ZoomQ: Adaptive Action Discretization for Continuous-Action RL

Taro Iyadomi[1] Larissa Xu[2] Khang Nguyen[3]
Ricardo Parada[4] William Chang[2] Senthil Yogamani[5]

[1]Northwestern University [2]University of California, Los Angeles [3]University of Southern California
[4]University of California, Riverside [5]Qualcomm

Abstract

Value-based reinforcement learning requires a maximization over the action space at every step, which is intractable when that space is continuous; actor-critic methods such as SAC approximate the maximizer with a separate policy network. Another approach is to discretize the action space, but the number of cells per axis N must be fixed before training and is costly to misjudge in either direction. We present ZoomQ, which converts that choice into an upper bound that can be set generously. ZoomQ maintains a hierarchical partition of each action coordinate, refines a cell only once that cell's own visit count crosses a resolution-dependent threshold, and learns values over the resulting cells with a branching deep Q-network. Restricting the partition to the action space, and factoring it across coordinates, keeps the discrete action count additive rather than exponential in the action dimension d_A. On four continuous-control tasks, an over-generous budget costs a uniform grid much of its return (`cheetah-run` falls from 431 to 145 as N grows from 16 to 128) while ZoomQ is largely unaffected (320 at $N = 128$). For a tabular idealization carrying a separate action partition per state, we prove regret with leading term $\tilde{\mathcal{O}}(K^{(z_A+1)/(z_A+2)})$ in the number of episodes K, where an action-only zooming dimension $z_A \leq d_A$ replaces the joint state-action zooming dimension of prior work, together with a sublinear bound on the number of cells instantiated.

1 Introduction

Continuous-action reinforcement learning is dominated in practice by actor-critic methods such as DDPG, TD3, SAC, and PPO, which represent policy and value as separate networks. While powerful, they also carry well-known costs: brittle hyperparameters, entropy or temperature tuning, and instability when the value function is poorly conditioned. A long-standing alternative is to *discretize* the action space and apply value-based machinery to the resulting finite-arm problem. The most well-known approach, uniform discretization at scale ε, converts an MDP with action space $\mathcal{A} \subseteq \mathbb{R}^{d_A}$ into one with $\Theta(\varepsilon^{-d_A})$ arms (Kleinberg et al., 2019; Sinclair et al., 2019), and remains the standard discrete-action baseline in continuous control (Tang and Agrawal, 2020).

Both weaknesses trace to the same decision: the resolution is a single constant, applied everywhere and chosen in advance. Because it is uniform, it spends equal resolution on profitable and unprofitable regions, at a cost that scales with the covering dimension d_A. Because it is chosen in advance, it must be chosen correctly: set too coarse, the policy class cannot express the action the task requires; set too fine, the value function must separate many near-duplicate arms from a fixed interaction budget. The practitioner therefore faces a task-specific tuning problem with a penalty on both sides.

Another approach is *adaptive* discretization (Kleinberg et al., 2019; Bubeck et al., 2011; Sinclair et al., 2019; Cao and Krishnamurthy, 2020), which aims to remove the choice by maintaining a hierarchical partition and subdividing a region only once the statistical error of its value estimate falls

below the variation of the value function across it. Regret then scales with the *zooming dimension* of the instance, a quantity bounded above by the covering dimension and frequently smaller (Kleinberg et al., 2019; Cao and Krishnamurthy, 2020). Two properties of the zooming algorithm of (Cao and Krishnamurthy, 2020) nonetheless obstruct its deployment. First, its partition is defined over the *joint* state-action space $\mathcal{S} \times \mathcal{A}$, so covering that space at resolution r costs on the order of $r^{-(d_S+d_A)}$ cells. The exponent carries the observation dimension: a twenty-dimensional observation already demands more than 10^6 cells at $r = 1/2$, far beyond the interaction budget available to fill them, so the partition stays at whatever resolution it was initialized with. Second, the algorithm is tabular, storing one value per cell with no function approximation, so nothing learned at one state transfers to any other.

We present ZoomQ, which addresses both. The partition is maintained over the action space alone, *factored* across coordinates so that the number of discrete actions grows additively rather than exponentially in d_A, and values are represented by a branching deep Q-network (Tavakoli et al., 2018). We retain the buffering phase of (Cao and Krishnamurthy, 2020): a newly created cell is withheld from the selection rule but receives a scheduled fraction of its parent's plays, so that it is promoted on samples of its own rather than on an estimate inherited from the parent.

Contributions.

- **Method.** ZoomQ combines a factored, action-only box partition with a branching Q-network and UCB action selection (Section 3). The partition admits a dyadic tree (Appendix A), replacing the ball-based bookkeeping of (Cao and Krishnamurthy, 2020) with prefix comparisons.
- **Theory.** For a tabular idealization maintaining a separate action partition per state, we prove regret $\tilde{\mathcal{O}}(K^{(z_A+1)/(z_A+2)})$ (Theorem 4), together with a sublinear bound on the realized partition size (Proposition 5). The exponent is governed by an action-only zooming dimension $z_A \le d_A$ in place of the joint z_{joint} of (Cao and Krishnamurthy, 2020), which can reach $d_S + d_A$, so the state dimension leaves the rate entirely. The cost is a factor $S^{1/(z_A+2)}$ in the leading term, $S = |\mathcal{S}|$, incurred by the per-state tables that let the analysis invoke Lipschitzness in the action alone (Section 4).
- **Experiments.** On three DeepMind Control Suite tasks and `racetrack-v0`, the two discretizations are comparable at small and moderate budgets. At the largest budget the uniform grid falls from its own best return on every task, sharply where the action space is wide: 431 → 145 on `cheetah-run` and 653 → 356 on `cartpole-swingup`, against 320 and 715 for ZoomQ (Section 5).

Related Work. *Adaptive discretization and zooming.* (Kleinberg et al., 2019) and (Bubeck et al., 2011) established the zooming dimension for Lipschitz bandits, and (Slivkins, 2014) extended it to contextual bandits in metric spaces. (Sinclair et al., 2019) gave the first multi-step adaptive-discretization algorithm for episodic RL, analyzed with covering-dimension regret, and (Song and Sun, 2019) gave a related model-free analysis. (Cao and Krishnamurthy, 2020) obtained the first instance-dependent zooming-dimension regret for episodic RL, combining the partition of (Sinclair et al., 2019) with a buffering phase and the clipped decomposition of (Simchowitz and Jamieson, 2019); (Podimata and Slivkins, 2021) extended these ideas to adversarial rewards. These works partition the joint state-action space, whereas we partition the action space alone.

Value-based continuous control by discretization. We adopt the branching Q-network of (Tavakoli et al., 2018) as our value representation. (Tang and Agrawal, 2020) argue that uniform action discretization can be competitive with continuous-action methods on standard benchmarks, and (Seyde et al., 2022) show that decoupling the value function across action dimensions scales discrete critics to high action dimension.

