Action-Gradient Monte Carlo Tree Search for Non-Parametric Continuous (PO)MDPs

Idan Lev-Yehudi[1], **Michael Novitsky**[1], **Moran Barenboim**[1], **Ron Benchetrit**[2] and **Vadim Indelman**[3,4]
[1]Technion Autonomous Systems Program (TASP), Technion – Israel Institute of Technology
[2]Faculty of Computer Science, Technion – Israel Institute of Technology
[3]Stephen B. Klein Faculty of Aerospace Engineering, Technion – Israel Institute of Technology
[4]Faculty of Data and Decision Sciences, Technion – Israel Institute of Technology
idanlev@campus.technion.ac.il, vadim.indelman@technion.ac.il,

Abstract

Online planning in continuous state, action, and observation spaces remains challenging for autonomous systems. While Monte Carlo Tree Search (MCTS) scales effectively via sampling, most continuous (PO)MDP solvers do not exploit gradient-based action optimization. We propose Action-Gradient MCTS (AGMCTS), a framework that combines global tree search with local gradient-based action refinement, while maintaining consistent value estimates. We provide three key theoretical contributions: (1) an action score gradient theorem for particle belief states; (2) the Multiple Importance Sampling (MIS) Tree that supports frequent action-branch updates by reusing prior samples without introducing estimator drift; and (3) tractable action score gradients for smooth generative models using the Area Formula. Empirical results demonstrate that AGMCTS outperforms state-of-the-art sample-based solvers in multiple challenging continuous MDP and POMDP benchmarks.

1 Introduction

Planning under uncertainty is central to AI and robotics, where state, action, and observation spaces are often continuous [Lauri *et al.*, 2022]. Markov Decision Processes (MDPs) and Partially Observable Markov Decision Processes (POMDPs) model such problems, but exact solutions are generally intractable [Papadimitriou and Tsitsiklis, 1987; Madani *et al.*, 2003], motivating efficient approximation.

Online solvers compute actions on demand from the current state or belief [Kurniawati, 2022], often via tree search such as Monte Carlo Tree Search (MCTS) [Silver and Veness, 2010] and Determinized Sparse Partially Observable Tree (DESPOT) [Ye *et al.*, 2017]. Earlier continuous-POMDP offline methods often used fixed-dimensional parametric beliefs, e.g., Gaussian beliefs [Porta *et al.*, 2006]; recent continuous MDP/POMDP online methods improve scalability via progressive widening, including Double Progressive Widening (DPW) for MDPs and POMCPOW/PFT-DPW for POMDPs [Couëtoux *et al.*, 2011; Sunberg and Kochenderfer, 2018].

Most sampling-based continuous POMDP planners keep sampled actions fixed. VG-UCT [Lee *et al.*, 2020] refines continuous-MDP actions, but does not correct value-estimate drift after action updates. Action-Gradient MCTS (AGMCTS) integrates gradient steps into global MCTS via MIS, and is the first such hybrid strategy for non-parametric continuous POMDPs with sample-based beliefs.

1.1 Contributions

Theoretical. We derive an action score gradient theorem for MDPs and POMDPs in online tree search with importance weighting. We introduce the Multiple Importance Sampling (MIS) Tree and its *action update* operation, allowing frequent gradient steps while maintaining value-estimate consistency. We also compute action score gradients tractably for smooth generative models via the Area Formula.

Algorithmic. We present AGMCTS (Algorithm 1), blending MCTS action exploration with online gradient refinement for MDPs/POMDPs under DPW or alternative continuous-action selection heuristics.

Experimental. Across continuous MDP and POMDP benchmarks, AGMCTS variants show notable gains in most domains and competitive performance overall.

1.2 Related Work

Online Optimization in MCTS. Lee *et al.* [2020] have proposed VG-UCT for integrating gradient steps into MCTS in continuous MDPs. Their Lipschitz-continuous transition and reward setting gives a deterministic-MDP convergence result; we give estimator gradients under weaker assumptions and address value-estimate drift from action updates.

Black-box optimization methods have been used to bias action selection in MCTS to more promising regions, e.g. Voronoi optimistic optimization (VOO) [Kim *et al.*, 2020] and Voronoi progressive widening (VPW) [Lim *et al.*, 2021; Hoerger *et al.*, 2024], kernel regression [Yee *et al.*, 2016] and Bayesian optimization [Morere *et al.*, 2018; Mern *et al.*, 2021] methods. These complement AGMCTS by improving action proposals rather than refinements.

(PO)MDP Action-Gradients. An action-gradient corresponds to the single-step gradient of a policy with a deterministic immediate action. Policy gradients in POMDPs typically assume stochastic policies [Baxter and Bartlett, 2001], memory-less policies [Azizzadenesheli *et al.*, 2018], or finite action spaces [Hong *et al.*, 2024], while we handle belief-dependent policies with continuous actions. Silver *et al.* [2014] studied deterministic policies in MDPs; we extend to non-parametric POMDPs.

Information Reuse in Planning. Leurent and Maillard [2020] considered information sharing in discrete MDPs by merging similar states. Novitsky *et al.* [2025] have introduced previous knowledge utilization in POMDPs via Multiple Importance Sampling (MIS) [Veach and Guibas, 1995]. We reuse samples across sibling action branches via the MIS Tree, avoiding retrieval from an external belief database. POMDP planning has used MIS for sample reuse, e.g. [Farhi and Indelman, 2021], mainly to compute observation likelihoods. AdaOps and CMCGS cluster values of similar observations/states [Wu *et al.*, 2021; Kujanpää *et al.*, 2024], whereas we recompute value estimates under updated actions.

2 Background

MDPs and POMDPs. We consider MDPs in the form $\langle \mathcal{S}, \mathcal{A}, p_T, r, \gamma, L, b_0 \rangle$. The state and action spaces are $\mathcal{S} \subseteq \mathbb{R}^{n_s}$ and $\mathcal{A} \subseteq \mathbb{R}^{n_a}$. The transition model $p_T(s'|s,a)$ is the probability of arriving at state s' when taking action $a \in \mathcal{A}$ at state $s \in \mathcal{S}$. The reward function $r(s,a,s') \in \mathbb{R}$ gives the immediate reward of transitioning from state s to s' by taking action a, and the expected reward is $r(s,a) \triangleq \mathbb{E}_{s'|s,a}[r(s,a,s')]$. The discount factor is $\gamma \in (0,1]$. The MDP starts at time 0 and terminates after $L \in \mathbb{N} \cup \{\infty\}$ steps, and if $L = \infty$ then we assume $\gamma < 1$. The initial state is drawn from the distribution $s_0 \sim b_0$.

We assume $\pi = (\pi_t)_{t=0}^{L}$ is a (possibly time-dependent) deterministic policy, but we note our theoretical results generalize to stochastic policies as well. The value function of a policy π at time t is the expected sum of discounted rewards until the horizon: $V_t^\pi(s_t) \triangleq \mathbb{E}_{s_{t+1:L}|s_t,\pi}[\sum_{i=t}^{L} \gamma^{i-t} r(s_i, \pi_i(s_i), s_{i+1})]$. We denote next-step variables with primes, e.g. s' follows s. We define the action-value function as $Q_t^\pi(s,a) \triangleq \mathbb{E}_{s'|s,a}[r(s,a,s') + \gamma V_{t+1}^\pi(s')]$. Our goal is to compute a policy that maximizes the value function, i.e. find $\pi^* \in \arg\max_\pi V_0^\pi(s_0)$.

