMDP solutions are computationally intractable for large problems [2], leading to the development of online planning methods that have enabled practical applications in robotics [3, 4]. As a result of the complexity of a POMDP and its online, approximate nature, exact computation of VOA becomes intractable for real-time decision-making, especially when there are many possible interventions to consider and there is a need to reevaluate the policy to determine the effect interventions will have on the agent's decisions.

With the objective of supporting effective real-time assistance decisions, we formulate **Value of Assistance (VOA)** for POMDP agents and address computational challenges by developing domain-agnostic heuristics. These enable rapid evaluation of potential helping actions while accounting for their long-term effects. Our empirical evaluations on both a standard POMDP benchmark and a two-agent robotic manipulation scenario compare several heuristic approaches, with our Full-Information heuristic demonstrating near-optimal action selection at significantly faster computation speeds compared to direct VOA estimation, effectively predicting intervention impacts and supporting optimal assistance decisions.[2]

2 Related Work

POMDPs have proven effective for single-robot tasks like navigation [5], tracking [6], and crowd interaction [7], with recent extensions to continuous domains [8]. Comprehensive surveys [4, 9] highlight their growing applications in robotics [10, 11]. While Dec-POMDP frameworks provide theoretical foundations [12] and algorithmic approaches [13] for decentralized decision-making, they aim to solve for complete joint policies, which is more complex than our setting. Earlier work on quantifying assistive value in human-robot interaction [14] introduced **helpfulness** as a metric based on effort reduction, but primarily in deterministic settings with full task knowledge. We focus instead on the more targeted problem of evaluating specific assistance opportunities in real-time in partially observable stochastic environments.

The concept of Value of Assistance (VOA) was previously explored for robot navigation [15], assessing how localization affects a robot's expected cost. Our work differs by addressing robotic agents using online POMDP planning rather than fixed control policies, and by focusing on complex manipulation tasks where assistance affects both immediate performance and future decisions.

3 Preliminaries

Partially Observable Markov Decision Process (POMDP) [1] is defined as a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, \mathcal{O}, \gamma \rangle$, where $\mathcal{S}$ is a set of states, $\mathcal{A}$ is a set of actions, $\mathcal{T}(s'|s,a)$ is the transi-

[2] The code and supplementary materials are available at: https://github.com/CLAIR-LAB-TECHNION/voa-online-pomdp

tion probability function, $\mathcal{R}(s, a)$ is the reward function, Ω is the observation set, $\mathcal{O}(o|s', a)$ is the observation probability function, and γ is a discount factor. POMDP agents cannot directly observe the true state. Instead, they make decisions based on their belief $\beta \in \mathcal{B}$, which is a probability distribution $\beta : \mathcal{S} \to [0, 1](\sum_{s \in \mathcal{S}} \beta(s) = 1)$ over states given past observations and actions. An agent updates its belief after taking action a and receiving observation o using a belief update function $\tau : \mathcal{B} \times A \times \Omega \to \mathcal{B}$. The agent's policy $\pi : \mathcal{B} \to \Delta(A)$ maps beliefs to distributions over actions. For a given policy π, the value function $V^{\pi}(\beta) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s^t, \pi(\beta^t)) \mid \beta^0 = \beta\right]$ represents the expected sum of discounted rewards. An optimal policy π^* maximizes this value function, which satisfies the Bellman equation $V^*(\beta) = \max_{a \in A}\left[R(\beta, a) + \gamma \sum_{o \in \Omega} P(o|\beta, a) V^*(\beta_{a,o})\right]$ where $R(\beta, a) = \sum_{s \in S} R(s, a)\beta(s)$ is the expected immediate reward.

Online Planning in POMDPs. Finding optimal policies for POMDPs is PSPACE-complete [2], making exact solution methods impractical for most real-world applications. This computational challenge is particularly acute in robotics, where state and observation spaces are often large or continuous. Even approximate methods that pre-compute policies offline struggle with the curse of dimensionality and the curse of history [16]. Online planning algorithms address these challenges by computing actions only for the current belief at runtime rather than solving the POMDP for all possible beliefs. This approach allows handling of much larger state spaces than offline methods, making it particularly suitable for robotic applications [3]. The trade-off is that online planning typically produces sub-optimal policies due to computational constraints at runtime. Nevertheless, these methods have proven effective in practice [17, 18, 4, 3].

Partially Observable Monte Carlo Planning (POMCP) [17] is a widely-used online planning algorithm that combines Monte Carlo Tree Search (MCTS) with particle-based belief representation [19]. Nodes represent uncertainty about the state through sets of particles, with each particle corresponding to a possible state consistent with the action-observation history. As illustrated in Figure 2, the algorithm constructs a search tree where nodes alternate between representing actions (hexagons) and observations (ellipses). At each iteration, POMCP traverses the tree using UCB1-based action selection [20] until reaching a leaf node, and then uses a rollout policy π_{rollout} to complete the iteration. Values are backpropagated through the visited nodes to guide future action selection. The policy induced by POMCP is inherently stochastic due to its sampling-based nature and limited planning time. POMCP has demonstrated strong performance across various domains [17], particularly in robotics, where its ability to handle large state spaces and provide real-time decision-making is valuable.

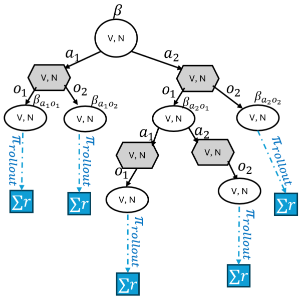

Figure 2: Illustration of a POMCP search tree.

4 Value of Assistance in POMDPs

We model a robotic task as a POMDP $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, \mathcal{O} \rangle$ with stochastic dynamics and noisy sensors. The agent employs online planning and replans at each time step, resulting in a stochastic policy $\pi : \beta \to \Delta(\mathcal{A})$. This policy may be sub-optimal. Upon executing an action a and receiving observation o, the agent updates its belief through a belief update function: $\beta' = \tau(\beta, a, o)$.

We consider assistance through helping actions $\alpha \in \mathrm{A}_{\mathcal{H}}$ which can modify the environment state according to a transition function $\mathcal{T}_{\mathcal{H}}(s'|s, \alpha)$ and provide observation $\omega \in \Omega_{\mathcal{H}}$ according to a stochastic observation function $\mathcal{O}_{\mathcal{H}}(\omega|s', \alpha)$. Since assistance may be provided by a different agent, the set $\Omega_{\mathcal{H}}$ of possible observations may differ from the agent's observation set Ω. After a helping action $\alpha \in \mathrm{A}_{\mathcal{H}}$ is performed, the agent updates its belief with its update function $\tau(\beta, \alpha, \omega)$ that considers both state changes and the new observations and then continues executing its policy in the modified environment. For simplicity, we assume uniform cost across helping actions, though this framework can be trivially extended to incorporate different action costs. Additionally, we assume here that agents share the same belief, which can be achieved either through direct communication or through sharing the action-observation history. This assumption will be relaxed in future work.

To quantify the impact of assistance, we define VOA for a helping action given a belief as the difference between the expected reward the agent will accumulate with and without assistance. This value arises from both potential changes in the environment state and updates to the agent's belief based on received observations. We first express VOA in terms of belief-conditioned probabilities:

$$\mathcal{U}_{VoA}(\alpha, \beta) = \mathbb{E}_{\substack{s' \sim P(s'|\beta,\alpha) \\ \omega \sim \mathcal{O}_{\mathcal{H}}(\omega|s',\alpha)}} \left[V^{\pi}(\beta')\right] - V^{\pi}(\beta) \tag{1}$$

where V^{π} is the value function of the induced actor's policy π and $\beta' = \tau(\beta, \alpha, \omega)$ is the updated belief. By conditioning on the state, we can express this in terms of the transition and observation models of the helping agent. The complete derivation of Equation 2 is provided in Appendix 1.1.

$$\mathcal{U}_{VoA}(\alpha, \beta) = \mathbb{E}_{s\sim\beta}[\mathbb{E}_{\substack{s' \sim \mathcal{T}_{\mathcal{H}}(s'|s,\alpha) \\ \omega \sim \mathcal{O}_{\mathcal{H}}(\omega|s',\alpha)}} [V^{\pi}(\beta')]] - V^{\pi}(\beta) \tag{2}$$

For purely observational helping actions that do not change the state of the environment, VOA reduces to:

$$\mathcal{U}_{VoA}(\alpha, \beta) = \mathbb{E}_{s\sim\beta}[\mathbb{E}_{\omega\sim\mathcal{O}_{\mathcal{H}}(\omega|s,\alpha)}[V^{\pi}(\beta')]] - V^{\pi}(\beta) \tag{3}$$

This special case corresponds to value of information (VOI) in POMDPs [21, 22, 23, 24, 25, 9, 26]. VOA extends this well-studied concept to both information gathering and state-changing assistance.

5 Assessing VOA

Computing VOA for POMDP agents presents a unique policy evaluation challenge as we demonstrate in the analysis of Algorithm 1. The assisted agent employs online planning, computing actions at runtime through methods like POMCP [17] (see Figure 2) . Evaluating such a policy is computationally intensive, as each step in a simulated trajectory requires selecting the agent's action using long-term reasoning. This makes exact VOA computation intractable for real-time decision-making, especially when evaluating multiple potential helping actions.

In principle, it may be possible to formulate and solve our assistance scenario by extending the agent's POMDP to include the helping actions and their effects. However, since we assume assistance can only change the state and belief of the agent, it is redundant to recompute the entire policy, and we can instead focus on assessing the effect of the intervention.