Coarse-to-fine control resolution. Closest to ZoomQ is GQN (Seyde et al., 2024), which grows a discrete action space from coarse to fine inside decoupled Q-learning and scales to $d_A = 38$. CQN (Seo et al., 2024) re-discretizes at every decision, zooming into the highest-valued interval over a fixed number of levels, and (Liu et al., 2025) model action dimensions autoregressively under a coarse-to-fine discretization. All three refine on a schedule or to a depth fixed in advance and shared across the whole action space. ZoomQ instead refines a cell only once that cell's own visit count crosses a resolution-dependent threshold, so the partition is non-uniform and the budget is an upper bound rather than a target; it also conditions each new cell through buffering with atomic sibling promotion, and is the only one of the four with a regret analysis. We do not compare against them empirically. Further discussion is in Appendix D.

2 Preliminaries

Markov Decision Processes (MDPs). We consider a finite-horizon episodic MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, H, P, r)$ with action space $\mathcal{A} \subseteq [0,1]^{d_A}$ (rescaled if needed), horizon $H \in \mathbb{N}$, transition kernel P, and bounded reward $r : \mathcal{S} \times \mathcal{A} \to [0,1]$. A non-stationary policy $\pi = (\pi_h)_{h=1}^H$ induces

$$V_h^\pi(s) = \mathbb{E}_\pi\left[\sum_{h'=h}^{H} r(s_{h'}, a_{h'}) \,\middle|\, s_h = s\right],$$
$$Q_h^\pi(s,a) = r(s,a) + \mathbb{E}\big[V_{h+1}^\pi(s') \mid s, a\big],$$

with optimal counterparts $V_h^\star$ and $Q_h^\star$. Over K episodes from initial states $\{s_1^k\}$, the regret of an algorithm collecting rewards r_h^k is

$$\mathrm{Reg}(K) \;=\; \sum_{k=1}^{K}\Big(V_1^\star(s_1^k) - \sum_{h=1}^{H} r_h^k\Big).$$

We take D_A to be the ℓ_∞ metric on $\mathcal{A}$ throughout.

Assumption 1 (Action-Lipschitz $Q^\star$) *There exists $L_A > 0$ such that, for all $h \in [H]$, $s \in \mathcal{S}$, and $a, a' \in \mathcal{A}$, $|Q_h^\star(s,a) - Q_h^\star(s,a')| \le L_A \cdot D_A(a,a')$.*

Assumption 1 is strictly weaker than the joint Lipschitz condition of (Cao and Krishnamurthy, 2020), which also constrains variation across states; here $Q^\star$ is constrained along the action coordinate alone, at each state separately. It holds whenever the reward and the transition kernel are Lipschitz in the action, as they are under the smooth simulated dynamics of all four benchmarks.

Dyadic boxes and box partitions. A *box* with corner $\ell \in [0,1]^{d_A}$ and side vector $w \in (0,1]^{d_A}$ is the axis-aligned product $C(\ell, w) = \prod_{j=1}^{d_A}[\ell_j, \ell_j + w_j)$, half-open except on faces where $\ell_j + w_j = 1$, so that every action lies in exactly one box of a partition and no tie-breaking is needed. Its *scale* is the ℓ_∞ diameter $s(C) = \max_j w_j$. A *box partition* of $\mathcal{A}$ is a collection of pairwise disjoint boxes whose union covers $\mathcal{A}$, and membership costs d_A scalar comparisons.

A *binary split along axis j* replaces $C(\ell, w)$ with the two children obtained by halving the jth side, $C(\ell, w')$ and $C(\ell + \frac{w_j}{2} e_j, w')$ with $w'_j = w_j/2$ and $w'_i = w_i$ for $i \neq j$. By doing so, the children tile the parent, so replacing a box by its children preserves both coverage and disjointness. Splitting cycles the axis as $j(m) = (m \bmod d_A) + 1$ at depth m, so $s(C)$ halves once every d_A levels and a box at depth m has scale $2^{-\lfloor m/d_A \rfloor}$.

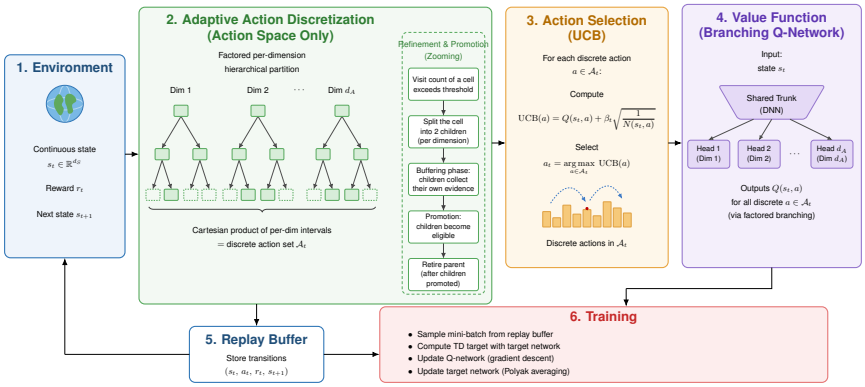


Figure 1: Overview of ZoomQ.

Zooming dimension on the action space. Following (Cao and Krishnamurthy, 2020, Definitions 3 and 4), we adapt the zooming dimension to the action-only setting. Define the stage-h sub-optimality gap $\mathrm{gap}_h(s,a) = V_h^\star(s) - Q_h^\star(s,a)$ and the near-optimal set at scale r as

$$\mathcal{Z}_{h,r} = \{(s,a) : \mathrm{gap}_h(s,a) \le \left(\tfrac{4(H+1)\max(1,L_A)}{d_{\max}} + 2L_A\right) r\},$$

where $d_{\max} = \mathrm{diam}(\mathcal{A}) = 1$. For fixed s, let $\mathcal{A}_{h,r}^\star(s) = \{a \in \mathcal{A} : (s,a) \in \mathcal{Z}_{h,r}\}$ be the slice of r-near-optimal actions at s, and let $N_r^{\mathrm{pack}}(\cdot)$ denote the r-packing number under D_A.

Definition 2 (Action zooming dimension) *The* action zooming dimension z_A *is the smallest exponent d such that, uniformly over states and stages, the r-packing number of the near-optimal action slice grows no faster than r^{-d} as the scale r shrinks:*

$$z_A = \inf\Big\{d \ge 0 \;:\; \sup_{s\in\mathcal{S}} N_r^{\mathrm{pack}}\big(\mathcal{A}_{h,r}^\star(s)\big) \le c\, r^{-d}, \quad \forall\, h \in [H],\ r \in (0,1]\Big\}.$$

By construction $z_A \le d_A$, with equality when the near-optimal action set fills $\mathcal{A}$ at every scale.