A POMDP adds $\langle \mathcal{O}, p_O \rangle$ to the MDP tuple. The observation space is $\mathcal{O} \subseteq \mathbb{R}^{n_o}$, and the observation model $p_O(o|s)$ is the conditional probability of receiving an observation $o \in \mathcal{O}$ at $s \in \mathcal{S}$. Similarly to Sunberg and Kochenderfer [2018], we assume that we can both sample from and evaluate $p_O(o|s)$.

A history at time t is defined as a sequence of b_0, followed by actions taken and observations received until t: $H_t \triangleq (b_0, a_0, o_1, \ldots, a_{t-1}, o_t)$. In partially observable settings, the agent has to maintain a probability distribution of the current state given past actions and observations, known as the belief. The belief at time t is $b_t(s_t) \triangleq p(s_t|H_t)$. We define $H_t^- \triangleq (b_0, a_0, o_1, \ldots, a_{t-1})$, i.e. the history until time t without the last measurement, and correspondingly the propagated belief $b_t^-(s_t) \triangleq p(s_t|H_t^-)$. It has been shown that optimal decision-making can be made given the belief, instead of considering the entire history [Kaelbling *et al.*, 1998], meaning a POMDP is an MDP on the belief space [Åström, 1965]. Policies and value functions extend from MDPs to POMDPs by equivalent definitions on beliefs as states.

Importance Sampling. Importance Sampling (IS) [Kloek and Van Dijk, 1978] is a technique in Monte-Carlo (MC) estimation where a proposal distribution q is used to generate samples. The IS estimator for $g(x) = \mathbb{E}_{x\sim p}[f(x)]$ is $\hat{g}_q = N^{-1} \sum_{i=1}^{N} \rho_q^p(x^i) f(x^i)$, for importance ratios $\rho_q^p(x) \triangleq p(x)/q(x)$. The IS estimator $\hat{g}_q$ is unbiased if $q(x) = 0$ implies $p(x) = 0$. Often, the self-normalized IS (SNIS) estimator is formulated when we only have access to an unnormalized version of p, or for variance reduction purposes, and is given by $\tilde{g}_q = (\sum_{i=1}^{N} \rho_q^p(x^i))^{-1} \sum_{i=1}^{N} \rho_q^p(x^i) f(x^i)$. While biased, under weak assumptions it is consistent [Owen, 2013, 9.2]. We denote SNIS estimators throughout with $\tilde{[\cdot]}$.

MIS [Veach and Guibas, 1995] is a method to leverage several sampling methods $\{q_i\}_{i=1}^n$, by calculating a weighted average of IS estimators with n_i samples each: The corresponding MIS estimator is $\hat{g}_{\text{MIS}} = \sum_{i=1}^{n} n_i^{-1} \sum_{j=1}^{n_i} w_i(x^{i,j}) \rho_{q_i}^p(x^{i,j}) f(x^{i,j})$. If the weights w_i satisfy requirements: (MIS-I) $\sum_{i=1}^{n} w_i(x) = 1$ whenever $f(x) \neq 0$; (MIS-II) $w_i(x) = 0$ whenever $q_i(x) = 0$; then $\hat{g}_{\text{MIS}}$ is unbiased. Many weighting strategies exist; the balance heuristic $w_i(x) = n_i q_i(x) / \sum_k n_k q_k(x)$, has its variance provably bounded from the optimal weighting strategy for independent samples. Analogously to SNIS, self-normalized MIS (SN-MIS) can be defined as $\tilde{g}_{\text{MIS}} = \beta^{-1} \sum_{i=1}^{n} \sum_{j=1}^{n_i} w_i(x^{i,j}) \rho_{q_i}^p(x^{i,j}) f(x^{i,j})$ for $\beta = \sum_{i=1}^{n} \sum_{j=1}^{n_i} w_i(x^{i,j}) \rho_{q_i}^p(x^{i,j})$ [Metelli *et al.*, 2020].

Particle Beliefs. Particle filters are often used to represent a belief non-parametrically [Doucet *et al.*, 2001, 1.3.2]. We denote particle beliefs with $\bar{b}$. The particle belief of J particles is the ordered set of state-weight pairs $\bar{b} = ((s^j, \lambda^j))_{j=1}^J$ (following [Lim *et al.*, 2023]), and is defined as the discrete distribution $p(s|\bar{b}) = \sum_{j=1}^{J} \lambda^j \cdot \delta(s - s^j) / \sum_{j=1}^{J} \lambda^j$. For computational simplicity, we assume throughout the paper that the bootstrap filter is used to update particle beliefs [Doucet *et al.*, 2001, 1.3.2]. At each time step, the bootstrap filter advances sampled particles using the transition model, reweights them based on the observation model, and resamples to avoid weight degeneracy [Kong *et al.*, 1994].

MCTS and DPW. MCTS is an algorithm used to quickly explore large state spaces [Browne *et al.*, 2012]. It iteratively repeats four steps to build a search tree that approximates the action-values, using a best-first strategy: (1) *Selection*: Starting from the root node, descend recursively until a node to which children can be added is found; (2) *Expansion*: A new node is added as a child, according to an action expansion strategy; (3) *Simulation*: A simulation is run from the new node according to the rollout (default) policy; (4) *Backpropagation*: The simulation result is backpropagated through the selected nodes to update the action-values. Often UCT [Koc-

sis and Szepesvári, 2006] is used at *selection* to balance between exploration of new actions and exploitation of promising ones. Double Progressive Widening (DPW) [Couëtoux *et al.*, 2011] is a technique to limit the branching factor from being infinite in continuous settings. The number of children of a node is artificially limited to kN^α for N visitations, for fixed $k > 0$ and $0 < \alpha < 1$.

3 Action Gradients and MIS Trees

Here we present the theoretical basis of AGMCTS. We derive action score gradients for local refinement, introduce MIS Trees for consistent value estimates, and discuss density computation for smooth black-box simulators.

3.1 (PO)MDP Action Score Gradients

In POMDPs the posterior belief transition probability $p(b'|b,a)$ is generally intractable. We approach the belief update structure via the propagated belief density:

$$p(b'|b,a) = \int p(b'|b'^-,o')p(o'|b'^-)p(b'^-|b,a)\,\mathrm{d}b'^-\,\mathrm{d}o'. \quad (1)$$

The propagated belief b'^- is obtained after action a but before observation o'. Exact beliefs make $p(b'^-|b,a)$ a Dirac delta because b'^- is deterministic in (b,a); for particle beliefs $\bar{b}, \bar{b}'^-$ it is non-degenerate because particles are sampled through stochastic transitions:

Lemma 1. *Let $\bar{b} = ((s^j,\lambda^j))_{j=1}^J$, $\bar{b}'^- = ((s'^{-,j},\lambda'^{-,j}))_{j=1}^J$ be ordered particle beliefs. The probability that $\bar{b}'^-$ is a propagated belief in the bootstrap filter, given $\bar{b}$ and action a, is:*

$$p(\bar{b}'^-|\bar{b},a) = \textstyle\prod_{j=1}^J p_T(s'^{-,j}|s^j,a), \quad (2)$$

whenever $\lambda'^{-,j} = \lambda^j$ for all $j = 1,\dots,J$, and zero otherwise. When $p(\bar{b}'^-|\bar{b},a) > 0$ it holds that:

$$\nabla_a \log p(\bar{b}'^-|\bar{b},a) = \textstyle\sum_{j=1}^J \nabla_a \log p_T(s'^{-,j}|s^j,a). \quad (3)$$

We use this to introduce importance ratios over the future propagated belief $\bar{b}'^-$ rather than the posterior belief $\bar{b}'$.