Our suggested approach is a Monte Carlo algorithm for estimating VOA. Notably, Monte Carlo sampling appears at multiple levels in our approach. At the policy computation level, the agent uses Monte Carlo tree iterations internally for action selection, with each iteration consisting of tree traversal and value estimation through rollouts at leaf nodes. At a higher level, we use Monte Carlo sampling to estimate VOA through complete policy evaluation episodes. To avoid confusion, we will refer to the agent's internal processes as *tree iterations* and to the VOA estimation trajectories as *policy evaluation episodes* throughout the paper.

Our approach, described in Algorithm 1, applies a Monte Carlo estimation approach to assess the difference in expected returns between two scenarios: with and without assistance. Note that we present and analyze the finite-horizon case, though the approach applies equally to discounted infinite-horizon problems (Appendix 1.3).

In lines 1-4, we initialize the necessary data structures and parameters. For each state sampled from the belief (line 6), the algorithm first simulates the effect of the helping action (lines 7,8,9). It then performs two L-step rollouts using the agent's policy: one from the post-help state and belief (line 9), and another from the original state and belief (line 10). Figure 3 [left] illustrates how each policy evaluation episode is executed by repeatedly constructing planning trees, sampling transitions, and updating beliefs. The final estimate is computed through importance sampling (line 16), where samples are weighted according to their probability under the current belief.

The algorithm's computational cost depends on several key parameters: k (number of sampled states), L (episode length), and the planning algorithm specific parameters. We analyze for POMCP with: N_s (number of tree simulations), N_D (maximum depth), and N_P (particles representing the belief). As shown in Appendix 1.2, the total complexity is $O(kL(N_sN_D + N_P))$. This high cost

Algorithm 1: VOA Prediction Through Policy Evaluation Episodes (Finite Horizon)

```
1  Input: POMDP Model $\mathcal{M}$, Actor's Policy $\pi$, Belief $\beta$, Helping Action
       $\alpha$, Resource Limit $\text{res}_{\text{max}}$, Max Iterations $k$, Time Horizon $L$
   Output: Predicted Value of Assistance $\widehat{\mathcal{U}_{VoA}}(\beta, \alpha)$
2  Initialize empty arrays $V_\triangle$ and $S$;
3  $\text{res}_{\text{used}} \leftarrow 0$;
4  $i \leftarrow 0$;
5  while ($i < k$ and $res_{used} < res_{max}$) do
6      Sample initial state $s_i \sim \beta$;
7      Sample $s_i^{\mathcal{H}} \sim \mathcal{T}_{\mathcal{H}}(\cdot|s_i, \alpha)$ ;                  // State after help
8      Sample $o^{\mathcal{H}} \sim \mathcal{O}_{\mathcal{H}}(\cdot|s_i^{\mathcal{H}}, \alpha)$ ;   // Observation after help
9      $\beta^{\mathcal{H}} \leftarrow \tau(\beta, \alpha, o^{\mathcal{H}})$ ;                        // Belief after help
10     $G_{help} \leftarrow$ EpSim($\mathcal{M}, s_i^{\mathcal{H}}, \beta^{\mathcal{H}}, \pi, L$);  // Return with help
11     $G_{no\_help} \leftarrow$ EpSim($\mathcal{M}, s_i, \beta, \pi, L$) ;     // Return no help
12     $V_\triangle$.append($G_{help} - G_{no\_help}$);
13     $S$.append($s_i$);
14     $i \leftarrow i + 1$;
15     Update($\text{res}_{\text{used}}$);
16 $\widehat{\mathcal{U}_{VoA}}(\beta, \alpha) \leftarrow \frac{\sum_{j=1}^{i} \beta(S[j]) V_\triangle[j]}{\sum_{j=1}^{i} \beta(S[j])}$ ;
17 return $\widehat{\mathcal{U}_{VoA}}(\beta, \alpha)$ ;

   // Note: This function simulates a
      complete policy evaluation episode.
18 Function EpSim($\mathcal{M}, s, \beta, \pi, L$):
19     $G \leftarrow 0$;
20     $s_{curr} \leftarrow s$;
21     $\beta_{curr} \leftarrow \beta$;
22     for $t \leftarrow 1$ to $L$ do
23         $a \leftarrow \pi(\beta_{curr})$ ;        // Constructs a
             search tree
24         Sample $s_{next} \sim \mathcal{T}(\cdot|s_{curr}, a)$;
25         Sample $o \sim \mathcal{O}(\cdot|s_{next}, a)$;
26         $G \leftarrow G + \mathcal{R}(s_{curr}, a)$;
27         $\beta_{curr} \leftarrow \tau(\beta_{curr}, a, o)$;
28         $s_{curr} \leftarrow s_{next}$;
29     return $G$
```

stems from the need to construct and search new trees for each step of the simulated trajectories. While different MCTS algorithms may vary in their handling of observations and belief updates, they face similar computational challenges.

6 Heuristics for VOA

The computational complexity analysis of Algorithm 1 reveals significant challenges in estimating VOA for POMDP agents using online planning approaches. While the algorithm provides a principled approach, its computational requirements make it impractical for real-time assistance decisions when evaluating multiple helping actions.

This limitation motivates the development of computationally efficient heuristics for approximating VOA. We define a VOA heuristic $h(\alpha, \beta) \propto \mathcal{U}_{VoA}(\alpha, \beta)$ as a function that maps a belief and helping action to a real value. A key requirement is not perfect numerical accuracy, but rather the ability to efficiently identify promising helping actions.

We aim to propose domain-agnostic heuristics that leverage fundamental properties of POMDPs and their value functions. These heuristics are particularly valuable in scenarios where a helper agent must select from multiple possible actions under time and computational constraints, as selecting an action that ranks slightly below optimal may be preferable to the computational delay of exact evaluation. In the following paragraphs, we describe three heuristics with full details provided in Appendix 1.4. Each targets a different aspect of the computational complexity in Algorithm 1: reducing the required planning steps, eliminating expensive tree construction, or reformulating the problem as deterministic planning.

First-Action Value Heuristic h_{FA}. Our first heuristic addresses the computational burden of simulating complete L-step trajectories (lines 23-27 in Algorithm 1). Instead of these full episodes, h_{FA} approximates VOA by planning only for the first step, using the agent's regular MCTS search approach and using the resulting tree to estimate the value (Figure 3 [mid-left]). We replace the policy value function V^π with the root node value estimate:

$$h_{FA}(\alpha, \beta) = \mathbb{E}_{s \sim \beta}[\mathbb{E}_{\substack{s' \sim \mathcal{T}_{\mathcal{H}}(\cdot|s,\alpha) \\ \omega \sim \mathcal{O}_{\mathcal{H}}(\cdot|s',\alpha)}}[V_{\text{root}}(\tau_{\mathcal{H}}(\beta, \alpha, \omega))]] - V_{\text{root}}(\beta) \tag{4}$$

This modification requires only a single planning step per sampled state, reducing computational complexity from $O(kL(N_s N_D + N_P))$ to $O(k(N_s N_D + N_P))$.

Rollout-Policy Heuristic $h_{\pi_{\textbf{Rollout}}}$. The second heuristic addresses the cost of repeatedly constructing search trees during episode simulation (line 23 in Algorithm 1). Rather than using the full planning policy, $h_{\pi_{\text{Rollout}}}$ directly uses the planner's rollout policy π_{Rollout} for action selection (Figure 3

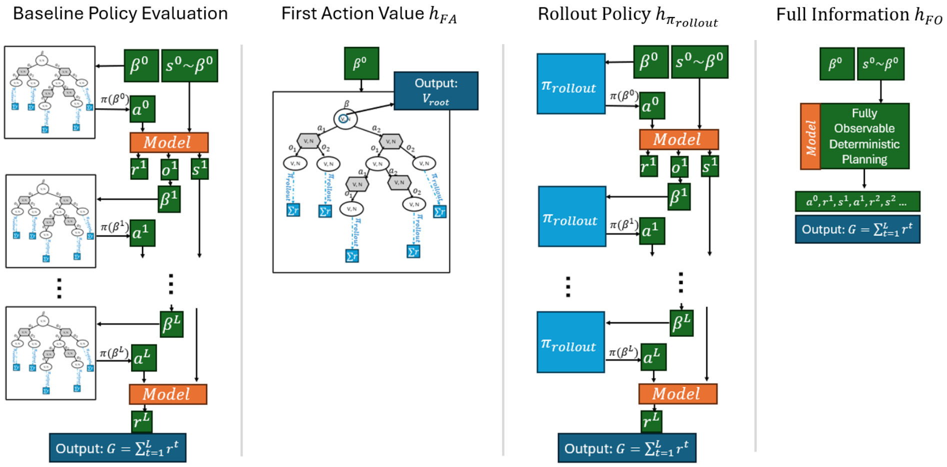
Baseline Policy Evaluation
First Action Value h_{FA}
Rollout Policy $h_{\pi_{rollout}}$
Full Information h_{FO}
β^0
$s^0 \sim \beta^0$
$\pi(\beta^0)$
a^0
Model
r^1
o^1
s^1
β^1
$\pi(\beta^1)$
a^1
β^L
$\pi(\beta^L)$
a^L
r^L
Output: $G = \sum_{t=1}^{L} r^t$
Output: V_{root}
β
V, N
a_1
a_2
o_1
o_2
$\pi_{rollout}$
Σr
Fully Observable Deterministic Planning
$a^0, r^1, s^1, a^1, r^2, s^2$ ...

Figure 3: VOA estimation approaches: Baseline Policy Evaluation [leftmost], First-Action Value heuristic, Rollout-Policy heuristic, and Full-Information heuristic [rightmost].

[mid-right]). We replace the planning policy value function with the rollout policy value:

$$h_{\pi_{\text{Rollout}}}(\alpha, \beta) = \mathbb{E}_{s \sim \beta}[\mathbb{E}_{\substack{s' \sim \mathcal{T}_{\mathcal{H}}(\cdot|s,\alpha) \\ \omega \sim \mathcal{O}_{\mathcal{H}}(\cdot|s',\alpha)}} [V^{\pi_{\text{Rollout}}}(\beta')]] - V^{\pi_{\text{Rollout}}}(\beta) \tag{5}$$

This eliminates the expensive tree construction operations, reducing computational complexity from $O(kL(N_s N_D + N_P))$ to $O(kL(N_P))$.