3 The ZoomQ Algorithm

ZoomQ holds two objects: d_A trees of boxes, one per action coordinate, and a branching Q-network valuing their leaves. A leaf is *active* if it may be selected, or *buffering* if it is still accumulating an estimate of its own; the active leaves partition their coordinate's range $[0,1]$. Two visit thresholds move leaves between the states, each depending on a cell's own count and scale rather than on a global schedule, which is what leaves the partition non-uniform. Figure 1 summarizes one environment step, Algorithm 1 gives pseudocode, and Appendix A shows a partition evolving over training.

3.1 Partition, values, and action selection

Factored partition. Each axis i carries an independent tree whose boxes are intervals in $[0,1]$, with an active set $\mathcal{P}_k^{(i)}$ and a buffering set $\mathcal{B}_k^{(i)}$. Because an action is chosen one coordinate at a time, the joint action partition $\prod_i \mathcal{P}_k^{(i)}$ is never materialized; storage is $\sum_i |\mathcal{P}_k^{(i)}|$, additive in d_A rather than exponential. A single *budget* N_{tot} caps the total across axes, $\sum_i(|\mathcal{P}_k^{(i)}| + |\mathcal{B}_k^{(i)}|) \le N_{\mathrm{tot}}$; the cap

is on the sum, so one coordinate may refine while another stays coarse. Value heads are resized as the partition changes, rows appended when children are created and removed when a parent retires, so ZoomQ never holds more output capacity than the budget allows. Setting $N_{\text{tot}} = Nd_A$ against a uniform grid of N cells per axis therefore matches the arms on cell count and bounds ZoomQ's parameter count by Uniform's. Each box carries a visit count $n(C)$ and its scale $s(C)$; active sets are initialized to a uniform grid of 2^{d_0} cells per axis (we use $d_0 \in \{1, 3\}$) and buffering sets start empty.

Value representation. A shared multilayer-perceptron trunk maps the observation s to a latent vector, from which d_A linear heads produce, for each axis i, action-values $\hat{Q}^{(i)}(s, C)$ over that axis's current boxes $C \in \mathcal{P}_k^{(i)} \cup \mathcal{B}_k^{(i)}$. Buffering boxes are given heads so that they can be trained, and masked from selection. Training is temporal-difference regression on replayed transitions against a target network, with the branching target

$$y \;=\; r + \gamma \cdot \frac{1}{d_A} \sum_{i=1}^{d_A} \max_{C \in \mathcal{P}^{(i)}} \hat{Q}^{(i)}_{\text{target}}(s', C), \tag{1}$$

where $\gamma \in (0, 1)$ is a discount factor and the axis-wise maxima range over *active* boxes only. Discounting is an implementation choice, standard on these benchmarks; the finite-horizon setting of Section 2 and the analysis of Section 4 are undiscounted. The maxima are the action-branching decomposition of (Tavakoli et al., 2018), and it is an approximation rather than only a saving: choosing each coordinate by its own arg max recovers the joint maximizer exactly when $Q^\star$ separates across coordinates, and not otherwise. We inherit it rather than introduce it, and both discretization arms use the same network and the same decomposition, so it does not confound the comparison in Section 5.

Action selection. At each step, for each axis i independently, an active box is chosen by an upper-confidence rule,

$$C_h^{(i)} \;=\; \operatorname*{arg\,max}_{C \in \mathcal{P}_k^{(i)}} \left[\hat{Q}^{(i)}(s_h, C) \;+\; c\sqrt{\frac{\log t}{n(C)}} \right], \tag{2}$$

with t the cumulative environment-step count and $c > 0$ a fixed exploration coefficient. The played action is the vector of selected centers, $a_h = (\text{center}(C_h^{(1)}), \ldots, \text{center}(C_h^{(d_A)}))$; the actions available at episode k are therefore exactly the centers of the active cells. The idealization of Section 4 is more permissive, crediting any played a to the cell containing it. Buffering boxes are excluded from both the arg max in (2) and the target maxima in (1). At evaluation the bonus is dropped and the greedy action is played. Note that the policy still depends on s, through the shared trunk; only the partition is state-independent.

3.2 When to split: the rule for N_{split}

A box $C \in \mathcal{P}_k^{(i)}$ becomes eligible to split once its visit count crosses a resolution-dependent threshold:

$$n(C) \;\geq\; N_{\text{split}}(C) \;:=\; c_1 \cdot s(C)^{-2}. \tag{3}$$

The exponent is set by where the standard error $\mathcal{O}(\sigma/\sqrt{n})$ of the value estimate falls below the Lipschitz variation $L_A \cdot s(C)$ across the box, which is the point at which finer structure becomes resolvable. We use $c_1 = 1$ in the implementation; the analysis of Section 4 requires a larger constant.

When (3) holds, C is split into two children of side $s(C)/2$. Each child is created with a fresh value head, visit count 0, and placed in the buffering set $\mathcal{B}_k^{(i)}$. Following the correction to the analysis of (Cao and Krishnamurthy, 2020), children do not inherit the parent's estimate at creation, so a child is promoted on evidence of its own. The parent remains in $\mathcal{P}_k^{(i)}$ and continues to be played while its children buffer. When several boxes are eligible at once and refining all of them would exceed the budget, splits are prioritized by visit count.

3.3 When to promote: the rule for $N_{\min}$

A buffering child $C' \in \mathcal{B}_k^{(i)}$ accumulates data through a scheduled fraction of its parent's plays. Every B-th time the parent C is selected by (2), the play is *redirected* to a buffering child of C and the transition is stored under that child's head. The tabular idealization uses period $B = H + 1$; the implementation uses $B = 16$ and picks the least-updated sibling.

A child is promoted to active once

$$n(C') \;\geq\; N_{\min}(C') \;:=\; \tfrac{1}{4}\, c_1 \cdot s(C')^{-2}, \tag{4}$$

following (Cao and Krishnamurthy, 2020, Section 4). On the one-dimensional coordinate trees of Section 3.1 a split halves the scale, so $s(C') = s(C)/2$ and $N_{\min}(C') = N_{\text{split}}(C)$. The factor $\frac{1}{4}$ is fixed by that correspondence rather than chosen. In the d_A-dimensional idealization a split halves only one side, so there $N_{\min}(C') \leq N_{\text{split}}(C)$, with equality when the split axis is the widest.

3.4 The budget as a ceiling

One chief advantage of ZoomQ is that refinement is never driven by the budget: neither (3) nor (4) refers to N_{tot}, so raising it can only lift a constraint, never create a split. What differs between the two schemes is what a higher budget buys. Resolving to scale r costs a uniform grid $\Theta(1/r)$ cells per axis, created at once and competing for the same data, the mechanism behind the large-N degradation of Section 5. On the other hand, an interval tree refining along one branch reaches r with $\Theta(\log(1/r))$ cells, since each split replaces one cell by two. Where value is sharply hierarchical the same budget therefore buys exponentially finer resolution; where it is not, the tree spends its cells broadly and degenerates to the grid. Proposition 5 bounds the realized count in general.