In tree search, after updating action a to $\breve{a}$, we estimate $Q_t^\pi(s,\breve{a})$ by reusing samples from $Q_t^\pi(s,a)$ instead of recomputing the subtree. Because repeated updates create samples from multiple proposals, we use IS/MIS. Equations (4)-(5) give the single-proposal action-reuse form:

$$Q_t^\pi(s,\breve{a}) = \mathbb{E}_{s'|q}[\rho_q^{p_T}(s')(r(s,\breve{a},s') + \gamma V_{t+1}^\pi(s'))], \quad (4)$$

$$Q_t^\pi(\bar{b},\breve{a}) = \mathbb{E}_{\bar{b}'^-,o',\bar{b}'|q}[\rho_q^p(\bar{b}'^-)(r(\bar{b},\breve{a},\bar{b}') + \gamma V_{t+1}^\pi(\bar{b}'))], \quad (5)$$

where $\rho_q^{p_T}(s') \triangleq p_T(s'|s,\breve{a})/q(s'|s,\breve{a})$, and $\rho_q^p(\bar{b}'^-) \triangleq p(\bar{b}'^-|\bar{b},\breve{a})/q(\bar{b}'^-|\bar{b},\breve{a})$ are the importance ratios. The ratios require proposal support to cover target support. We use $q(s'|s,\breve{a}) := p_T(s'|s,a)$, reusing samples from the previous action a, but generally other choices are possible too. In Section 3.2, we extend Equations (4)-(5) to an MIS setting with multiple proposals, where assumptions (MIS-I) and (MIS-II) suffice for unbiasedness (Section 2).

We introduce our main results of action score gradients.

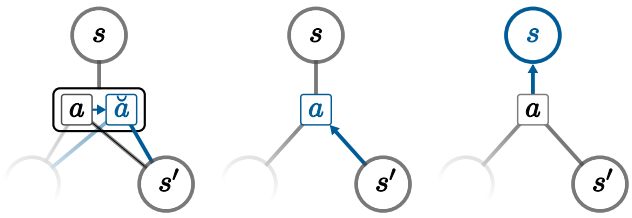

Figure 1: Update operations in the MIS Tree, left to right, highlighted in cyan: (1) *Action update*, a novel operation, computes a consistent action-value estimate for a new action $\breve{a}$ reusing previous successor states; (2) *Action backpropagation*; (3) *State backpropagation*.

Theorem 1. *Assume: (i) $\rho_q^{p_T}(s')$, $\rho_q^p(\bar{b}'^-)$ are well-defined; (ii) r, V are bounded; (iii) there exist integrable g_r, g_T w.r.t. the MDP/POMDP measures such that $\|\nabla_a r\| \le g_r$ and $\|\nabla_a \log p_T\| \le g_T$. For POMDPs, assume the assumptions of Lemma 1 hold. Then, the action score gradient satisfies:*

$$\nabla_{\breve{a}} Q_t^\pi(s,\breve{a}) = \mathbb{E}_{s'|q}[\rho_q^{p_T}(s')[\nabla_{\breve{a}} \log p_T(s'|s,\breve{a}) \\ (r(s,\breve{a},s') + \gamma V_{t+1}^\pi(s') - B(s)) + \nabla_{\breve{a}} r(s,\breve{a},s')]], \quad (6)$$

$$\nabla_{\breve{a}} Q_t^\pi(\bar{b},\breve{a}) = \mathbb{E}_{\bar{b}'^-,o',\bar{b}'|q}[\rho_q^p(\bar{b}'^-)[\nabla_{\breve{a}} \log p(\bar{b}'^-|\bar{b},\breve{a}) \\ (r(\bar{b},\breve{a},\bar{b}') + \gamma V_{t+1}^\pi(\bar{b}') - B(\bar{b})) + \nabla_{\breve{a}} r(\bar{b},\breve{a},\bar{b}')]], \quad (7)$$

where $B(s)$ (resp. $B(\bar{b})$) is any baseline independent of $\breve{a}$ for estimator variance reduction [Schulman et al.*, 2016], e.g. $B(s) = V_t^\pi(s)$.*

The proof follows policy-gradient arguments [Sutton *et al.*, 1999]. We give all proofs in the Appendix [Lev-Yehudi *et al.*, 2026]. The result allows computing action gradients with samples taken from the proposal q. The gradient favors immediate reward and likely high-value successors; since s' is integrated out, $V_{t+1}^\pi(s')$ has no direct $\breve{a}$-gradient. Compared with recursive estimates like VG-UCT [Lee *et al.*, 2020], these score gradients are branch-local in the MIS Tree, and apply to belief-dependent policies via Equation (1).

3.2 MIS Trees for Action-Adaptive MCTS

We introduce the MIS Tree, a novel search-tree whose action-value estimates are MIS estimators. It supports standard backpropagation plus *action update*: replacing an action label while reusing previous successor states for a consistent updated action-value estimate. Figure 1 illustrates these operations. We formulate the MIS Tree recursively over state nodes s and action nodes (s,a). For POMDPs, we use a particle-filter tree (PFT) [Sunberg and Kochenderfer, 2018], replacing s by ordered particle belief states $\bar{b}$ throughout.

Each state node s has a set of child actions $\mathcal{A}_s \triangleq \{a^i\}_i$. Each action node (s,a) has a set of child successor states $\mathcal{S}_{sa} \triangleq \{s'^{,i}\}_i$. The visitation counts of state and action nodes are $n(s)$ and $n(s,a)$. We store for each successor $s'^{,i}$ the proposal action a^i_{prop}, namely the action under which $s'^{,i}$ was originally sampled. This stored proposal action may differ from the current action label a after action updates. We also store the value estimate $\hat{V}(s'^{,i})$, defined by:

$$\hat{V}(s) \triangleq n(s)^{-1} \textstyle\sum_{i=1}^{|\mathcal{A}_s|} n(s,a^i)\hat{Q}(s,a^i). \quad (8)$$

Algorithm 1 AGMCTS. Highlighted lines mark AGMCTS-specific MIS and action-optimization steps.

procedure SIMULATE(s, d)
if $d = 0$ **then**
  $v \leftarrow$ ROLLOUT(s) {*Rollout until terminal state*}
  UPDATETERMINAL(s, v) {*Eq.* (17)}
  return v
$a \leftarrow$ ACTIONPROGWIDEN(s) {*Vanilla or VPW*}
$addBranch \leftarrow$ ACTIONOPT (s, a, d)
if $|\mathcal{S}_{sa}| \leq k_o \cdot n(s,a)^{\alpha_o}$ OR $addBranch$ **then**
  $s', r \sim G(s, a)$ {$\bar{b}'^{-}, \bar{b}', r \sim G(\bar{b}, a)$ *in POMDPs*}
  $\mathcal{S}_{sa} \leftarrow \mathcal{S}_{sa} \cup \{s'\}$
  $v \leftarrow$ ROLLOUT$(s', d-1)$
else
  $s' \sim$ Unif$(\mathcal{S}_{sa})$, $r \leftarrow$ REWARD(s, a, s')
  $v \leftarrow$ SIMULATE$(s', d-1)$
  UPDATEMIS(s, a, s', r) {*Eqs.* (15), (16), (18)}

procedure ACTIONOPT(s, a, d)
$addBranch \leftarrow$ FALSE
for all $k = 1, \ldots, K_{\text{opt}}$ **do**
  $g_a^q \leftarrow \hat{\nabla}_a \tilde{Q}(s, a)$ {*Eq.* (21)}
  $\breve{a} \leftarrow$ OPT(s, a, g_a^q) {*Adam/other*}
  $\breve{a} \leftarrow$CLIPNORM$(\breve{a}, T_{d_a}^{\max})$
  ACTIONUPDATE$(s, a, \breve{a})$ {*Eq.* (11)-(14), (18)}
  $addBranch \leftarrow addBranch \vee (\forall s'^{,i} \in \mathcal{S}_{s\breve{a}} : \rho_{\breve{a}}^i(s'^{,i}) \leq T_\rho^{add})$
return $addBranch$