Full-Information Heuristic h_{FO}**.** Our third heuristic addresses the VOA estimation challenge by using a fully observable deterministic relaxation of the problem (Figure 3 [right]). Instead of performing policy evaluation episodes in lines 10 and 11 of Algorithm 1, we create a fully observable deterministic variant through *all-outcome determinization*, where the agent can choose any outcome from each action's stochastic transition support. For each state sampled from the belief, we perform deterministic planning as if that sampled state were the true state, computing returns both with and without assistance. Importantly, this does not give the agent actual "full information" during execution, it simply allows for more efficient planning during VOA estimation by temporarily setting aside uncertainty for each sampled state. This approach converts the complex POMDP planning problem into a much simpler deterministic planning problem inspired by [27, 28], benefiting from decades of research in deterministic planning algorithms [29].

To compute VOA, we replace the value function V^π with the deterministic optimal plan value U:

$$h_{FO}(\alpha, \beta) = \mathbb{E}_{s \sim \beta}[\mathbb{E}_{s' \sim \mathcal{T}_{\mathcal{H}}(\cdot|s,\alpha)}[U(s')] - U(s)] \tag{6}$$

The computational complexity reduces to $O(k|S||A|B)$ for basic state-space search, where B is the maximum branching factor in the transition support. In Appendix 1.5, we show that $U(s)$ provides an upper bound on the optimal POMDP value function, though VOA computed this way isn't necessarily an upper bound on the actual VOA.

7 Empirical Evaluation

Our empirical evaluation aims to assess the effectiveness of our proposed heuristics in estimating the Value of Assistance (VOA). The key question we address is whether these computationally efficient heuristics can reliably identify beneficial helping actions despite the relaxations that are made. While we demonstrate our approach using POMCP, our heuristics readily extend to other online MCTS algorithms, including POMCPOW [30] and DESPOT [31] variants, as they share the core MCTS structure with root value estimates and rollout policies. We examine this in two distinct environments.

RockSample. We evaluate on RockSample(11,11) [32], a POMDP benchmark where an agent navigates a grid with rocks to locate the valuable ones using a noisy long-range sensor. This large version of the environment requires online planning approaches like POMCP [17], as its state space is too large for exact solutions. We augmented this benchmark with helping actions that allow clustering rocks in different areas of the map, generating approximately 50 distinct assistance options per environment instance. This creates a challenging assistance evaluation problem with many alternatives to rank. Detailed descriptions of the experimental setup are provided in Appendix 1.10.

Robotic Manipulation (POMAN). Our second domain involves the multi-robot manipulation scenario introduced in Figure 1. A robot equipped with a parallel gripper must manipulate objects while navigating among large obstacles. The robot's perception system integrates YOLO-world object detection [33] with stereo depth data to estimate partially observable object positions (Figure 4). A second robot equipped with a vacuum gripper can assist by repositioning the large obstacles, which can significantly impact task performance by modifying accessibility and visibility. The environment specifications and perception system implementation are provided in Appendix 1.11.

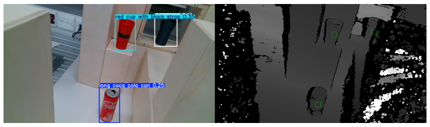

Figure 4: [left] Demonstrating YOLO-world's[33] object detection in the robot's camera view, and [right] the corresponding depth information from the stereo camera used to estimate object positions.

Our experiments used POMCP for online planning, with detailed implementation provided in Appendix 1.8. In both environments, each POMCP planning step requires 1-3 seconds, which is reasonable during task execution but making VOA computation through direct policy evaluation prohibitively expensive. To establish ground truth, we computed empirical VOA through extensive offline simulation (300 episodes per belief-action pair, each requiring several hours). Since these values remain estimates, we computed 95% confidence intervals using bootstrap resampling [34]. We implemented the three heuristics from Section 6, evaluating each across varying sample sizes. For RockSample, we tested an enhanced First-Action Value variant that increased computational resources for the single planning step (10,000 simulations at depth 100, compared to the standard 2,000 simulations at depth 20).

7.1 Evaluation Metrics

We evaluate our heuristics using complementary metrics that capture different performance aspects. Detailed definitions and additional metrics are provided in Appendix 1.9:

Partial Order Agreement. Measures how well a heuristic's ordering of helping actions agrees with the empirical VOA's partial ordering, considering confidence intervals. Scores range from 0 to 1, with higher values indicating better agreement.

Normalized Regret. Quantifies the value lost by selecting the heuristic's recommended action instead of the truly optimal one. This directly measures practical impact on decision quality. Values range from 0 to 1, with lower values indicating better performance.

Top-k Accuracy. Measures what fraction of the truly best k actions are identified by the heuristic. We report top-5 for RockSample (with approximately 50 possible actions) and top-2 for POMAN (4 actions). Higher values (0-1 scale) indicate better performance.

Computation Time. Average seconds required to compute heuristic values per belief-action pair, directly measuring real-time feasibility.

Figure 5 presents our evaluation results across all metrics for both domains. Each graph compares heuristic performance, organized left to right: baseline VOA estimation ($\widehat{\mathcal{U}_{VoA}}$) as described in Algorithm 1 in red, First-Action Value heuristic (h_{FA}) in yellow, enhanced First-Action Value (h^+_{FA}) in green, Rollout-Policy ($h_{\pi_{\text{Rollout}}}$) in blue, and Full-Information Value (h_{FO}) in purple. Within each approach, grouped bars represent different numbers of sampled states, demonstrating performance stability with varying computational resources. Note that h^+_{FA} appears only in RockSample results, and $h_{\pi_{\text{Rollout}}}$ is omitted from POMAN plots due to consistently zero VOA estimates in that domain. In Appendix 1.12 we perform a more elaborate analysis of the results.

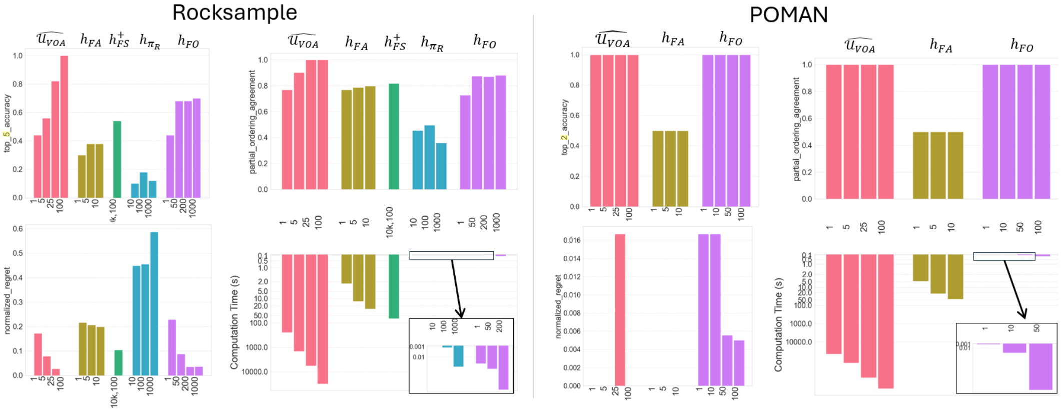

Figure 5: Results for both domains. [top left] Top-5/Top-2 accuracy for RockSample/POMAN, [top right] Partial Order Agreement, [bottom right] Computation Time, [bottom left] Normalized regret.

The baseline Monte Carlo VOA estimation clearly demonstrates the impact of computational resources. Increasing the number of sampled initial states consistently improves metric performance. However, computation times show that this approach is impractical in real-time.

The Full-Information Value heuristic (h_{FO}) demonstrated superior performance in both domains. In RockSample, with 200 initial state samples, it achieved a partial agreement score of 0.88 and maintained computation times under 0.1 seconds per evaluation. It reached near-zero regret in both domains (< 0.05 in RockSample and < 0.005 in POMAN). In POMAN, it consistently identified the two beneficial actions, reliably distinguishing them from less useful ones.

While the First-Action Value heuristic (h_{FA}) showed promising results in RockSample, and in some metrics in POMAN, it required 2-20 seconds of computation time due to its reliance on constructing complete MCTS trees for each sampled state. It also demonstrated limitations in capturing long-term effects, particularly evident in POMAN where it often identified only one of the two beneficial actions, as its value estimates are based solely on the immediate planning step rather than complete trajectory evaluation.

The Rollout-Policy heuristic ($h_{\pi_{\text{Rollout}}}$), despite being fastest computationally, performed poorly. In RockSample, it showed near-zero correlation with empirical VOA, while in POMAN, it proved entirely ineffective, generating zero VOA estimates for all actions. These results highlights both the challenge of policy evaluation and the significance of using actual planning policy rather than simplified approximations.

8 Conclusion

This work addresses a fundamental challenge in multi-robot collaboration: efficiently evaluating the potential benefit of helping actions for online POMDP agents. We formulate Value of Assistance (VOA) and develop computationally efficient heuristics to approximate it without requiring complete policy evaluation. Our empirical evaluation across both a standard POMDP benchmark and a robotic manipulation task demonstrates that the Full-Information heuristic achieves the best compute-value balance, providing reliable assistance decisions in under 0.1 seconds while consistently distinguishing beneficial actions from less useful ones. This work provides a foundation for developing collaborative robotic systems capable of principled real-time assistance decisions, with future directions including sequential assistance planning and broader multi-robot applications. A key extension is to relax the shared-belief assumption, enabling agents to collaborate effectively even when their beliefs diverge, an essential step toward robust real-world deployment.