4 Theoretical guarantee

Theorem 4 bounds regret at a sublinear rate whose leading exponent depends on an action-only zooming dimension, and Proposition 5 bounds the number of cells instantiated after K episodes. Both concern a *tabular idealization* of ZoomQ, defined next; Algorithm 1 is a heuristic instantiation and is not an object governed by these bounds.

The analyzed algorithm. We make the following assumptions. The state space is finite, $|\mathcal{S}| = S$, and each state carries its own action partition. For every $s \in \mathcal{S}$, the algorithm maintains active and buffering sets $\mathcal{P}_k(s), \mathcal{B}_k(s)$, a visit counter $n^k(s, C)$, and an optimistic estimate $\hat{Q}_h^k(s, C)$. A visit to s with action a updates only the entry $(s, \text{leaf}_k^s(a))$, and the rules of Sections 3.2 and 3.3 are applied per state. Every sample averaged into $\hat{Q}_h^k(s, C)$ is collected at the same state, and only the action varies within C. No comparison of $Q^\star$ across states therefore arises, and Assumption 1 suffices where (Cao and Krishnamurthy, 2020) require joint Lipschitzness. The price is the factor S in (5).

Algorithm 1 ZoomQ: factored action-space zooming with a branching Q-network

Input: axis-wise active/buffering sets $\{\mathcal{P}^{(i)}, \mathcal{B}^{(i)}\}_{i=1}^{d_A}$; branching Q-network $\hat{Q}$ and target $\hat{Q}_{\text{target}}$; replay buffer $\mathcal{D}$; budget N_{tot}; redirection period B; exploration coefficient c; maintenance period T_{maint}.
for environment step $t = 1, 2, \ldots$ **do**
  Observe s.
  for axis $i = 1, \ldots, d_A$ **do**
    $C^{(i)} \leftarrow \arg\max_{C \in \mathcal{P}^{(i)}} \left[\hat{Q}^{(i)}(s, C) + c\sqrt{\log t / n(C)}\right]$. ▷ UCB over active boxes, Eq. (2)
    $n(C^{(i)})$ += 1.
    if $C^{(i)}$ has buffering children **and** $n(C^{(i)}) \bmod B = 0$ **then**
      $C' \leftarrow$ least-updated child of $C^{(i)}$ in $\mathcal{B}^{(i)}$; $\hat{C}^{(i)} \leftarrow C'$; $n(C')$ += 1. ▷ redirect to buffering child
    else
      $\hat{C}^{(i)} \leftarrow C^{(i)}$.
    end if
  end for
  Play $a = (\text{center}(\hat{C}^{(1)}), \ldots, \text{center}(\hat{C}^{(d_A)}))$; observe r, s'; store $(s, (\hat{C}^{(i)})_i, r, s')$ in $\mathcal{D}$.
  Sample a minibatch from $\mathcal{D}$; update $\hat{Q}$ by TD regression toward the branching target (1) (active boxes only); Polyak-update $\hat{Q}_{\text{target}}$.
  if $t \bmod T_{\text{maint}} = 0$ **then** ▷ maintenance
    for parent C such that both children reach $N_{\min}$ (Eq. (4)) **do**
      Move both children $\mathcal{B}^{(i)} \to \mathcal{P}^{(i)}$; relabel transitions stored under C onto a graduated child; retire C and compact the value heads.
    end for
    for active C with $n(C) \geq N_{\text{split}}(C)$ (Eq. (3)), most-visited first while budget permits **do**
      Split C into two children of side $s(C)/2$ with fresh heads; place both in $\mathcal{B}^{(i)}$; C remains in $\mathcal{P}^{(i)}$.
    end for
  end if
end for

Assumption 3 (Optimistic-Q conditions) *In the tabular idealization, $\mathcal{S}$ is finite, rewards are bounded in $[0, 1]$, the value-target noise is conditionally σ-sub-Gaussian given the trajectory filtration $\mathcal{G}_h^k$ with $\sigma = \Theta(H)$, the learning rate is $\alpha_t = (H+1)/(H+t)$ with $t = n^k(s_h^k, C_h^k)$, and each box carries the confidence bonus $b_t = \Theta(\sigma\sqrt{H \log(SHK/\delta)/t})$.*

These are the conditions of (Cao and Krishnamurthy, 2020, Section 4) and (Jin et al., 2018), specialized to state-indexed box tables. The learning-rate schedule makes the recursive weights α_t^i a convergent series. The scale $\sigma = \Theta(H)$ is forced rather than assumed: with rewards in $[0, 1]$ the next-state value $V_{h+1}^\star$ ranges over $[0, H]$, so the value-target noise is bounded, and hence sub-Gaussian, at scale H rather than at scale 1.

Theorem 4 (Regret of tabular ZoomQ) *Fix $\delta \in (0, 1)$ and $K \geq 1$, and suppose Assumptions 1 and 3 hold. Then, with probability at least $1 - \delta$, the regret of the tabular action-only box-zooming Q-learner after K episodes satisfies*

$$\text{Reg}(K) \leq \tilde{\mathcal{O}}\Big(\underbrace{H\big(SH^2\big)^{\frac{1}{z_A+2}} K^{\frac{z_A+1}{z_A+2}}}_{\text{zooming}} + \underbrace{H^2 S^{\frac{2}{d_A+2}} (KH)^{\frac{d_A}{d_A+2}}}_{\text{first visits}} \Big), \tag{5}$$

where z_A is the action zooming dimension (Definition 2) and $\tilde{\mathcal{O}}$ suppresses factors logarithmic in $S, H, K, 1/\delta$.

Note that this implies $\mathrm{Reg}(K)$ is sublinear in K, and the zooming term dominates whenever $d_A \le 2z_A + 2$, in particular whenever $d_A \le 2$.

Proposition 5 (Realized partition size) *Under the same conditions, the total number of boxes ever instantiated across all states, active or buffering, after K episodes is at most*

$$M_K := \sum_{s \in \mathcal{S}} \big(|\mathcal{P}_K(s)| + |\mathcal{B}_K(s)|\big) \le \tilde{\mathcal{O}}\Big(S^{\frac{2}{d_A+2}} (KH)^{\frac{d_A}{d_A+2}}\Big), \tag{6}$$

and the number of those boxes that are near-optimal at their own state, in the sense of (15), is at most $\tilde{\mathcal{O}}(S^{2/(z_A+2)}(KH)^{z_A/(z_A+2)})$.

The leading regret exponent is governed by $z_A \le d_A$ and is unaffected by how the state space is represented. The realized cell count is governed by K, H, and S rather than by N_{tot}, so any N_{tot} exceeding the right-hand side of (6) leaves the partition, and hence the regret bound, unchanged. This is the formal counterpart of Section 3.4.

Proof sketch. The full proof is in Appendix C. Write $\mathrm{clip}[\mu \mid \nu] = \mu \cdot \mathbf{1}\{\mu \ge \nu\}$ for the operator of (Simchowitz and Jamieson, 2019), which zeroes a surplus below the threshold ν. Iterating the optimistic value-error recursion and applying the clipped decomposition bounds regret by a sum, over stages and state-box pairs, of clipped per-box surpluses, plus a martingale term and a first-visit term.