The visitation counts satisfy:

$$n(s) = \sum_{i=1}^{|\mathcal{A}_s|} n(s, a^i), \quad n(s,a) = \sum_{i=1}^{|\mathcal{S}_{sa}|} n(s'^{,i})_{+1}. \quad (9)$$

Here, $n(s')_{+1} \triangleq n(s') + 1$ is the post-expansion count contribution of successor s'. The $+1$ accounts for the initial rollout value stored when the successor node is created, while $n(s')$ counts later visits. For action nodes, $\hat{Q}(s,a) \triangleq \hat{r}(s,a) + \gamma \hat{V}_f(s,a)$, where $\hat{r}$ estimates immediate reward and $\hat{V}_f$ estimates next-step value. Our key observation is that this decomposition lets *action update* recompute $\hat{V}_f$ from older successor samples and their stored value estimates.

Generally in MIS, each proposal distribution q_i has n_i samples. In our situation, we assume that each $s'^{,i} \in \mathcal{S}_{sa}$ was sampled from $p_T(\cdot|s, a^i_{\text{prop}})$, with its associated single value estimate $V(s'^{,i})$. Therefore, we adapt the MIS definition to combine successor-specific estimators, based on the associated proposal action, branch importance ratio $\rho_a^i(s')$ and MIS weights $w_i(s')$:

$$\hat{V}_f(s,a) \triangleq \sum_{i=1}^{|\mathcal{S}_{sa}|} w_i(s'^{,i}) \rho_a^i(s'^{,i}) \hat{V}(s'^{,i}), \quad (10)$$

$$\rho_a^i(s') \triangleq p_T(s'|s,a) / p_T(s'|s, a^i_{\text{prop}}), \quad (11)$$

and for POMDPs $\rho_a^i(\bar{b}') \triangleq p(\bar{b}'^{-}|\bar{b}, a) / p(\bar{b}'^{-}|\bar{b}, a^i_{\text{prop}})$. Analogously, replace $\hat{V}(s'^{,i})$ with $r(s,a,s'^{,i})$ to obtain $\hat{r}(s,a)$.

The following theorem shows that the MIS conditions suffice for unbiasedness of the MIS Tree:

Theorem 2. *If $w_i(s')$ satisfy requirements (MIS-I) and (MIS-II) (see Section 2), then the MIS Tree yields unbiased value and action-value estimates for a given tree structure.*

The MIS balance heuristic requires $O(|\mathcal{S}_{sa}|)$ density evaluations per recursive sample update [Novitsky *et al.*, 2025]. For efficiency, we use self-normalized MIS (SN-MIS) [Metelli *et al.*, 2020] with naive weights $w_i(s') \propto n(s'^{,i})_{+1}$, which we found to perform well in practice:

$$\tilde{V}_f(s,a) \triangleq \sum_{i=1}^{|\mathcal{S}_{sa}|} n(s'^{,i})_{+1} \rho_a^i(s'^{,i}) \hat{V}(s'^{,i}) / \eta(s,a), \quad (12)$$

$$\eta(s,a) \triangleq \sum_{i=1}^{|\mathcal{S}_{sa}|} n(s'^{,i})_{+1} \rho_a^i(s'^{,i}). \quad (13)$$

Using SN-MIS, all MCTS operations are $O(1)$ except $O(|\mathcal{S}_{sa}|)$ *action update*. We give full MIS Tree derivations, numerically stable log-likelihood forms, and analogous immediate-reward updates in the Appendix.

Action Update. Action update replaces (s,a) by $(s,\breve{a})$, recomputes all ratios, (12), (13), and immediate rewards in $O(|\mathcal{S}_{sa}|)$ time.

In POMDPs, the explicit computation of the new transition likelihood $p(\bar{b}'^{-}|\bar{b}, \breve{a})$ via (2) requires J evaluations of the transition density, for J particles in $\bar{b}$ and $\bar{b}'^{-}$, which may be computationally expensive. As $p(\bar{b}'^{-}|\bar{b}, \breve{a})$ is a product of probabilities, a direct MC estimate based on a subset of particles results in a large bias. We resort to estimating changes in $\log p(\bar{b}'^{-}|\bar{b}, \breve{a})$. For small $\|\breve{a} - a\|$ we approximate[1]

$$\log \rho^i_{\breve{a}}(\bar{b}'^{-}) - \log \rho^i_a(\bar{b}'^{-}) \approx (\nabla_a \log \rho^i_a(\bar{b}'^{-}))^T \delta a$$
$$= (\nabla_a \log p(\bar{b}'^{-}|\bar{b}, a))^T \delta a = (\text{Eq. (3)})^T \delta a, \quad (14)$$

and due to Equation (3) being a sum, it admits an unbiased MC estimate by subsampling particles.

Action Backpropagation. Let $s'^{,i}$ be an updated node. Hence, we updated $n(s'^{,i})$ to $n'(s'^{,i})$ and $\hat{V}(s'^{,i})$ to $\hat{V}'(s'^{,i})$. We update $n(s,a)$ by (9) and perform

$$\eta'(s,a) = \eta(s,a) + \rho_a^i(s'^{,i})(n'(s'^{,i}) - n(s'^{,i})), \quad (15)$$

$$\tilde{V}'_f(s,a) = (\eta'(s,a))^{-1} \big(\eta(s,a)\tilde{V}_f(s,a) + \rho_a^i(s'^{,i})(n'(s'^{,i})\hat{V}'(s'^{,i}) - n(s'^{,i})\hat{V}(s'^{,i}))\big). \quad (16)$$

Assuming a general $n'(s')$ also supports cases where we delete branches in the tree, i.e. $n'(s') < 1$. For *state expansion*, the same equations apply by initializing $n(s') = 0$.

State Backpropagation. Let s be a state node. If s is at the maximum tree depth, $\hat{V}(s)$ is based only on rollouts. We update the running average with the new rollout value v',

$$\hat{V}'(s) = \hat{V}(s) + (n'(s) - n(s))(v' - \hat{V}(s))/n'(s)_{+1}, \quad (17)$$

If s is not a terminal state, let its recently updated action-value child be (s,a), with updated $n'(s,a)$ and $\tilde{Q}'(s,a)$. We update $n'(s)$ by (9) and perform

$$\hat{V}'(s) = (n'(s))^{-1}(n(s)\hat{V}(s) + n'(s,a)\tilde{Q}'(s,a) - n(s,a)\tilde{Q}(s,a)). \quad (18)$$

[1]This is obtained by the first-order approximation $f(x + \delta x) \approx f(x) + \nabla f(x)^T \delta x$, applied to $\log \rho^i_a(s')$ where $\delta a = \breve{a} - a$.