9 Limitations

9.1 Single-Step Assistance

Our focus was on first step assistance. Nevertheless, our framework can be applied to evaluate helping actions throughout POMDP execution, not just at the initial step through receding horizon. However, considering sequences of future helping actions remains an open challenge. The current formulation assumes a single helping action, as the possibility of future assistance creates a significant distribution shift in the actor's value function - the helping agent would need to reason about states and beliefs differently knowing that help might be available later. This transforms the problem into a more complex domain of assistance planning over extended horizons, where the helper must reason about both immediate and future intervention opportunities. This limitation points to interesting future work in multi-step assistance strategies.

9.2 Actor Model Access

The effectiveness of our heuristics relies on access to the actor's POMDP model, planning algorithm parameters, and belief update function. While this assumption is reasonable in collaborative robotics settings where systems are designed to work together, it may limit applicability in scenarios where the actor's model or policy details are unknown or only partially observable.

9.3 Uncertainty Sources

Our evaluation focused primarily on scenarios where uncertainty arises from partial observability and stochastic transitions. Further investigation is needed to understand how other sources of uncertainty affect the relative performance of these heuristics - particularly epistemic uncertainty from model representation limitations and approximations.

References

[1] A. R. Cassandra, L. P. Kaelbling, and M. L. Littman. Acting optimally in partially observable stochastic domains. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 1023–1028, 1994.

[2] C. H. Papadimitriou and J. N. Tsitsiklis. The complexity of markov decision processes. *Mathematics of Operations Research*, 12(3):441–450, 1987. ISSN 0364765X, 15265471. URL http://www.jstor.org/stable/3689975.

[3] M. Lauri, D. Hsu, and J. Pajarinen. Partially observable markov decision processes in robotics: A survey. *IEEE Transactions on Robotics*, 39(1):21–40, Feb. 2023. ISSN 1941-0468. doi:10.1109/tro.2022.3200138. URL http://dx.doi.org/10.1109/TRO.2022.3200138.

[4] H. Kurniawati. Partially observable markov decision processes and robotics. *Annual Review of Control, Robotics, and Autonomous Systems*, 5(Volume 5, 2022):253–277, 2022. ISSN 2573-5144. doi:https://doi.org/10.1146/annurev-control-042920-092451. URL https://www.annualreviews.org/content/journals/10.1146/annurev-control-042920-092451.

[5] R. Simmons, J. Fern, R. Goodwin, and S. Koenig. Xavier: An autonomous mobile robot on the web. 04 1999.

[6] A. Goldhoorn, A. Garrell, R. Alquezar, and A. Sanfeliu. Searching and tracking people with cooperative mobile robots. *Autonomous Robots*, 42, 04 2018. doi:10.1007/s10514-017-9681-6.

[7] H. Bai, S. Cai, N. Ye, D. Hsu, and W. Lee. Intention-aware online pomdp planning for autonomous driving in a crowd. *Proceedings - IEEE International Conference on Robotics and Automation*, 2015:454–460, 06 2015. doi:10.1109/ICRA.2015.7139219.

[8] A. Curtis, L. P. Kaelbling, and S. Jain. Task-directed exploration in continuous pomdps for robotic manipulation of articulated objects. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 7200–7207. IEEE, 2023.

[9] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1):99–134, 1998. ISSN 0004-3702. doi:https://doi.org/10.1016/S0004-3702(98)00023-X. URL https://www.sciencedirect.com/science/article/pii/S000437029800023X.

[10] S. Paul, M. Nicolescu, and M. Nicolescu. Enhancing human–robot collaboration through a multi-module interaction framework with sensor fusion: Object recognition, verbal communication, user of interest detection, gesture and gaze recognition. *Sensors*, 23:5798, 06 2023. doi:10.3390/s23135798.

[11] J. Capitan, M. T. Spaan, L. Merino, and A. Ollero. Decentralized multi-robot cooperation with auctioned pomdps. In *2012 IEEE International Conference on Robotics and Automation*, pages 3323–3328, 2012. doi:10.1109/ICRA.2012.6224917.

[12] D. S. Bernstein, S. Zilberstein, and N. Immerman. The complexity of decentralized control of markov decision processes, 2013. URL https://arxiv.org/abs/1301.3836.

[13] C. Amato, G. Konidaris, G. Cruz, C. A. Maynor, J. P. How, and L. P. Kaelbling. Planning for decentralized control of multiple robots under uncertainty. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1241–1248. IEEE, 2015.

[14] R. G. Freedman, S. J. Levine, B. C. Williams, and S. Zilberstein. Helpfulness as a key metric of human-robot collaboration, 2020. URL https://arxiv.org/abs/2010.04914.

[15] A. Amuzig, D. Dovrat, and S. Keren. Value of assistance for mobile agents. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3484–3491. IEEE, 2023.

[16] T. Smith and R. Simmons. Point-based pomdp algorithms: Improved analysis and implementation, 2012. URL https://arxiv.org/abs/1207.1412.

[17] D. Silver and J. Veness. Monte-carlo planning in large pomdps. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 23, pages 2164–2172. Curran Associates, Inc., 2010. URL https://proceedings.neurips.cc/paper_files/paper/2010/file/edfbe1afcf9246bb0d40eb4d8027d90f-Paper.pdf.

[18] N. Ye, A. Somani, D. Hsu, and W. S. Lee. Despot: Online pomdp planning with regularization. *Journal of Artificial Intelligence Research*, 58:231–266, Jan. 2017. ISSN 1076-9757. doi:10.1613/jair.5328. URL http://dx.doi.org/10.1613/jair.5328.

[19] D. Salmond and N. J. Gordon. An introduction to particle filters. 2006. URL https://api.semanticscholar.org/CorpusID:777437.

[20] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256, 2002. URL https://api.semanticscholar.org/CorpusID:207609497.

[21] C. Basich, J. Peterson, and S. Zilberstein. Planning with intermittent state observability: Knowing when to act blind. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11657–11664, 2022. doi:10.1109/IROS47612.2022.9981883.

[22] K. Wray and S. Zilberstein. A pomdp formulation of proactive learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1), Mar. 2016. doi:10.1609/aaai.v30i1.10400. URL https://ojs.aaai.org/index.php/AAAI/article/view/10400.

[23] G. Pokharel. Increasing the value of information during planning in uncertain environments, 2024. URL https://arxiv.org/abs/2409.13754.

[24] C. Song, C. Zhang, A. Shafieezadeh, and R. Xiao. Value of information analysis in non-stationary stochastic decision environments: A reliability-assisted pomdp approach. *Reliability Engineering and System Safety*, 217:108034, 2022. ISSN 0951-8320. doi:https://doi.org/10.1016/j.ress.2021.108034. URL https://www.sciencedirect.com/science/article/pii/S095183202100541X.

[25] M. Saifullah, C. Andriotis, and K. Papakonstantinou. The role of value of information in multi-agent deep reinforcement learning for optimal decision-making under uncertainty. 2023. URL https://icasp14.com/. 14th International Conference on Applications of Statistics and Probability in Civil Engineering 2023, ICASP14 ; Conference date: 09-07-2023 Through 13-07-2023.

[26] N. Armstrong-Crews and M. Veloso. Oracular partially observable markov decision processes: A very special case. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 2477–2482, 2007. doi:10.1109/ROBOT.2007.363691.

[27] Mausam and A. Kolobov. *Planning with Markov Decision Processes: An AI Perspective*. Morgan & Claypool Publishers, 2012. ISBN 1608458865.

[28] A. K. Mausam and D. S. Weld. Determinize, solve, and generalize: Classical planning for mdp heuristics. 2009. URL https://api.semanticscholar.org/CorpusID:13457525.

[29] H. Geffner and B. Bonet. *A Concise Introduction to Models and Methods for Automated Planning*, volume 26 of *Synthesis Lectures on Artificial Intelligence and Machine Learning*. Morgan & Claypool Publishers, 2013. ISBN 978-1-60845-969-8. doi:10.2200/S00513ED1V01Y201306AIM022.

[30] Z. Sunberg and M. Kochenderfer. Online algorithms for pomdps with continuous state, action, and observation spaces, 2018. URL https://arxiv.org/abs/1709.06196.

[31] N. Ye, A. Somani, D. Hsu, and W. S. Lee. Despot: Online pomdp planning with regularization. *Journal of Artificial Intelligence Research*, 58:231–266, Jan. 2017. ISSN 1076-9757. doi:10.1613/jair.5328. URL http://dx.doi.org/10.1613/jair.5328.

[32] J. M. Porta, N. Vlassis, M. T. Spaan, and P. Poupart. Point-based value iteration for continuous pomdps. *J. Mach. Learn. Res.*, 7:2329–2367, dec 2006. ISSN 1532-4435.

[33] T. Cheng, L. Song, Y. Ge, W. Liu, X. Wang, and Y. Shan. Yolo-world: Real-time open-vocabulary object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16901–16911, June 2024.

[34] B. Efron. Bootstrap Methods: Another Look at the Jackknife. *The Annals of Statistics*, 7 (1):1 – 26, 1979. doi:10.1214/aos/1176344552. URL https://doi.org/10.1214/aos/1176344552.

[35] K. Zheng and S. Tellex. pomdp_py: A framework to build and solve pomdp problems. In *ICAPS 2020 Workshop on Planning and Robotics (PlanRob)*, 2020. URL https://icaps20subpages.icaps-conference.org/wp-content/uploads/2020/10/14-PlanRob_2020_paper_3.pdf. Arxiv link: ”https://arxiv.org/pdf/2004.10099.pdf”.

1 Appendix

1.1 Analytical Derivation of the Value of Assistance

In this appendix, we provide the complete derivation of VOA formula presented in Section 4. We show how to express the VOA using the helper's transition and observation models, starting from its definition in terms of expected value difference.