Lemma 8 bounds the surplus of a box C at state s with t visits by a statistical term $\tilde{\mathcal{O}}(\sigma\sqrt{H/t})$ plus an action-Lipschitz bias $L_A s(C)$. Only the bias uses geometry, and because the table is indexed by state, every sample entering it was collected at s; Assumption 1 therefore suffices and no cross-state comparison of $Q^\star$ arises. This is precisely where (Cao and Krishnamurthy, 2020) require a joint Lipschitz condition. The clipped decomposition then splits the boxes at each scale r into those whose minimum gap exceeds $\Theta(r)$, whose surplus the clip zeroes outright, and those whose gap is small. At a *fixed* state, a small-gap box of scale r has $\|a - a_0\|_\infty \le r$ throughout, so $\mathrm{gap}_h(s,a) \le \mathrm{gap}_h(s,a_0) + L_A r$ and the whole box is near-optimal at s. Such boxes form an r-packing of $\mathcal{A}^\star_{h,r}(s)$, giving at most $c\,r^{-z_A}$ of them (Lemma 11) and $Sc\,r^{-z_A}$ after summing over states. Counting per state is essential: a shared partition would instead have to pack $\bigcup_s \mathcal{A}^\star_{h,r}(s)$, which z_A does not control (Remark 12).

Balancing the per-scale contributions against the contribution ignored below a cutoff gives $r_0 = (SH^2/K)^{1/(z_A+2)}$ and the zooming term of (5). Proposition 5 follows from a separate counting argument: a box of scale r needs $\Theta(r^{-2})$ visits to split and the total budget is KH, so at most $\mathcal{O}(KHr^2)$ boxes of scale r are ever created, while geometry caps the count at $\mathcal{O}(Sr^{-d_A})$ and Lemma 11 caps the near-optimal ones at $Sc\,r^{-z_A}$. Taking the smaller bound at each scale bounds M_K, and hence the first-visit term. The two results are proved independently, so the argument is not circular.

Comparisons. A uniform box grid at scale ε attains the covering-dimension rate $\tilde{\mathcal{O}}(K^{(d_A+1)/(d_A+2)})$, on which (5) improves polynomially whenever $z_A < d_A$. The joint algorithm of (Cao and Krishnamurthy, 2020) achieves $\tilde{\mathcal{O}}(K^{(z_{\mathrm{joint}}+1)/(z_{\mathrm{joint}}+2)})$ with z_{joint} as large as $d_S + d_A$, whereas $z_A \le d_A$. The comparison is not free: (Cao and Krishnamurthy, 2020) generalize across states through the

partition and so apply to continuous $\mathcal{S}$, while (5) buys its exponent by keeping a table per state, at a cost of $S^{1/(z_A+2)}$ in the leading term. The two results are complementary rather than ordered, and whether the separation manifests at the budgets studied here is a separate, empirical question.

5 Experiments

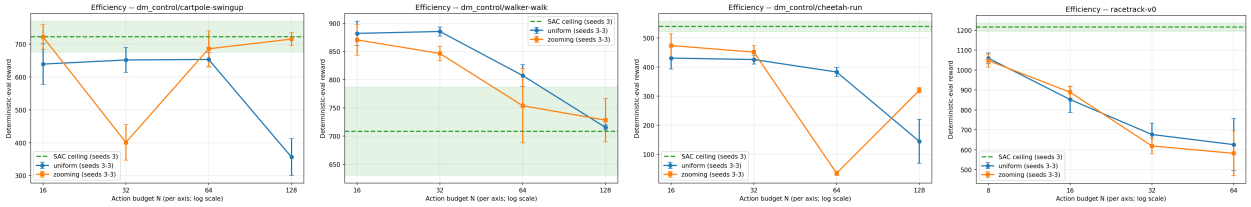
Setup. We evaluate ZoomQ on four continuous-control tasks: three from the DeepMind Control Suite (Tunyasuvunakool et al., 2020), simulated with MuJoCo (Todorov et al., 2012) (in particular: `cartpole-swingup`, $d_A = 1$; `walker-walk`, $d_A = 6$; and `cheetah-run`, $d_A = 6$), and finally `racetrack-v0` from `highway-env` (Leurent, 2018) ($d_A = 1$, flattened kinematic observation of dimension 25). We use discount $\gamma = 0.99$ throughout. The discrete-action arms share the same branching Q-network (Tavakoli et al., 2018), a two-layer multilayer-perceptron trunk (width 256) supplying d_A linear value heads, trained against a Polyak-averaged target network with Adam and a replay buffer. Learning rate, batch size, and target smoothing are 3×10^{-4}, 256, and $\tau = 5 \times 10^{-3}$ on the DeepMind Control tasks, and 5×10^{-4}, 128, and $\tau = 10^{-2}$ on `racetrack-v0`. ZoomQ uses $c = 0.3$, $B = 16$, maintenance every 2000 steps, and initial resolution 2^{d_0} cells per axis with $d_0 = 1$ (DeepMind Control) or 3 (racetrack).

Baselines and protocol. We compare against (i) Uniform, a fixed grid of N cells per axis with the same branching Q-network and matched budget $N_{\text{tot}} = Nd_A$, and (ii) SAC (Haarnoja et al., 2018), a budget-free continuous-action reference (from-scratch implementation, same trunk width). To isolate the partition from the exploration policy, both discretization arms select actions with the same rule (2) and differ only in whether the grid is fixed or refined. The discretization arms sweep $N \in \{16, 32, 64, 128\}$ on the DeepMind Control tasks and $N \in \{8, 16, 32, 64\}$ on `racetrack-v0`, three seeds each $(42, 43, 44)$. During training we periodically run deterministic (greedy, no-exploration) evaluation episodes and log their mean return; all reported numbers and curves are this *deterministic-evaluation* return rather than the exploration-contaminated training return, which we found to understate the greedy policy. A run is summarized by the mean of its last three evaluation points, and bands are standard errors across seeds.

At matched budget, the two discretizations are comparable. Table 1 and Figure 2 show no consistent advantage for either arm at small-to-moderate N. Uniform is marginally ahead on `walker-walk` at $N \in \{16, 32, 64\}$, the `cartpole-swingup` ordering is mixed and dominated by seed variance, and on `racetrack-v0` the arms track each other closely. Since the seed-to-seed spread is comparable to the between-arm differences, we do not read a matched-budget advantage for ZoomQ from these runs. Section 3.4 predicts as much: at moderate N both arms use most of their cells, and the two mechanisms coincide.