3.3 Densities of Smooth Generative Models via the Area Formula

Thus far we have assumed computable $p_T(s'|s,a)$ and $\nabla_a \log p_T(s'|s,a)$. In practice, many simulators provide instead $s' = f(s,a,\xi)$ with noise $\xi \sim p_\xi(\cdot|s,a) \in \mathbb{R}^{n_\xi}$. If $n_\xi = n_s$ and $\xi \mapsto f(s,a,\xi)$ is bijective and differentiable, the change-of-variables formula states that for $s' = f(s,a,\xi^*)$:

$$p_T(s'|s,a) = p_\xi(\xi^*|s,a)\,|D_\xi f(s,a,\xi^*)|^{-1}, \tag{19}$$

where $D_\xi f$ is the Jacobian matrix of f w.r.t. ξ. However, it is often the case that $n_\xi < n_s$ and $\xi \mapsto f(s,a,\xi)$ is an embedding into an n_ξ-dimensional manifold rather than a bijection. Then $p_T(s'|s,a)$ is a density with respect to the induced manifold measure, not Lebesgue measure, and the Area Formula [Federer, 1969, Theorem 3.2.3] applies. Negro [2022] gives the following locally Lipschitz form for f and the induced density p_T:

Theorem 4.1. *[Negro, 2022] If f is locally Lipschitz, and it holds that $\operatorname{rank}(D_\xi f) = n_\xi$ a.e., then the induced probability measure over s' is absolutely continuous w.r.t. the Hausdorff measure $\mathcal{H}^{n_\xi}$ on $\mathbb{R}^{n_s}$, and its Radon-Nikodym derivative is*

$$p_T(s'|s,a) = \textstyle\sum_{\Xi^*} p_\xi(\xi^*|s,a)(J_{n_\xi} f(s,a,\xi^*))^{-1}, \tag{20}$$

for $\Xi^* = \{\xi^* : s' = f(s,a,\xi^*)\}$, and 0 when $\Xi^* = \emptyset$. The n_ξ-dimensional Jacobian is given by the Cauchy-Binet formula: $J_{n_\xi} f(s,a,\xi) = \sqrt{\det((D_\xi f)^\intercal \cdot (D_\xi f))}$.

We further discuss these assumptions in the Appendix.

Robotics simulators often integrate continuous-time dynamics with stochasticity injected at the input or output. In both cases below, once p_T is computable, $\nabla_a \log p_T$ follows by chain rule and automatic differentiation.

Input Noise to Simulator. Let the input noise be described by the function $h(s,a,\xi) = (\tilde{s},\tilde{a})$. The simulator then produces a successor state via $s' = f(\tilde{s},\tilde{a})$. If for a new action $\breve{a}$ there exists a unique ξ^* satisfying $h(s,\breve{a},\xi^*) = (\tilde{s},\tilde{a})$, i.e. such that the input to the simulator remains unchanged, then we can evaluate the sensitivity of the output state with respect to the noise: $D_\xi f(s,\breve{a},\xi^*) = D_{(\tilde{s},\tilde{a})} f \cdot \frac{\partial h}{\partial \xi}|_{s,\breve{a},\xi^*}$. Moreover, the Jacobian matrix $D_{(\tilde{s},\tilde{a})} f$ can be cached from the original sample (s,a,ξ,s'), allowing to compute the density without requiring additional simulator queries.

Noise in Simulator Output. Let the simulator output be $h(s,a)$, and the next state be given by $s' = g(h(s,a),\xi)$. If we can find all ξ^* such that $s' = g(h(s,\breve{a}),\xi^*)$, then we compute the Jacobian via $D_\xi f(s,\breve{a},\xi^*) = \frac{\partial g}{\partial \xi}|_{h(s,\breve{a}),\xi^*}$, requiring a single forward simulation $h(s,\breve{a})$ for all ξ^*.

4 AGMCTS Algorithm

AGMCTS combines MCTS with frequent action-gradient optimization, and correcting online value estimates via MIS. It uses a forward simulator $G(s,a)$ with assumed access to p_T and p_O, controls branching via DPW, and extends to POMDPs via ordered particle-belief states like PFT-DPW [Sunberg and Kochenderfer, 2018]. Algorithm 1 keeps the standard simulation loop; highlighted components add ACTIONOPT and MIS *action update* before expansion so gradients influence expansion. We detail the implementation in the Appendix.

Monte Carlo Gradient Estimation. In POMDPs, we estimate (3) with K_b^∇ state particles via $\nabla_a \log p(\bar{b}'^-|\bar{b},a) \approx (J/K_b^\nabla)\sum_{l=1}^{K_b^\nabla} \nabla_a \log p_T(s^{-,j_l}|s^{j_l},a)$, where indices j_l are sampled uniformly from $[1,\dots,J]$, and J is the particle count of $\bar{b}$ and $\bar{b}'^-$. We choose a baseline function of the current value estimate $B(s) = \hat{V}(s)$ ($B(\bar{b}) = \hat{V}(\bar{b})$ in POMDPs), effectively estimating advantage gradients. As we linearize the weight updates as in Equation (14), $\nabla_a \rho_a^i(s'^{,i})$ has to be computed for all $s'^{,i} \in \mathcal{S}_{sa}$. To save computations, we sum over all successor states $s' \in \mathcal{S}_{sa}$ when computing $\hat{\nabla}\tilde{Q}(s,a)$:

$$\hat{\nabla}_a \tilde{Q}(s,a) = \textstyle\sum_{i=1}^{|\mathcal{S}_{sa}|} n(s'^{,i})_{+1}\rho_a^i(s'^{,i}) \cdot (\nabla_a \log p_T(s'^{,i}|s,a) \cdot (r(s,a,s'^{,i}) + \gamma \hat{V}(s'^{,i}) - B(s)) + \nabla_a r(s,a,s'^{,i}))/\eta(s,a), \tag{21}$$

and we cache $\nabla_a \log p_T(s'^{,i}|s,a)$ for later *action updates*.

Gradient Optimization. We used Adam [Kingma and Ba, 2015] for its step-size normalization, which is crucial for high-variance gradients. We perform K_{opt} consecutive gradient updates to control the accuracy-compute trade-off, and additional clipping of $\|\breve{a} - a\|$ to $T_{d_a}^{\max}$ to stabilize updates.

Threshold for Adding Successor Nodes. If $\rho_{\breve{a}}^i(s'^{,i}) < T_\rho^{add}$ for all $s'^{,i} \in \mathcal{S}_{s\breve{a}}$, we force a new successor sample, with optional low-relevance deletion.

Action Sampling Heuristics. We use uniform action sampling and Voronoi progressive widening (VPW) [Lim *et al.*, 2021]; VPW biases samples through Voronoi partitions, demonstrating that AGMCTS can benefit from different action proposal mechanisms.

5 Experiments

We evaluate AGMCTS on continuous MDP/POMDP benchmarks. MDP baselines are DPW/VPW; POMDP baselines are PFT-DPW/PFT-VPW and POMCPOW/VOMCPOW from POMDPs.jl [Egorov *et al.*, 2017]. AGMCTS also uses POMDPs.jl. VPW and result analysis follow the codebase of [Lim *et al.*, 2021]; we give additional hyperparameters, result tables, and implementation details in the Appendix.