In Equation 1, the expected value after assistance depends on two distributions: $P(s'|\beta, \alpha)$ which represents the probability of transitioning to state s' given the current belief and helping action, and $\mathcal{O}_{\mathcal{H}}(\omega|s', \alpha)$ which is defined in our model. We aim to rewrite the equation for VOA by expressing $P(s'|\beta, \alpha)$ using the transition function $\mathcal{T}_{\mathcal{H}}(s'|s, \alpha)$ which is conditioned on the actual state rather than on the belief.

To achieve this, we first note that:

$$P(s'|\beta, \alpha) = \int_{s \in \mathcal{S}} P(s'|s, \beta, \alpha) P(s|\beta) ds \tag{7}$$

$$= \int_{s \in \mathcal{S}} P(s'|s, \alpha) P(s|\beta) ds \tag{8}$$

$$= \int_{s \in \mathcal{S}} \mathcal{T}_{\mathcal{H}}(s'|s, \alpha) \beta(s) ds \tag{9}$$

In Equation 7 we use the law of total probability. In 8 we use the fact that given the state is known, the probability of transitioning to s' does not depend on the belief. In Equation 9 we plug in $\mathcal{T}_{\mathcal{H}}$ and β in their definition.

Now we can use that to write the first term in VOA as follows:

$$\mathbb{E}_{\substack{s' \sim P(s'|\beta, \alpha) \\ \omega \sim \mathcal{O}_{\mathcal{H}}(\omega|s', \alpha)}} [V^\pi(\beta')] \tag{10}$$

$$= \int_{s'} \int_{\omega} P(s'|\beta, \alpha) \mathcal{O}_{\mathcal{H}}(\omega|s', \alpha) V^\pi(\beta') d\omega ds' \tag{11}$$

$$= \int_{s'} \int_{\omega} \int_{s} \mathcal{T}_{\mathcal{H}}(s'|s, \alpha) \beta(s) \mathcal{O}_{\mathcal{H}}(\omega|s', \alpha) V^\pi(\beta') ds d\omega ds' \tag{12}$$

$$= \int_{s} \int_{s'} \int_{\omega} \mathcal{T}_{\mathcal{H}}(s'|s, \alpha) \mathcal{O}_{\mathcal{H}}(\omega|s', \alpha) \beta(s) V^\pi(\beta') d\omega ds' ds \tag{13}$$

$$= \mathbb{E}_{s \sim \beta}[\mathbb{E}_{s' \sim \mathcal{T}_{\mathcal{H}}(s'|s, \alpha)}[\mathbb{E}_{\omega \sim \mathcal{O}_{\mathcal{H}}(\omega|s', \alpha)}[V^\pi(\beta')]]] \tag{14}$$

$$= \mathbb{E}_{s \sim \beta}[\mathbb{E}_{\substack{s' \sim \mathcal{T}_{\mathcal{H}}(\cdot|s, \alpha) \\ \omega \sim \mathcal{O}_{\mathcal{H}}(\cdot|s', \alpha)}} [V^\pi(\beta')]] \tag{15}$$

$$= \mathbb{E}_{s \sim \beta}[\mathbb{E}_{\substack{s' \sim \mathcal{T}_{\mathcal{H}}(\cdot|s, \alpha) \\ \omega \sim \mathcal{O}_{\mathcal{H}}(\cdot|s', \alpha)}} [V^\pi(\tau_{\mathcal{H}}(\beta, \alpha, \omega))]] \tag{16}$$

Note that while the initial belief β is a distribution over s, the updated belief β' is a distribution over the post-action states s'.

In Equation 11 we write the nested expectations explicitly as integrals. In Equation 12, we use Equation 9 to express the probability $P(s'|\beta, \alpha)$ in terms of $\mathcal{T}_{\mathcal{H}}$ and β. In Equation 13 we change the order of integration, according to Fubini's Theorem, this change is valid if the integrand is absolutely integrable. This integrability is justified because $\mathcal{T}_{\mathcal{H}}$, $\mathcal{O}_{\mathcal{H}}$ and β are probability distributions (which are positive and integrable by definition), and V^π is bounded in practical POMDP applications, which implies absolute integrability of the whole integrand. In Equation 14, we rewrite the three integrals as the nested expectations they represent, explicitly showing how the value depends on sampling the initial state from the belief, the next state from the transition model, and the observation from the sensor model. The last equation provides an alternative representation using the joint distribution over next states and observations.

Leveraging equation 15, we can express the value of assistance in terms of the model's distributions:

$$\mathcal{U}_{VoA}(\alpha, \beta) = \mathbb{E}_{s \sim \beta}[\mathbb{E}_{\substack{s' \sim \mathcal{T}_{\mathcal{H}}(\cdot|s,\alpha) \\ \omega \sim \mathcal{O}_{\mathcal{H}}(\cdot|s',\alpha)}} [V^{\pi}(\beta')]] - V^{\pi}(\beta) \tag{17}$$

1.2 Complexity Analysis Details for Algorithm 1

The computational complexity of Algorithm 1 depends on the following parameters:

- k: number of sampled initial states for VOA estimation
- L: episode length
- N_s: number of tree simulations in POMCP
- N_D: maximum simulation depth in POMCP
- N_P: number of particles representing the belief

Each policy query requires constructing and searching a tree with N_s tree simulations of maximum depth N_D. During each simulation, particles are updated along the traversed paths through the tree, resulting in $O(N_s N_D)$ time for a single policy query [17]. Each belief update operation, whether after receiving helper observation or during particle reinvigoration following action execution, requires $O(N_P)$ operations.

The algorithm consists of two phases described in lines 6-8 and 9-10: helping action simulation (sampling states and observations in $O(1)$ time and updating belief in $O(N_P)$ time) followed by policy evaluation episodes. During each episode, the steps in lines 22-26 are repeated L times: a POMCP policy query ($O(N_s N_D)$), state and observation sampling ($O(1)$), and particle reinvigoration ($O(N_P)$). Therefore, each iteration requires $O(L(N_s N_D + N_P))$ time.

For the complete algorithm with k iterations, this leads to a total computational complexity of $O(kL(N_s N_D + N_P))$. This complexity can be broken down as follows:

- Each POMCP tree construction and search: $O(N_s N_D)$
- Each belief update: $O(N_P)$
- Each episode (L steps): $O(L(N_s N_D + N_P))$
- Two episodes per iteration: $O(2L(N_s N_D + N_P))$ = $O(L(N_s N_D + N_P))$
- k iterations: $O(kL(N_s N_D + N_P))$

This analysis assumes that state sampling and observation sampling operations take constant time $O(1)$, which is typical for most POMDP implementations. If these operations have significant computational cost in specific applications, their complexity would need to be added to the per-step cost.

1.3 Discounted Reward Version of Algorithm 1

Algorithm 2 presents the discounted infinite-horizon extension of our VOA estimation approach. The key modification involves incorporating a discount factor $\gamma \in [0, 1)$ to properly weight future rewards. The computational complexity remains $O(kL(N_s N_D + N_P))$, identical to the finite-horizon version.

1.4 Heuristics for VOA: Detailed Descriptions

1.4.1 First-Action Value Heuristic h_{FA}

The main computational burden in Algorithm 1 comes from performing complete policy evaluation episodes, where each step requires planning and belief updates (lines 22-26). To address this, our first heuristic h_{FA} approximates the value of assistive action by planning only for the first step with and without assistance, using the same MCTS search approach used by the agent during execution (as illustrated in Figure 3 [mid-left]).

For each sampled initial state, we first compute POMCP's value estimate from the original belief $V_{\text{root}}(\beta)$, then simulate the effect of assistance to obtain an updated belief and compute POMCP's

Algorithm 2: VOA Prediction Through Policy Evaluation Episodes (Discounted Infinite Horizon)

```
1  Input: POMDP Model $\mathcal{M}$, Actor's Policy $\pi$, Belief $\beta$, Helping Action
       $\alpha$, Discount Factor $\gamma$, Resource Limit res$_{\max}$, Max Iterations $k$,
       Simulation Horizon $H$
   Output: Predicted Value of Assistance $\widehat{\mathcal{U}_{VoA}}(\beta, \alpha)$
2  Initialize empty arrays $V_\Delta$ and $S$;
3  res$_{\text{used}} \leftarrow 0$;
4  $i \leftarrow 0$;
5  while ($i < k$ and $res_{used} < res_{max}$) do
6      Sample initial state $s_i \sim \beta$;
7      Sample $s_i^{\mathcal{H}} \sim \mathcal{T}_{\mathcal{H}}(\cdot|s_i, \alpha)$ ;            // State after help
8      Sample $o^{\mathcal{H}} \sim \mathcal{O}_{\mathcal{H}}(\cdot|s_i^{\mathcal{H}}, \alpha)$ ;   // Observation after help
9      $\beta^{\mathcal{H}} \leftarrow \tau(\beta, \alpha, o^{\mathcal{H}})$ ;                  // Belief after help
10     $G_{help} \leftarrow$ EpSim($\mathcal{M}, s_i^{\mathcal{H}}, \beta^{\mathcal{H}}, \pi, \gamma, H$);     // Return with
         help
11     $G_{no\_help} \leftarrow$ EpSim($\mathcal{M}, s_i, \beta, \pi, \gamma, H$) ; // Return no help
12     $V_\Delta$.append($G_{help} - G_{no\_help}$);
13     $S$.append($s_i$);
14     $i \leftarrow i + 1$;
15     Update(res$_{\text{used}}$);
16 $\widehat{\mathcal{U}_{VoA}}(\beta, \alpha) \leftarrow \frac{\sum_{j=1}^{i} \beta(S[j]) V_\Delta[j]}{\sum_{j=1}^{i} \beta(S[j])}$ ;
17 return $\widehat{\mathcal{U}_{VoA}}(\beta, \alpha)$ ;

   // Note: This function simulates a
      complete policy evaluation episode
      with discounting.
18 Function EpSim($\mathcal{M}, s, \beta, \pi, \gamma, H$):
19     $G \leftarrow 0$;
20     $s_{curr} \leftarrow s$;
21     $\beta_{curr} \leftarrow \beta$;
22     for $t \leftarrow 1$ to $H$ do
23         $a \leftarrow \pi(\beta_{curr})$ ;       // Constructs a
             search tree
24         Sample $s_{next} \sim \mathcal{T}(\cdot|s_{curr}, a)$;
25         Sample $o \sim \mathcal{O}(\cdot|s_{next}, a)$;
26         $G \leftarrow G + \gamma^{t-1}\mathcal{R}(s_{curr}, a)$;
27         $\beta_{curr} \leftarrow \tau(\beta_{curr}, a, o)$;
28         $s_{curr} \leftarrow s_{next}$;
29     return $G$
```

value estimate from this new belief $V_{\text{root}}(\tau_{\mathcal{H}}(\beta, \alpha, \omega))$. The value estimates are obtained from the root node of the constructed MCTS search tree.