Uniform grids degrade with a larger budget, whereas ZoomQ does not. The consistent difference appears at the largest budget. Uniform falls from its own best value on every task, most sharply on `cheetah-run` ($431 \to 145$ from N=16 to 128) and `cartpole-swingup` ($653 \to 356$ from N=64 to 128), while ZoomQ holds 715 and 320 at N=128. Inspecting the final partitions shows why: ZoomQ does not spend its budget, instantiating 56/128 cells at N=128 on `cartpole-swingup`, 484/768 on `walker-walk`, and 570/768 on `cheetah-run`. This is the behavior Proposition 5 predicts.

We are careful about what this does and does not establish. It does not show that ZoomQ allocates a matched number of cells more effectively, since at large N the arms no longer hold equal active-cell counts and no matched-count advantage appears elsewhere in the sweep. What it shows is



(a) cartpole-swingup (d_A=1) **(b)** walker-walk (d_A=6) **(c)** cheetah-run (d_A=6) **(d)** racetrack-v0 (d_A=1)

Figure 2: Efficiency: deterministic-evaluation return versus action budget N for UNIFORM and ZOOMQ, with SAC as a budget-free reference band (three seeds, bands showing standard errors). Neither discretization holds a consistent advantage at small-to-moderate N. At the largest N the uniform grid declines while ZOOMQ does not, because it leaves part of its budget unspent (see text). SAC exceeds both on three of the four tasks and trails both on `walker-walk`.

Table 1: Final deterministic-evaluation return (mean over three seeds) by task, arm, and action budget N. DeepMind Control tasks use $N \in \{16, 32, 64, 128\}$ and `racetrack-v0` uses $N \in \{8, 16, 32, 64\}$. SAC is budget-free (shown once per task). Seed standard errors range from roughly 5 to 130 return units, so differences between the two discretizations are frequently within the seed spread. These three-seed results should be read as preliminary.

Task	Arm	Action budget N				SAC
DeepMind Control Suite	($N = 16,\ 32,\ 64,\ 128$)					
cartpole-swingup (d_A=1)	Uniform	639	652	653	356	723
	ZOOMQ	722	401	686	715	
walker-walk (d_A=6)	Uniform	882	886	808	716	709
	ZOOMQ	871	847	754	728	
cheetah-run (d_A=6)	Uniform	431	426	383	145	539
	ZOOMQ	474	451	35	320	
highway-env	($N = 8,\ 16,\ 32,\ 64$)					
racetrack-v0 (d_A=1)	Uniform	1059	852	677	626	1215
	ZOOMQ	1048	889	619	582	

narrower and more useful: an over-generous N costs the uniform grid a large fraction of its return and costs ZOOMQ little, so the resolution need not be identified in advance. One configuration departs from the pattern, ZOOMQ at N=64 on `cheetah-run` (return 35), where a single seed collapsed; we attribute this to the value instability reported below, though at three seeds the explanations cannot be separated.

Sensitivity is comparable, but to a different quantity. We compared the spread of UNIFORM across N against that of ZOOMQ across its budget, redirection period $B \in \{8, 16, 32\}$, and initial depth d_0 (Appendix B, Figure 5). The ranges are of similar magnitude: ZOOMQ-to-UNIFORM spread ratios are 0.84, 1.07, 1.08, and 1.53 on `walker-walk`, `racetrack-v0`, `cartpole-swingup`, and `cheetah-run`. Adaptive discretization therefore does not eliminate hyperparameter sensitivity; it relocates it, from a resolution that must be chosen correctly to a schedule that trades between the limiting behaviors of Section 3.4. This is weaker than tuning-free operation, and is the only claim we make.

SAC is usually but not always an upper bound. SAC exceeds both discretizations on three of the four tasks: by 65 on `cheetah-run` (539 against a best discretized value of 474), by 156 on `racetrack-v0` (1215 against 1059), and by 1 on `cartpole-swingup`. On `walker-walk` it trails both, reaching 709 against 886 for UNIFORM and 871 for ZOOMQ. Running a single budget ($N = 32$) over a longer horizon (Appendix B) leaves both discretization arms below SAC on every task. On `racetrack-v0` both value-based arms peak near 150k steps and decay through the end of the 600k-step run while SAC stays stable; we attribute this to value instability under prolonged training, common to UNIFORM and ZOOMQ and not specific to the adaptive partition, and note it as a limitation on asymptotic conclusions for that task.

6 Conclusion

We described ZOOMQ, an action-only, factored adaptive-discretization scheme for value-based deep RL, built around a split threshold of $\Theta(s(C)^{-2})$ visits and a promotion threshold a quarter as large. Restricting the partition to the action space keeps the discrete action count additive in d_A and, in a per-state tabular idealization, makes the rate-determining quantity $z_A \leq d_A$ rather than the joint z_{joint} of (Cao and Krishnamurthy, 2020). The central finding concerns the action budget: for a uniform grid it is a resolution parameter with a penalty on both sides, whereas for ZOOMQ it is a ceiling the realized partition need not reach (Proposition 5). Empirically, the adaptive partition performs comparably to a matched-budget uniform grid at small and moderate budgets, and its distinguishing behavior appears at large budgets, where the uniform grid degrades and ZOOMQ does not.

Limitations. The central limitation is statistical. All results use three seeds, and the seed-to-seed standard errors (roughly 5 to 130 return units) are comparable to, and in several cells larger than, the differences between arms. We cannot separate the effects we report from run-to-run noise, and the large-N pattern, though consistent in sign across tasks, rests on few runs; the `cheetah-run` N=64 collapse illustrates how much a single seed can move an entry. All comparisons should be read as preliminary and directional. We also do not compare empirically against the coarse-to-fine methods discussed above, and the value-based arms exhibit late-training instability on `racetrack-v0`.

Another concerns the gap between the analyzed object and the implementation. Theorem 4 keeps a separate action partition and value table for every state, allowing Assumption 1 to carry the analysis without any cross-state comparison of $Q^\star$, at an extra factor $S = |\mathcal{S}|$. The bound is therefore vacuous on the continuous state spaces of our experiments, where the deep network supplies the cross-state generalization the analysis forgoes. Establishing an action-only rate for a state-generalizing partition remains open.

The third is by design: the action partition is state-independent. The policy depends on the observation through the value function, but the discretization itself is global, so where the useful resolution shifts sharply with context a single partition will be suboptimal regardless of budget.

Future work. A resolution-adaptive schedule that spends a large budget in full would test whether deeper zooming improves on uniform discretization rather than only avoiding its failure mode; stabilizing the value-based arms would make the SAC comparison conclusive; and a state-aware partition would let the discretization adapt to local structure.

References

Sébastien Bubeck, Rémi Munos, Gilles Stoltz, and Csaba Szepesvári. X-armed bandits. *Journal of Machine Learning Research*, 12, 2011.

Tongyi Cao and Akshay Krishnamurthy. Provably adaptive reinforcement learning in metric spaces. In *Advances in Neural Information Processing Systems*, 2020.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. Pmlr, 2018.