5.1 Benchmark Domains

***d*D-Continuous Light-Dark POMDP.** This domain generalizes the Light-Dark POMDP [Platt *et al.*, 2010] to arbitrary dimension d and combines a narrow goal-centered reward peak, a surrounding moat penalty, a non-Gaussian initial belief, and observation noise that changes sharply across the state space. Here $\mathcal{S} = \mathcal{O} = \mathbb{R}^d$, $b_0 = S_{0.5}^{d-1}$, goal $s_g = (\mathbf{0}_{d-1}, 2.5)$, beacon $s_b = (2.5, \mathbf{0}_{d-1})$, $\gamma = 0.99$, terminal radius $T_g = 0.2$, and horizon $L = 6$. Actions lie in $B_{1.5}(\mathbf{0}_d)$, transitions are $s' \sim \mathcal{N}(s+a, \sigma_T^2\mathbf{I})$ with $\sigma_T = 0.025$, and observations are relative beacon measurements $o \sim \mathcal{N}(s - s_b, \sigma_O(\|s-s_b\|)^2\mathbf{I})$ with $\sigma_O(x) = \min\{15, 0.01(x+x^8)\}$. The reward depends only on the successor state and combines three terms: a sharp positive bump

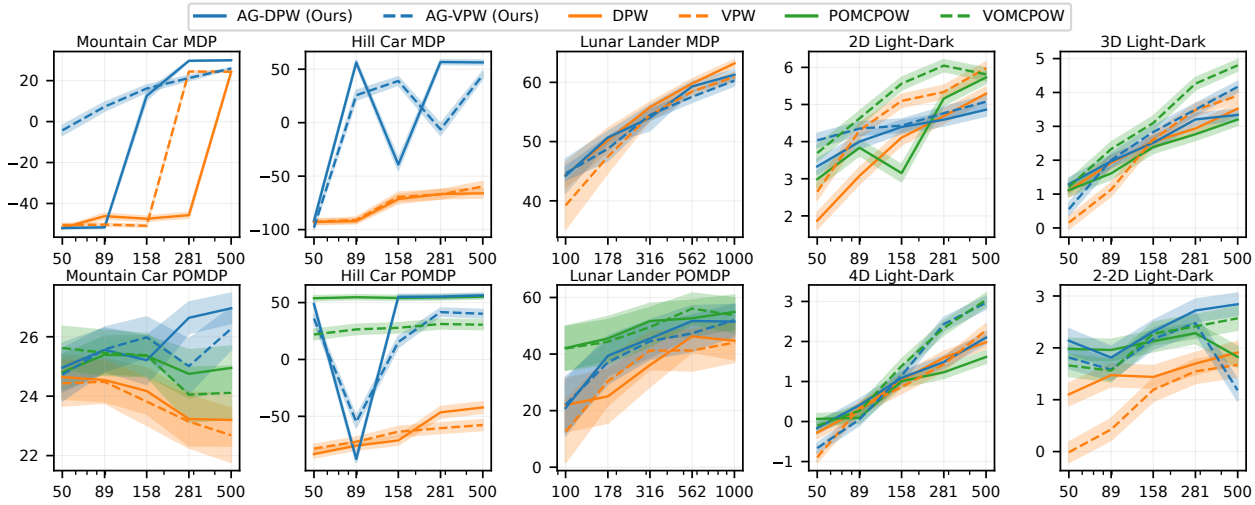

Figure 2: Average returns over 1000 seeds, shaded areas show ± 2 standard error. The x-axis shows the simulations budget per planning session. Ours: AGMCTS algorithms Action-Gradient DPW (AG-DPW) for MDPs, and its particle-filter tree (PFT) adaptation for POMDPs. Orange: DPW for MDPs and PFT-DPW for POMDPs. Green: POMCPOW (POMDP domains only). We test both vanilla action sampling (solid lines) and VPW (dashed lines), resulting in 4 MDP and 6 POMDP algorithms.

at the goal, a surrounding moat penalty that discourages random wandering near the target, and a small quadratic distance penalty. With $\Delta_g = \|s' - s_g\|$, the constants are $R_{\text{goal}} = 10$, $R_{\text{moat}} = 2$, and $R_{\text{dist}} = 0.02$, giving $R_{\text{goal}} e^{-\frac{1}{2}(\Delta_g/(0.5T_g))^2} - R_{\text{moat}} e^{-\frac{1}{2}((\Delta_g - 5T_g)/T_g)^2} - R_{\text{dist}}\Delta_g^2$. Rollouts move toward s_g but add $\mathcal{N}(\mathbf{0}, 0.1^2\mathbf{I})$ action noise, providing a noisy goal-directed default policy. We benchmark $d = 2, 3, 4$.

Two-Agent dD-Continuous Light-Dark POMDP. This variant uses $\mathcal{S} = \mathcal{O} = \mathbb{R}^{2d}$, and $\mathcal{A} = B_{1.5}(\mathbf{0}_d) \times B_{1.5}(\mathbf{0}_d)$. Each agent has the same transition and individual reward structure, with the positive goal term obtained only when both agents are within T_g. The observation model couples the agents: agent 1 observes its relative beacon position, while agent 2 observes its relative position to agent 1 with the same distance-dependent covariance. Thus agent 2 receives position information through agent 1 rather than directly through the fixed beacon, while the reward grants success jointly. We benchmark $d = 2$.

Mountain Car and Hill Car MDP/POMDP. These domains are based on Mountain Car [Singh and Sutton, 1996] and Car on the Hill [Ernst *et al.*, 2005]; $\mathcal{S} = \mathbb{R}^2$ is position and velocity, and $\mathcal{A} = [-a_{\max}, a_{\max}]$ accelerates/brakes the car while avoiding the left boundary. In POMDPs, velocity is unobserved and $z \sim \mathcal{N}(x, 0.03^2)$. These domains have momentum dynamics and threshold termination: the car may move away from the goal to gain speed, while overshooting the safe speed range causes termination. The reward is 100 at $x \geq x_{\max}$, -100 at $x < x_{\min}$ or $|v| \geq v_{\max}$, and -0.1 otherwise; $\gamma = 0.99$. Rollouts use the standard momentum heuristic, pushing with $a_{\max}$ when $v > 0$ and $-a_{\max}$ otherwise, which provides a useful but myopic default. Our modified Mountain Car instance uses $x_{\min} = -1.5$, $x_{\max} = 0.5$, $v_{\max} = 0.05$, $a_{\max} = 1.0$, horizon $L = 200$, $v' = v + 0.001\tilde{a} - 0.0025\cos(3x)$, and $x' = x + v'$. Hill Car uses horizon $L = 30$ and integrates nonlinear continuous-time dynamics for $\Delta t = 0.1$, with the same noisy-clamped action interface used for AGMCTS density evaluation.

The two car domains differ in where complexity appears. Mountain Car has a long horizon under a tight velocity bound, so momentum-building policies can terminate by exceeding the speed limit. Hill Car has a shorter horizon, but each transition is obtained by integrating nonlinear continuous-time dynamics. In both domains, additive action-input noise $\xi \sim \mathcal{N}(0, 0.1^2)$ is clamped as $\tilde{a} = \text{clip}(a + \xi, -a_{\max}, a_{\max})$ before simulation, yielding transition distributions with continuous interior components and discrete clipping masses.

Lunar Lander MDP/POMDP. We use Lunar Lander as in [Mern *et al.*, 2021; Lim *et al.*, 2021], with state $(x, y, \theta, \dot{x}, \dot{y}, \omega)$ and action $(F_x, T, \delta) \in [0, 15] \times [-5, 5] \times [-1, 1]$. Actions couple translation, orientation, and angular rate through the thrust direction and torque, and rewards combine threshold penalties with a landing reward. Here T is main thrust, F_x corrective thrust, and δ the offset producing torque $\tau = -\delta F_x$. With $m = 1$, $I = 10$, and $\Delta t = 0.4$, the deterministic model computes $f_x = \cos\theta F_x - \sin\theta T$, $f_y = \cos\theta T + \sin\theta F_x$, $\dot{x}' = \dot{x} + f_x\Delta t$, $\dot{y}' = \dot{y} + (f_y - 9)\Delta t$, $\omega' = \omega + \tau\Delta t/I$, $x' = x + \dot{x}\Delta t$, $y' = y + \dot{y}\Delta t$, and $\theta' = \theta + \omega\Delta t$. Gaussian output noise with scales $(0.1, 0.1, 0.01)$ is added to $\dot{x}'$, $\dot{y}'$, and ω', so transition densities are computed by solving for the output-noise variables under the Area Formula. In the POMDP variant, $o = (\tilde{\omega}, \tilde{\dot{x}}, \tilde{h})$ with noise scales $(1.0, 0.01, 0.1)$ and $\tilde{h}$ centered at $y/\cos\theta$. Rewards are -1000 if $x \geq 15$ or $\theta \geq 0.5$, $100 - x - \dot{y}^2$ if $y \leq 1$, and -1 otherwise. We use $\gamma = 0.99$ and horizon $L = 35$. The state distribution is a Gaussian with mean $(0, 50, 0, 0, -10, 0)$, and the rollout is a proportional stabilizer $(-0.1\dot{x}, -0.1\dot{y}, 0)$ under noisy partial observations.