It is important to note that POMCP's value estimates should not be interpreted as accurate estimates of the true value function. POMCP combines tree search with rollouts from leaf nodes, typically using a basic policy rather than the full planning policy that would be used during actual execution. Additionally, the search tree typically explores only a fraction of the full planning horizon, with leaf nodes evaluated using truncated rollouts. This means that the value estimates partially rely on these simplified evaluations.

To estimate VOA, we replace the policy value function V^π in the original VOA equation with POMCP's root node value estimate V_{root}:

$$h_{FA}(\alpha, \beta) = \mathbb{E}_{s\sim\beta}[\mathbb{E}_{\substack{s'\sim\mathcal{T}_{\mathcal{H}}(s'|s,\alpha) \\ \omega\sim\mathcal{O}_{\mathcal{H}}(\omega|s',\alpha)}}[V_{\text{root}}(\tau_{\mathcal{H}}(\beta, \alpha, \omega))]] - V_{\text{root}}(\beta) \tag{18}$$

Returning to algorithm 1, instead of computing complete episode returns in lines 9 and 10, h_{FA} performs a single planning step and uses the root node value. The elimination of the L-step rollouts in the algorithm's episode simulation function significantly reduces computation by requiring only a single planning step at each sampled state. The computational complexity reduces from $O(kL(N_sN_D + N_P))$ to $O(k(N_sN_D + N_P))$.

1.4.2 Rollout-Policy Heuristic $h_{\pi_{\textbf{Rollout}}}$

Another perspective on the main computational burden in Algorithm 1 focuses on the repeated construction of search trees for policy queries during episode simulation (line 23). Each tree construction itself may require thousands of rollout policy invocations at leaf nodes. Our second heuristic addresses this by utilizing the planner's rollout policy directly. While this policy may be purely random in some implementations, it can also incorporate domain knowledge when available [17].

Returning to algorithm 1, instead of constructing a search tree in line 23, we directly use the rollout policy π_{Rollout} for action selection (Figure 3 [mid-right]). To compute VOA, we replace the planning policy value function V^π in the original VOA equation with the value function of the rollout policy $V^{\pi_{\text{Rollout}}}$:

$$h_{\pi_{\text{Rollout}}}(\alpha, \beta) = \mathbb{E}_{s\sim\beta}[\mathbb{E}_{\substack{s'\sim\mathcal{T}_{\mathcal{H}}(s'|s,\alpha) \\ \omega\sim\mathcal{O}_{\mathcal{H}}(\omega|s',\alpha)}}[V^{\pi_{\text{Rollout}}}(\beta')]] - V^{\pi_{\text{Rollout}}}(\beta) \tag{19}$$

where $V^{\pi_{\text{Rollout}}}$ represents the value function when following the rollout policy. Since this policy is designed for rapid computation (being executed approximately $N_s \times N_D$ times during each tree construction), using it directly for policy evaluation drastically reduces computational cost compared to full planning.

The rollout policy inherently underestimates the true value function since it does not consider the long-term effect of decisions. However, the difference in values can still serve as a meaningful heuristic since the rollout policy's performance is affected by the same state and observation uncertainties that impact the full planning policy, and improvements in belief quality or state configuration may benefit both policies.

The computational complexity for k iterations reduces from $O(kL(N_sN_D + N_P))$ to $O(kL(N_P))$ since we eliminate the expensive planning operations (N_sN_D term). This significant reduction in computation per iteration allows for larger values of k, potentially providing stable value estimates.

1.4.3 Full-Information Heuristic h_{FO}

The challenge of estimating VOA stems largely from reasoning about partial observability and stochastic transitions in POMDPs. Our third heuristic addresses this by using a fully observable deterministic relaxation of the problem ((Figure 3 [right])).

Instead of performing policy evaluation episodes in lines 10 and 11 of Algorithm 1, we create a fully observable deterministic variant of the POMDP through all-outcome determinization, which means that the agent can choose the most favorable next state from the support of each action's transition distribution. For each sampled state, we perform deterministic planning from the fully observable state, computing returns both with and without the effects of assistance.

To compute VOA, we replace the policy value function V^π with the deterministic optimal plan value U, and remove the observation-related expectations since we use the full observability relaxation:

$$h_{FO}(\alpha, \beta) = \mathbb{E}_{s \sim \beta}[\mathbb{E}_{s' \sim \mathcal{T}_{\mathcal{H}}(\cdot|s,\alpha)}[U(s')] - U(s)] \tag{20}$$

where s is sampled from the current belief β, and $U(s)$ represents the accumulated reward under the optimal deterministic plan from state s.

This converts the complex POMDP planning problem into a much simpler deterministic planning problem, inspired by [27, 28], benefiting from decades of research in deterministic planning algorithms and heuristics [29]. Even a simple breadth-first search often performs well in this setting, as the state space that makes POMDPs intractable becomes manageable when eliminating both belief space planning and transition uncertainty. When available, domain-specific heuristics can further accelerate the planning process.

The computational complexity for k iterations becomes $O(k|S||A|B)$ for basic state-space search, where $|S|$ and $|A|$ represent the number of states and actions respectively, and B is the maximum branching factor in the transition support.

While this heuristic efficiently evaluates potential state changes, it would be inappropriate for actual POMDP action selection since it sidesteps fundamental POMDP challenges by ignoring both state uncertainty and stochastic transitions. Nevertheless, this simplification may be suitable for approximating the value of assistance. In appendix 1.5 we show that when the planner can select outcomes from the transition support, $U(s)$ provides an upper bound on the optimal POMDP value function V^*, and therefore also bounds any policy value V^π.

However, this approach has limitations in highly stochastic domains where actions can lead to many possible next states. In such cases, the large transition support makes outcome selection computationally expensive. A practical alternative is to use maximum likelihood determinization, selecting only the most probable next state for each action, though this forfeits the upper bound guarantee.

It is worth noting that while $U(s)$ provides an upper bound for the POMDP value function, we cannot guarantee that VOA computed using all-outcome determinization provides an upper bound on the actual VOA. This is because VOA measures differences between value functions - even when each individual value is upper bounded, their difference may be larger or smaller than the difference between the actual POMDP values.

1.5 Upper Bound Proof for Full-Information Value

We show that the value under all-outcome determinization provides an upper bound on the optimal POMDP value.

For a given belief β, let $V^*(\beta)$ denote the optimal value function:

$$V^*(\beta) = \max_\pi \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(\beta_t))|\beta_0 = \beta]$$

where the expectation is over state sequences generated according to the POMDP dynamics when following policy π.

Let π^* be a policy achieving this maximum. For a given state s, let $V^{\pi^*}(\beta|s)$ denote the value of executing this belief-based policy starting from state s:

$$V^{\pi^*}(\beta|s) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi^*(\beta_t))|s_0 = s, \beta_0 = \beta]$$

Let $U(s)$ denote the value of the optimal deterministic plan under full observability with all-outcome determinization, where at each step the planner can choose any next state from the support of the transition function.

Lemma 1. *For any state s:*

$$U(s) \geq V^{\pi^*}(\beta|s)$$

Proof. Consider the probability distribution over state-action sequences $(s_0, a_0, s_1, a_1, ...)$ induced by following π^* starting from $s_0 = s$. The value $V^{\pi^*}(\beta|s)$ is the expectation of $\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)$ over this distribution.

Since this is an expectation over a discrete probability space (state-action sequences), there must exist at least one sequence with value at least $V^{\pi^*}(\beta|s)$. Let $(s_0^*, a_0^*, s_1^*, a_1^*, ...)$ be such a sequence.

Under all-outcome determinization with full observability:

- At each step t, the planner observes state s_t^* exactly
- By construction, s_{t+1}^* is in the support of $T(\cdot|s_t^*, a_t^*)$
- The planner can choose s_{t+1}^* as the next state and a_t^* as the action

Therefore, the deterministic planner can construct a plan achieving value at least $V^{\pi^*}(\beta|s)$ by following this sequence. Since $U(s)$ represents the optimal such plan value, $U(s) \geq V^{\pi^*}(\beta|s)$. □

Theorem 1. *For any belief β:*

$$\mathbb{E}_{s\sim\beta}[U(s)] \geq V^*(\beta)$$

Proof. By definition of $V^*(\beta)$ and linearity of expectation:

$$V^*(\beta) = \mathbb{E}_{s\sim\beta}[V^{\pi^*}(\beta|s)]$$

From the lemma, we know that $U(s) \geq V^{\pi^*}(\beta|s)$ for all states s. Therefore:

$$\mathbb{E}_{s\sim\beta}[U(s)] \geq \mathbb{E}_{s\sim\beta}[V^{\pi^*}(\beta|s)] = V^*(\beta)$$

□

1.6 Discussion About Robustness to Belief Divergence

Our approach assumes both helper and actor agents maintain the same belief state. When this assumption is violated, the helper's VOA estimates are based on its own belief rather than the actor's true belief, potentially leading to suboptimal assistance decisions.