Min Hua, Xinda Qi, Dong Chen, Kun Jiang, Zemin Eitan Liu, Hongyu Sun, Quan Zhou, and Hongming Xu. Multi-agent reinforcement learning for connected and automated vehicles control: Recent advancements and future prospects. *IEEE Transactions on Automation Science and Engineering*, 2025.

Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is q-learning provably efficient? *Advances in neural information processing systems*, 31, 2018.

B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE transactions on intelligent transportation systems*, 23(6):4909–4926, 2021.

Robert Kleinberg, Aleksandrs Slivkins, and Eli Upfal. Bandits and experts in metric spaces. *Journal of the ACM*, 66(4), 2019.

Edouard Leurent. An environment for autonomous driving decision-making. https://github.com/Farama-Foundation/HighwayEnv, 2018.

Jijia Liu, Feng Gao, Qingmin Liao, Chao Yu, and Yu Wang. Learning from suboptimal data in continuous control via auto-regressive soft q-network. *arXiv preprint arXiv:2502.00288*, 2025.

Chara Podimata and Alex Slivkins. Adaptive discretization for adversarial lipschitz bandits. In *Conference on Learning Theory*, pages 3788–3805. PMLR, 2021.

Younggyo Seo, Jafar Uruç, and Stephen James. Continuous control with coarse-to-fine reinforcement learning. *arXiv preprint arXiv:2407.07787*, 2024.

Tim Seyde, Peter Werner, Wilko Schwarting, Igor Gilitschenski, Martin Riedmiller, Daniela Rus, and Markus Wulfmeier. Solving continuous control via q-learning. *arXiv preprint arXiv:2210.12566*, 2022.

Tim Seyde, Peter Werner, Wilko Schwarting, Markus Wulfmeier, and Daniela Rus. Growing q-networks: Solving continuous control tasks with adaptive control resolution. In *6th Annual Learning for Dynamics & Control Conference*, pages 1646–1661. PMLR, 2024.

Max Simchowitz and Kevin G. Jamieson. Non-asymptotic gap-dependent regret bounds for tabular MDPs. In *Advances in Neural Information Processing Systems*, 2019.

Sean R. Sinclair, Siddhartha Banerjee, and Christina Lee Yu. Adaptive discretization for episodic reinforcement learning in metric spaces. In *Proceedings of the ACM on Measurement and Analysis of Computing Systems (SIGMETRICS)*, 2019.

Aleksandrs Slivkins. Contextual bandits with similarity information. *Journal of Machine Learning Research*, 15, 2014.

Zhao Song and Wen Sun. Efficient model-free reinforcement learning in metric spaces. *arXiv preprint arXiv:1905.00475*, 2019.

Yunhao Tang and Shipra Agrawal. Discretizing continuous action space for on-policy optimization. In *AAAI Conference on Artificial Intelligence*, 2020.

Arash Tavakoli, Fabio Pardo, and Petar Kormushev. Action branching architectures for deep reinforcement learning. In *Proceedings of the aaai conference on artificial intelligence*, volume 32, 2018.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020.

Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge university press, 2019.

A The Dyadic Tree of Boxes

This appendix develops the box tree summarized in Section 2. The proofs in Appendix C repeatedly invoke ancestor and descendant relationships, and the comparison with the ball-based partition of (Cao and Krishnamurthy, 2020) is most direct in tree-theoretic terms.

The box tree. Define a tree $\mathcal{T}$ rooted at $C_\varnothing := [0,1]^{d_A}$. Every node of $\mathcal{T}$ is a box, and every internal node has two children, obtained by bisecting along a designated axis. We adopt deterministic axis cycling, splitting the children of a node at depth ℓ along axis $j(\ell) := (\ell \bmod d_A) + 1$. The resulting tree is therefore d_A-ary in cycle but binary in branching, with a unique left or right child at every internal node. We assign each node a binary path-encoding $b(C) \in \{0,1\}^{\mathrm{depth}(C)}$ recording the sequence of left-and-right choices on the root-to-C path, with $b(C_\varnothing)$ the empty string.

Definition 6 (Ancestor and descendant in the box tree) *For two boxes $C, C' \in \mathcal{T}$, we say that C is an* ancestor *of C' (and C' a* descendant *of C), written $C' \preceq C$, if C' is reachable from C by a finite sequence of child steps in $\mathcal{T}$. We say the relationship is* strict*, written $C' \prec C$, if additionally $C' \neq C$.*

The order $\preceq$ has three properties that make the box tree convenient.

Lemma 7 (Box-tree invariants) *For any two boxes $C, C' \in \mathcal{T}$:*

1. *(Inclusion equals descent.) $C' \preceq C$ if and only if $C' \subseteq C$.*
2. *(Disjointness or nesting.) Either $C \cap C' = \varnothing$, or one is an ancestor of the other.*
3. *(Path-encoding characterization.) $C' \preceq C$ if and only if $b(C)$ is a prefix of $b(C')$. The depth difference equals the difference in encoding lengths,* $\mathrm{depth}(C') - \mathrm{depth}(C) = |b(C')| - |b(C)|$.

Sketch: Properties (1) and (2) are immediate from the bisection rule. Each child is a half-open half of its parent, so descent corresponds to nested halving, and any two boxes in $\mathcal{T}$ either lie in disjoint half-spaces of a common ancestor (disjoint) or share a root-to-leaf path (nested). Property (3) restates descent as a string-prefix relation and follows from the unique left/right encoding at each internal node. □

The path-encoding of property (3) makes ancestor and descendant queries cheap. Testing $C' \preceq C$ reduces to a single prefix comparison on binary strings of length at most $\log_2(1/s_{\min})$.

Active leaves. ZoomQ maintains, at every episode k, an *active set* $\mathcal{P}_k$ and a *buffering set* $\mathcal{B}_k$ of boxes drawn from $\mathcal{T}$. The promotion rules of Section 3.3 guarantee that $\mathcal{P}_k$ and $\mathcal{B}_k$ obey three structural invariants throughout training:

1. (Coverage.) $\bigcup_{C \in \mathcal{P}_k} C = \mathcal{A} = [0,1]^{d_A}$.
2. (No overlap.) For distinct $C, C' \in \mathcal{P}_k$, $C \cap C' = \varnothing$, equivalently $C \not\prec C'$ and $C' \not\prec C$.
3. (Buffering as descent.) Every $C' \in \mathcal{B}_k$ has its parent in $\mathcal{P}_k$, i.e., $C' \prec C$ with $C \in \mathcal{P}_k$ and $\mathrm{depth}(C') = \mathrm{depth}(C) + 1$.

Combined with property (2) of Lemma 7, these invariants imply that $\mathcal{P}_k$ is the set of leaves of a *retracted* version of $\mathcal{T}$ in which subtrees rooted at promoted parents have been collapsed back to those parents. The buffering set $\mathcal{B}_k$ holds the depth-1 children of those leaves of $\mathcal{P}_k$ that have been

split but whose children have not yet completed buffering. In the deep implementation (Section 3.1), one such tree is maintained independently for each action coordinate and the joint action partition is the product of the coordinate-wise active sets. The single-tree description used here corresponds to the tabular idealization analyzed in Section 4.