Domain	Setting	DPW	VPW	AG-DPW	AG-VPW
2D Light-Dark	POMDP	.027	.027	.316	.191
3D Light-Dark	POMDP	.068	.073	.315	.154
4D Light-Dark	POMDP	.151	.155	.302	.318
2-2D Light-Dark	POMDP	.160	.173	.447	.464
Mountain Car	MDP	.016	.016	.148	.152
	POMDP	.042	.044	.928	.904
Hill Car	MDP	.026	.029	.744	1.028
	POMDP	.360	.347	1.636	1.109
Lunar Lander	MDP	.008	.009	.203	.216
	POMDP	.103	.102	.504	.508

POMDP domain	POMCPOW	VOMCPOW
2D Light-Dark	.033	.033
3D Light-Dark	.060	.063
4D Light-Dark	.141	.170
2-2D Light-Dark	.141	.169
Mountain Car	.116	.134
Hill Car	.348	.297
Lunar Lander	.136	.144

Table 1: Mean planning time in seconds at the maximum simulation budget of each domain. In POMDP domains, POMCPOW/VOMCPOW use a larger simulation budget to match runtimes similar to PFT-DPW/PFT-VPW.

5.2 Performance Evaluation

We tune hyperparameters with cross-entropy maximization at the maximum simulation budget. This includes UCT/DPW parameters, and Adam step size for AG variants, while other AG settings are manual. VPW Voronoi parameters use reasonable values from [Lim *et al.*, 2021]. The CE procedure runs 50 iterations; each iteration samples 150 parameter settings, evaluates each on 40 seeds, and refits a smoothed Gaussian proposal from the 30 elite settings. All reported means use 1000 shared seeds. Maximum planning budgets are 500 simulations for Light-Dark and Car domains and 1000 for Lunar Lander. These simulation-budget comparisons intentionally hold the number of model queries fixed, giving a clearer test of whether action-gradient refinement improves the same underlying tree-search procedures rather than conflating this effect with implementation-dependent wall-clock costs. For wall-clock matching, POMCPOW receives a larger state-trajectory budget matched to PFT-DPW runtime.

Transition-density computations match the domain structure: closed-form Gaussian densities in Light-Dark, clipped input-noise inversion in Car domains, and output-noise inversion in Lunar Lander. AG variants use $K_{\text{opt}} \in \{2, 3\}$ Adam updates with linearized importance-weight updates (14).

Figure 2 shows mean performance per planning budget. At maximum budget, an AG variant was preferable to its matched non-AG DPW/VPW-family baseline in 6/10 scenarios (all Hill/Mountain Car, 4D and 2x2D Light-Dark). Against matched baselines (AG-DPW vs. DPW, AG-VPW vs. VPW, and the corresponding PFT variants), AG improved significantly in 10/20 cases, was insignificant in 7/20, and underperformed in 3/20.

By best algorithm per scenario, AGMCTS led in Mountain/Hill Car MDPs, was best or statistically tied in 5/7 POMDPs, and underperformed in 2/7. Overall, it was competitive with the wall-clock-matched POMCPOW/VOMCPOW baselines and often improved over PFT-DPW/PFT-VPW. The largest gains appear in the Mountain/Hill Car domains, while Light-Dark and Lunar Lander show more mixed outcomes. The results suggest that AGMCTS is effective in domains where small action changes can lead to large changes in states and rewards, as in the very long-horizon Mountain Car or very narrow goal in the Light-Dark POMDPs. The performance difference in Lunar Lander is mostly insignificant, which may be due to the combination of large action dimensionality and sparse rewards, reducing gradient signal quality.

On the other hand, AG variants were 10–$35\times$ slower than DPW/VPW in MDPs and 2–$20\times$ slower than PFT-DPW/VPW in POMDPs, as shown in Table 1. AG variants have higher per-simulation cost because action updates revise successor weights and evaluate transition-density terms; these single-threaded timings therefore report implementation cost rather than a wall-clock optimum, since particle-belief updates in AGMCTS and PFT-DPW can be parallelized over particles. Thus, the practical trade-off is whether improved action quality per model query offsets this density and gradient overhead within an application-relevant latency budget.

6 Conclusions

We introduced action-gradient continuous-POMDP tree search with MIS-consistent action-branch updates. Using the Area Formula to obtain transition log-probabilities, AGMCTS improved several continuous MDP/POMDP benchmarks and was competitive overall, showing that online gradient refinement can strengthen MCTS. Future work includes convergence analysis and faster, less tuning-sensitive general planning algorithms.

Practical Considerations. AGMCTS requires transition-density access and tuning in addition to the per-simulation overhead. These costs should be weighed against application latency and modeling constraints.

Acknowledgments

This work was supported by the Israel Ministry of Innovation, Science and Technology.

References

[Åström, 1965] Karl Johan Åström. Optimal control of Markov processes with incomplete state information i. *Journal of mathematical analysis and applications*, 10:174–205, 1965.

[Azizzadenesheli *et al.*, 2018] Kamyar Azizzadenesheli, Yisong Yue, and Animashree Anandkumar. Policy gradient in partially observable environments: Approximation and convergence. *arXiv preprint arXiv:1810.07900*, 2018.

[Baxter and Bartlett, 2001] Jonathan Baxter and Peter L Bartlett. Infinite-horizon policy-gradient estimation. *journal of artificial intelligence research*, 15:319–350, 2001.

[Browne *et al.*, 2012] Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43, 2012.

[Couëtoux *et al.*, 2011] Adrien Couëtoux, Jean-Baptiste Hoock, Nataliya Sokolovska, Olivier Teytaud, and Nicolas Bonnard. Continuous upper confidence trees. In *International conference on learning and intelligent optimization*, pages 433–445. Springer, 2011.

[Doucet *et al.*, 2001] Arnaud Doucet, Nando de Freitas, and Neil Gordon, editors. *Sequential Monte Carlo Methods In Practice*. Springer-Verlag, New York, 2001.

[Egorov *et al.*, 2017] Maxim Egorov, Zachary N Sunberg, Edward Balaban, Tim A Wheeler, Jayesh K Gupta, and Mykel J Kochenderfer. POMDPs.jl: A framework for sequential decision making under uncertainty. *The Journal of Machine Learning Research*, 18(1):831–835, 2017.

[Ernst *et al.*, 2005] Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6, 2005.

[Farhi and Indelman, 2021] Elad I. Farhi and Vadim Indelman. iX-BSP: Incremental belief space planning. *arXiv preprint arXiv:2102.09539*, 2021.

[Federer, 1969] Herbert Federer. *Geometric measure theory*, volume 153 of *Grundlehren Math. Wiss.* Springer, Cham, 1969.