The impact severity depends on the degree of belief divergence and VOA sensitivity to belief differences. Small divergences in localized uncertainties (e.g., object positions) may have minimal impact since the relative ranking of helping actions often remains stable, while significant divergences about critical state variables can lead to poor assistance choices.

Several factors mitigate this limitation: frequent communication of observations can maintain belief synchronization, robust helping actions provide value across belief ranges, and collaborative tasks naturally create shared observation opportunities that align beliefs over time. Future work should investigate adaptive VOA estimation under belief uncertainty and communication strategies balancing synchronization with bandwidth constraints.

1.7 Comparison to Joint POMDP Planning

An alternative approach would be to formulate the problem as a joint POMDP, treating both robots as a single agent with combined state and action spaces (with helping actions included). This approach would naturally handle multi-step coordination and could find globally optimal policies.

However, our VOA approach addresses a different scenario: situations where the actor agent uses an existing, deployed policy that cannot be modified. This is common in heterogeneous systems or when integrating assistance into existing robotic systems. Joint POMDP planning also scales exponentially with combined state and action spaces and requires recomputing the entire joint policy, while our approach scales linearly with the number of helping actions and leverages existing planning capabilities. Our VOA framework is designed for applications where joint policy redesign is impractical, providing a computationally tractable solution.

1.8 Empirical Evaluation Setup

1.8.1 POMCP Implementation

We use POMCP through the pomdp_py library [35], which provides an efficient implementation with Cython-accelerated core components. For RockSample, we used $N_s = 2000$ simulations per step with maximum depth of $N_D = 20$. For the POMAN, we used $N_s = 8000$ simulations with maximum depth of $N_D = 35$.

In both environments, each POMCP planning step takes approximately 1-3 seconds. While this computational time is reasonable during task execution, it becomes a significant bottleneck when evaluating VOA through policy evaluation episodes, as each iteration requires multiple planning steps. This computational cost motivates our development of efficient heuristics.

1.8.2 Empirical VOA for Evaluation

Since the true VOA cannot be directly computed in our setting, we establish empirical VOA values through extensive offline computation to serve as ground truth for evaluating our heuristics. While this process is computationally prohibitive for real-world robotic assistance, we invest substantial computational resources to obtain reliable estimates purely for evaluation purposes.

For each belief-helping action pair, we conducted extensive policy evaluation episodes following Algorithm 1 with k=100 sampled states. To account for policy stochasticity, we simulated three episodes per state and averaged their returns, resulting in 300 total episode returns per pair. With each planning step taking 1-3 seconds and each evaluation requiring multiple planning steps across multiple episodes, a single belief-helping action pair evaluation took several hours.

Using this simulation data, we computed empirical VOA values and established confidence intervals through bootstrap resampling [34]. Specifically, for each bootstrap iteration, we sampled states with replacement from our collected data and computed VOA using importance sampling as in line 15 of Algorithm 1. This process was repeated 1,000 times with 100 resampled states per iteration, and we used the 2.5 and 97.5 percentiles of the resulting VOA distribution to establish 95% confidence intervals. To analyze how the full algorithm performs with different computational budgets, we also evaluate our heuristics against VOA values computed using varying sizes of subsets from this collected data.

1.8.3 Heuristics Configurations

We implemented each of the three heuristics as described in Section 6. We evaluate each heuristic across varying numbers of sampled initial states to assess their stability and computational requirements. For RockSample, we additionally tested an enhanced variant of the First-Action Value heuristic that dedicates more computational resources to the single planning step (10,000 simulations, depth 100) compared to the basic version using task execution parameters.

The Rollout-Policy heuristic uses the same domain-specific policies that serve as rollout policies in our POMCP implementation: for RockSample, we used the rollout policy described in [17], while for the POMAN, we employed a simple reward-maximizing policy.

1.9 Evaluation Metrics Details and Additional Results

1.9.1 Primary Metrics (Reported in Main Paper)

Partial Order Agreement. This metric evaluates how well a heuristic respects the ordering of helping actions when we can be confident about their relative values according to our confidence intervals. It is particularly valuable because it accounts for uncertainty in our empirical VOA estimates, only considering pairs of actions where we have statistical confidence about their ordering. For a belief β and helping actions α_j, α_k, let $[\text{CI}_{\text{low}}(\alpha), \text{CI}_{\text{high}}(\alpha)]$ denote the confidence interval of empirical VOA for helping action α. We say $\alpha_j \succ \alpha_k$ if $\text{CI}_{\text{low}}(\alpha_j) > \text{CI}_{\text{high}}(\alpha_k)$, establishing a strict partial ordering. Let P_β be the set of all such strictly ordered pairs for belief β. The agreement score for heuristic h is:

$$\text{Agree}(h, \beta) = \frac{|\{(\alpha_j, \alpha_k) \in P_\beta : h(\alpha_j, \beta) > h(\alpha_k, \beta)\}|}{|P_\beta|}$$

The final score is averaged across all beliefs. Score values range from 0 to 1, with 1 indicating perfect agreement.

Normalized Regret. This metric directly measures the practical impact of using a heuristic for action selection by quantifying how much value is lost by choosing the heuristic's recommended action rather than the truly optimal one. It answers the question: ”How much potential assistance value do we sacrifice by relying on this heuristic?” For each belief β, let α_h^* be the action with the highest heuristic value and α_E^* be the action with the highest empirical VOA. The normalized regret is:

$$\text{Regret}(h, \beta) = \frac{\text{VOA}(\alpha_E^*, \beta) - \text{VOA}(\alpha_h^*, \beta)}{\max_\alpha \text{VOA}(\alpha, \beta) - \min_\alpha \text{VOA}(\alpha, \beta)}$$

The normalization ensures values between 0 and 1, with lower values indicating better performance. A value of 0 means the heuristic selected the optimal action, while a value approaching 1 indicates the heuristic chose one of the worst possible actions.

Top-k Accuracy. This metric focuses on the heuristic's ability to identify the most valuable helping actions, which is crucial in real-world scenarios where we need to consider multiple high-value options. It measures what fraction of the truly best k actions are captured by the heuristic's top k predictions. Let $T_k^E(\beta)$ and $T_k^h(\beta)$ be the sets of top k helping actions according to empirical VOA and heuristic h respectively for belief β. Then:

$$\text{Top-k Accuracy}(h) = \frac{1}{|\mathcal{B}|} \sum_{\beta \in \mathcal{B}} \frac{|T_k^E(\beta) \cap T_k^h(\beta)|}{k}$$

where $|\mathcal{B}|$ is the number of beliefs evaluated, and $|T_k^E(\beta) \cap T_k^h(\beta)|$ represents the size of the intersection between the top k actions selected by ground truth and the heuristic. We report top-5 accuracy for RockSample (with approximately 50 possible actions depending on the environment instance) and top-2 accuracy for POMAN (with 4 possible actions) in the main paper. Values range from 0 to 1, with 1 indicating perfect identification of all top-k actions.

Computation Time. Average time in seconds required to compute the heuristic value for each belief-helping action pair. This metric directly measures the practical feasibility of using each heuristic in real-time scenarios.

Together, these metrics provide complementary views of heuristic performance: Partial Order Agreement captures reliability in comparing actions when we have high confidence in their relative values, Normalized Regret quantifies the practical impact of selection decisions, Top-k Accuracy measures the ability to identify groups of valuable actions, and Computation Time addresses real-world feasibility constraints.

1.9.2 Supplementary Metrics (Reported Only in Appendix)

Top-1 Accuracy. Similar to the top-k accuracy metric described above, but specifically measuring the heuristic's ability to identify the single best action ($k = 1$). This provides insight into how often the heuristic can pinpoint the absolutely optimal action. Values range from 0 to 1, with 1 indicating the heuristic always identifies the optimal action.

Top-k Selection Rate. While top-k accuracy measures how many of the k best actions are identified by the heuristic, this metric answers a different question: ”How often is the heuristic's top choice among the k best actions according to ground truth?” This is particularly relevant when we can only execute a single helping action and want to ensure it's at least among the better options. Formally, for each belief β:

$$\text{Top-k Selection Rate}(h) = \frac{1}{|\mathcal{B}|} \sum_{\beta \in \mathcal{B}} \mathbb{I}[\alpha_h^* \in T_k^E(\beta)]$$

where $\mathbb{I}$ is the indicator function, α_h^* is the action with highest heuristic value, and $T_k^E(\beta)$ is the set of top k actions according to empirical VOA. We report top-5 selection rate for RockSample and top-2 selection rate for POMAN. Values range from 0 to 1, with 1 indicating the heuristic's top choice is always among the top-k ground truth actions.

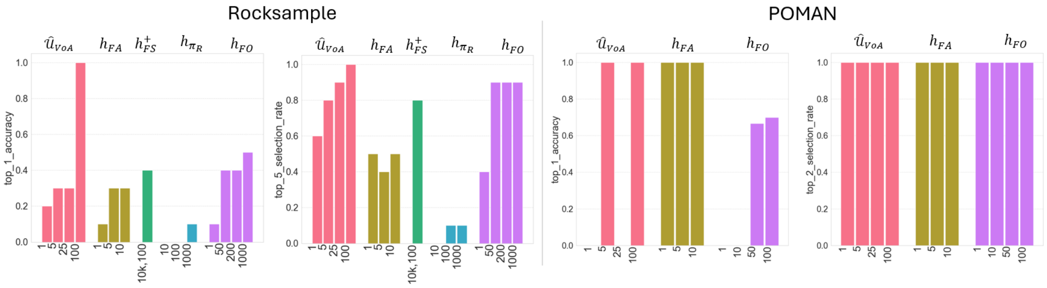

Figure 6: Supplementary metrics results for both domains. These metrics provide additional insights into heuristic performance beyond the primary metrics reported in the main paper.