[Hoerger *et al.*, 2024] Marcus Hoerger, Hanna Kurniawati, Dirk Kroese, and Nan Ye. Adaptive discretization using Voronoi trees for continuous POMDPs. *The International Journal of Robotics Research*, 43(9):1283–1298, 2024.

[Hong *et al.*, 2024] Mao Hong, Zhengling Qi, and Yanxun Xu. A policy gradient method for confounded POMDPs. In *The Twelfth International Conference on Learning Representations*, 2024.

[Kaelbling *et al.*, 1998] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1):99–134, 1998.

[Kim *et al.*, 2020] Beomjoon Kim, Kyungjae Lee, Sungbin Lim, Leslie Kaelbling, and Tomás Lozano-Pérez. Monte Carlo tree search in continuous spaces using Voronoi optimistic optimization with regret bounds. In *AAAI Conf. on Artificial Intelligence*, volume 34, pages 9916–9924, 2020.

[Kingma and Ba, 2015] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[Kloek and Van Dijk, 1978] Teun Kloek and Herman K Van Dijk. Bayesian estimates of equation system parameters: an application of integration by Monte Carlo. *Econometrica: Journal of the Econometric Society*, pages 1–19, 1978.

[Kocsis and Szepesvári, 2006] Levente Kocsis and Csaba Szepesvári. Bandit based Monte-Carlo planning. In *European conference on machine learning*, pages 282–293. Springer, 2006.

[Kong *et al.*, 1994] Augustine Kong, Jun S. Liu, and Wing Hung Wong. Sequential imputations and Bayesian missing data problems. *Journal of the American Statistical Association*, 89(425):278–288, 1994.

[Kujanpää *et al.*, 2024] Kalle Kujanpää, Amin Babadi, Yi Zhao, Juho Kannala, Alexander Ilin, and Joni Pajarinen. Continuous Monte Carlo graph search. In *Intl. Conf. on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1047–1056, 2024.

[Kurniawati, 2022] Hanna Kurniawati. Partially observable Markov decision processes and robotics. *Annual Review of Control, Robotics, and Autonomous Systems*, 5(1):253–277, 2022.

[Lauri *et al.*, 2022] Mikko Lauri, David Hsu, and Joni Pajarinen. Partially observable Markov decision processes in robotics: A survey. *IEEE Transactions on Robotics*, 39(1):21–40, 2022.

[Lee *et al.*, 2020] Jongmin Lee, Wonseok Jeon, Geon-Hyeong Kim, and Kee-Eung Kim. Monte-Carlo tree search in continuous action spaces with value gradients. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 4561–4568, 2020.

[Leurent and Maillard, 2020] Edouard Leurent and Odalric-Ambrym Maillard. Monte-Carlo graph search: the value of merging similar states. In Sinno Jialin Pan and Masashi Sugiyama, editors, *Asian Conference on Machine Learning (ACML 2020)*, pages 577 – 592, Bangkok, Thailand, November 18-20 2020.

[Lev-Yehudi *et al.*, 2026] Idan Lev-Yehudi, Michael Novitsky, Moran Barenboim, Ron Benchetrit, and Vadim Indelman. Value gradients with action adaptive search trees in continuous (PO)MDPs. *arXiv preprint arXiv:2503.12181*, 2026.

[Lim *et al.*, 2021] Michael H Lim, Claire J Tomlin, and Zachary N Sunberg. Voronoi progressive widening: efficient online solvers for continuous state, action, and observation POMDPs. In *2021 60th IEEE conference on decision and control (CDC)*, pages 4493–4500. IEEE, 2021.

[Lim *et al.*, 2023] Michael H Lim, Tyler J Becker, Mykel J Kochenderfer, Claire J Tomlin, and Zachary N Sunberg. Optimality guarantees for particle belief approximation of POMDPs. *Journal of Artificial Intelligence Research*, 77:1591–1636, 2023.

[Madani *et al.*, 2003] Omid Madani, Steve Hanks, and Anne Condon. On the undecidability of probabilistic planning and related stochastic optimization problems. *Artificial Intelligence*, 147(1-2):5–34, 2003.

[Mern *et al.*, 2021] John Mern, Anil Yildiz, Zachary Sunberg, Tapan Mukerji, and Mykel J Kochenderfer. Bayesian

optimized Monte Carlo planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11880–11887, 2021.

[Metelli *et al.*, 2020] Alberto Maria Metelli, Matteo Papini, Nico Montali, and Marcello Restelli. Importance sampling techniques for policy optimization. *J. Mach. Learn. Res.*, 21(141):1–75, 2020.

[Morere *et al.*, 2018] Philippe Morere, Roman Marchant, and Fabio Ramos. Continuous state-action-observation POMDPs for trajectory planning with Bayesian optimisation. In *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 8779–8786. IEEE, 2018.

[Negro, 2022] Luigi Negro. Sample distribution theory using coarea formula. *Communications in Statistics-Theory and Methods*, pages 1–26, 2022.

[Novitsky *et al.*, 2025] Michael Novitsky, Moran Barenboim, and Vadim Indelman. Previous knowledge utilization in online anytime belief space planning. *IEEE Robotics and Automation Letters (RA-L)*, 2025.

[Owen, 2013] Art B. Owen. *Monte Carlo theory, methods and examples*. https://artowen.su.domains/mc/, 2013.

[Papadimitriou and Tsitsiklis, 1987] Christos H. Papadimitriou and John N. Tsitsiklis. The complexity of Markov decision processes. *Mathematics of operations research*, 12(3):441–450, 1987.

[Platt *et al.*, 2010] Robert Platt, Jr., Russ Tedrake, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Belief space planning assuming maximum likelihood observations. In *Robotics: Science and Systems (RSS)*, pages 587–593, Zaragoza, Spain, 2010.

[Porta *et al.*, 2006] Josep M. Porta, Nikos Vlassis, Matthijs T.J. Spaan, and Pascal Poupart. Point-based value iteration for continuous POMDPs. *J. of Machine Learning Research*, 7:2329–2367, 2006.

[Schulman *et al.*, 2016] John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *Intl. Conf. on Learning Representations (ICLR)*, 2016.

[Silver and Veness, 2010] David Silver and Joel Veness. Monte-Carlo planning in large POMDPs. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2164–2172, 2010.

[Silver *et al.*, 2014] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, pages 387–395. Pmlr, 2014.

[Singh and Sutton, 1996] Satinder P Singh and Richard S Sutton. Reinforcement learning with replacing eligibility traces. *Machine learning*, 22:123–158, 1996.

[Sunberg and Kochenderfer, 2018] Zachary Sunberg and Mykel Kochenderfer. Online algorithms for POMDPs with continuous state, action, and observation spaces. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 28, 2018.

[Sutton *et al.*, 1999] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

[Veach and Guibas, 1995] Eric Veach and Leonidas J Guibas. Optimally combining sampling techniques for Monte Carlo rendering. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 419–428. ACM, 1995.

[Wu *et al.*, 2021] Chenyang Wu, Guoyu Yang, Zongzhang Zhang, Yang Yu, Dong Li, Wulong Liu, and Jianye Hao. Adaptive online packing-guided search for POMDPs. *Advances in Neural Information Processing Systems*, 34:28419–28430, 2021.

[Ye *et al.*, 2017] Nan Ye, Adhiraj Somani, David Hsu, and Wee Sun Lee. DESPOT: Online POMDP planning with regularization. *JAIR*, 58:231–266, 2017.

[Yee *et al.*, 2016] Timothy Yee, Viliam Lisý, and Michael Bowling. Monte Carlo Tree Search in continuous action spaces with execution uncertainty. In Subbarao Kambhampati, editor, *IJCAI*, pages 690–697, 2016.