1.9.3 POMCP Root Node Variance Analysis

Figure 7 presents a variance analysis of POMCP root node value estimates in the RockSample domain, addressing concerns about the reliability of our First-Action Value heuristic which depends on these estimates. The analysis shows the distribution of root node values across multiple POMCP runs with identical belief states but different random seeds.

The results reveal a coefficient of variation of approximately 40%, indicating substantial variance in POMCP root node estimates under realistic simulation budgets. This high variance is consistent with known limitations of Monte Carlo tree search methods in stochastic domains and explains some of the performance limitations observed for the First-Action Value heuristic (h_{FA}) in our experimental results.

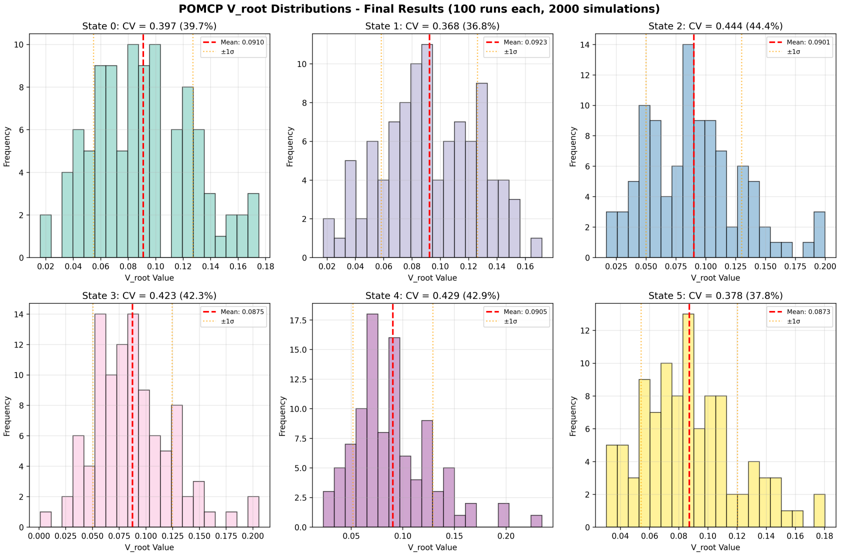

Figure 7: POMCP root node variance value analysis

1.10 RockSample Benchmark Details

We augment the RockSample(n,k) benchmark [32] with clustering help actions. These actions allow an assistant to reposition rocks to form clusters, making them both more efficiently observable and reducing the travel distance required between rocks, as the agent can check multiple rocks and collect the good ones while staying in the same area.

The helping actions are generated by identifying candidate rock clusters around three fixed positions in the second column of the grid (at n/4, n/2, and 3n/4 height), close to the agent's starting position at the leftmost column. For each position, we consider different subsets of rocks that could be moved to form a cluster. Rocks are selected using a sliding 6×6 window: for each window position in the grid, we consider combinations of up to 4 rocks within that window.

For each selected subset of rocks and target cluster position, we generate a configuration by attempting to place the rocks in available positions around the cluster center. A position is considered available if it is within grid bounds and not occupied by other rocks. The exact arrangement of rocks within the cluster is determined stochastically, as rocks are placed in random order around the center.

For VOA computation, these helping actions directly modify the state transition function by repositioning rocks. This affects both the agent's ability to efficiently observe multiple rocks with fewer sensing actions and its movement planning to collect valuable rocks. The VOA of each helping action depends on how much it reduces the agent's expected cost of completing the task, which varies based on the arrangement of valuable rocks (unknown to the helper) and the impact of clustering on sensing efficiency.

This formulation creates a challenging decision problem for the assistant. While clustering rocks near the agent's starting position can improve both sensing efficiency and motion planning, poor choices of which rocks to cluster or where to place them can actually harm performance by increasing travel distances to important rocks or making efficient observation sequences harder to execute.

1.11 Partially Observable Robotic Manipulation Environment Details

1.11.1 Environment Setup

The manipulation environment consists of a UR5e robotic arm operating in a workspace containing manipulated objects (cups and soda can) along with static obstacles and obstacles that can be moved by the assisting robot. The robot's motion planning relies on a pre-computed probabilistic roadmap (PRM) of approximately 400 configurations, where nodes represent robot configurations and edges represent feasible direct paths between them.

1.11.2 State Space

The state space includes:

- Robot configuration (position in the PRM)
- Positions of manipulated objects
- Positions of movable obstacles
- Task-specific states (cup placement status, liquid state)
- Gripper status (empty/holding object)
- Special roadmap configurations when holding the can to maintain vertical orientation

1.11.3 Action Space

The action space consists of:

- Navigation actions: Moving between connected configurations in the PRM (deterministic)
  - When holding the soda can, navigation is restricted to a specialized roadmap that maintains vertical orientation within a specified tolerance
  - This constraint limits motion in narrow spaces and affects reachability
- Manipulation actions: Available at specific configurations
  - Pick up object
  - Put down object
  - Pour liquid
- Sensing actions: Activate camera and process observations

1.11.4 Transition and Reward Model

- Navigation actions are deterministic but may become unavailable when obstacles block PRM edges or nodes
- Manipulation actions succeed with high probability but may fail with low probability
- Object positions change deterministically upon successful manipulation
- Rewards:
  - Negative step cost for all actions
  - Small positive rewards for placing cups on the table and disposing of the can
  - High reward for successfully pouring soda into cups

1.11.5 Observation Model

The agent receives observations about:

- Current robot configuration (fully observable)
- Gripper status (fully observable)
- Object positions (partially observable):
  - Only available when executing sensing actions
  - Detection probability depends on the number of objects in the field of view
  - Higher probability for closer objects and unoccluded views

1.11.6 Perception System Implementation

The perception system integrates YOLO-world object detection [33] with stereo depth camera data to create 3D position estimates of detected objects. These observations are used to update the agent's belief state through particle filtering. Each particle in the belief state represents a possible state configuration, and particles are weighted based on observation likelihood.

When executing a sensing action, the system:

1. Captures RGB and depth images from the robot's camera position
2. Processes the RGB image through YOLO-world to detect objects and their 2D bounding boxes
3. Maps detected objects to 3D positions using corresponding depth data
4. Updates the belief by reweighting particles based on how well they explain the observation
5. Performs particle reinvigoration when necessary to maintain distribution diversity

The VOA in this environment arises from both improved observability (removing occlusions to enhance perception) and improved accessibility (clearing paths to facilitate manipulation). Helping actions that move obstacles can significantly impact task performance by enabling the agent to more efficiently sense object positions and navigate to manipulation targets.

1.12 Extended Results Analysis

Our evaluation compared VOA estimation approaches across two domains with distinct characteristics: RockSample with approximately 50 possible helping actions, and POMAN with 4 actions naturally divided into beneficial (VOA $\approx$ 13) and minimal-benefit (VOA ¡ 0.5) groups.

1.12.1 RockSample Domain Analysis

Full-Information Heuristic With 200 initial states, this heuristic achieved a partial agreement score of 0.88 while maintaining computation times under 0.1 seconds per evaluation. Its effectiveness is demonstrated by 40% accuracy in selecting the optimal action and 68% overlap with the empirically top five actions. This heuristic's strength lies in effectively capturing how rock repositioning creates efficient observation and collection paths through deterministic planning.

First-Action Value Heuristic This heuristic showed promising results but required 2-20 seconds of computation time depending on the number of iterations. The enhanced variant (h^+_{FA}) with increased planning resources showed further improvements in metric performance. However, the longer computation times make it less practical for real-time assistance decisions when multiple helping actions must be evaluated.

Rollout-Policy Heuristic Despite being computationally fastest, this heuristic performed poorly across all metrics. Its near-zero correlation with empirical VOA highlights both the challenge of policy evaluation and the significance of using the actual planning policy rather than simplified approximations.

1.12.2 POMAN Domain Analysis

Full-Information Heuristic This heuristic consistently identified both beneficial actions across all sample sizes. Its normalized regret remained below 0.005, indicating near-optimal action selection. While not always selecting the single best action, it reliably distinguished the beneficial helping actions from the less useful ones.

First-Action Value Heuristic This heuristic showed mixed performance - while it successfully identified one of the beneficial actions, it often missed the second one as evident in the top-1 versus top-2 accuracy metrics. This limitation likely stems from evaluating actions based on a single planning step, which captures some but not all long-term effects of the complex manipulation task.

Rollout-Policy Heuristic This heuristic proved entirely ineffective in POMAN, generating zero VOA estimates for all actions regardless of sample size. Using the simple rollout policy, the agent could not make meaningful progress toward task completion in any scenario, resulting in identical performance with and without assistance.

1.12.3 Cross-Domain Observations

Our analysis reveals several key patterns:

1. **Computation-performance tradeoff:** Baseline Monte Carlo VOA estimation achieved high accuracy but at prohibitive computational cost, with computation times ranging from tens to hundreds of seconds per evaluation.
2. **Sample size stability:** Performance metrics showed relative stability when increasing number of samples, up to some value, across all heuristics, suggesting that moderate sample sizes are sufficient for reliable VOA estimation.
3. **Domain complexity impact:** The performance gap between First-Action Value and Full-Information heuristics was larger in POMAN than in RockSample, suggesting that domains with complex sequences of sensing and manipulation may require more sophisticated VOA estimation approaches.

In summary, the Full-Information heuristic provides the best balance of computational efficiency and accurate VOA estimation across diverse domains, enabling real-time assistance decisions in complex robotic scenarios.