The leaf-of-an-action map. The coverage and no-overlap invariants imply that every action $a \in \mathcal{A}$ lies in a unique active box. Define

$$\text{leaf}_k(a) \;:=\; \text{the unique } C \in \mathcal{P}_k \text{ with } a \in C. \tag{7}$$

This is the analog, in our setting, of the relevant-ball selection step of (Cao and Krishnamurthy, 2020). Operationally, $\text{leaf}_k(a)$ is computed by walking the tree from the root. At each internal node, we read the bit of a's position along the level's split axis to choose the left or right child, and we stop at the first node in $\mathcal{P}_k$. The cost is at most $\text{depth}_{\max}(\mathcal{P}_k) \le \log_2(1/s_{\min})$ scalar comparisons, where $s_{\min} = \min_{C \in \mathcal{P}_k} s(C)$.

Ancestors and descendants in the ball partition. The zooming algorithm of (Cao and Krishnamurthy, 2020) maintains a partition of $\mathcal{S} \times \mathcal{A}$ into balls, built by a different rule. When a ball B of radius r is split, the algorithm constructs an $r/2$-*net* of B's domain (Algorithm 1, line 10 of (Cao and Krishnamurthy, 2020)), and these net-points become centers of children with radius $r/2$. The resulting children are not in general a bisection of the parent, since their union may extend beyond B and they may overlap one another. The doubling factor Λ that appears in their regret bound is the maximum number of children that one such split can produce.

To define ancestors and descendants in this overlapping ball partition, (Cao and Krishnamurthy, 2020) use a domain-based construction. For an active ball $B \in \mathcal{P}_h^k$ with radius $r(B)$, the *domain* is

$$\text{dom}_h^k(B) \;:=\; B \;\setminus \bigcup_{B' \in \mathcal{P}_h^k,\, r(B') < r(B)} B', \tag{8}$$

i.e., the points in B that are not contained in any active ball of strictly smaller radius. The *relevant ball* at a point (s, a) is then the unique smallest active ball B such that $(s, a) \in \text{dom}_h^k(B)$. A ball B is an *ancestor* of a ball B' in this setup if $r(B) > r(B')$ and $B \supseteq B'$ in the metric, a relation that depends both on the splitting hierarchy, which records the order of creation, and on the current geometric layout, which determines the balls the algorithm has instantiated. The two need not agree. A ball can be a hierarchical descendant of B without being a metric subset of B, because the $r/2$-net may place children whose support extends beyond B.

Boxes eliminate every step of the above complication. Property (1) of Lemma 7 collapses geometric containment ($C' \subseteq C$) and tree descent ($C' \preceq C$) into the same relation. Property (2) eliminates overlap and removes the need for the domain construction (8). Property (3) reduces every ancestor query to a prefix comparison. The doubling factor specializes to $\Lambda = 2$, dropping the $\sqrt{H\Lambda + \Lambda + 1}$ factor in the surplus bound of (Cao and Krishnamurthy, 2020) to $\sqrt{2H + 3}$. All of these simplifications appear repeatedly in the proof in Appendix C. The trade-off is small, since the covering and packing numbers of $[0, 1]^{d_A}$ under boxes and under ℓ_∞-balls differ only by a constant factor independent of K (Kleinberg et al., 2019, Section 5), so the regret rate of Theorem 4 is identical to what one would obtain with ℓ_∞-balls. Only the bookkeeping changes.

B Additional Experimental Results

This appendix collects the two studies referenced in Section 5 but not plotted there.

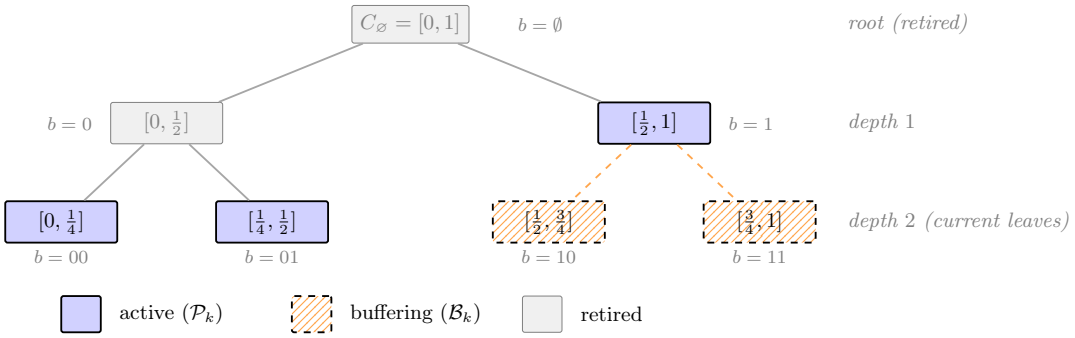


Figure 3: Snapshot of the dyadic box tree on a 1-D action space $\mathcal{A} = [0, 1]$. Each node is labeled with its binary path-encoding $b(C)$. The root has been split into two halves, each of which has in turn been split. The left subtree's children have completed buffering and are active leaves. The right subtree's children are still buffering, so their parent (with encoding $b = 1$) remains an active leaf and continues to be sampled. Active leaves $\mathcal{P}_k = \{[0, \frac{1}{4}], [\frac{1}{4}, \frac{1}{2}], [\frac{1}{2}, 1]\}$ tile $\mathcal{A}$. Any query action $a \in \mathcal{A}$ maps to its unique $\text{leaf}_k(a)$ by walking the tree. For example, $a = 0.6$ has $b(\text{leaf}_k(a)) = 1$, since at depth 1 we go right (into $[\frac{1}{2}, 1]$) and that node is in $\mathcal{P}_k$.

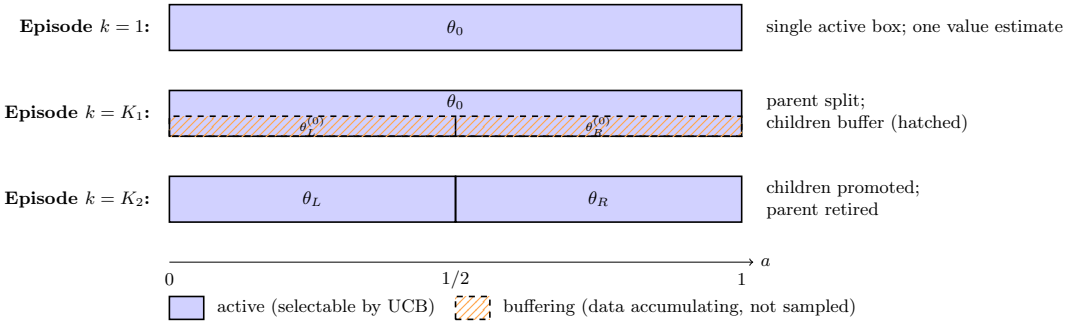
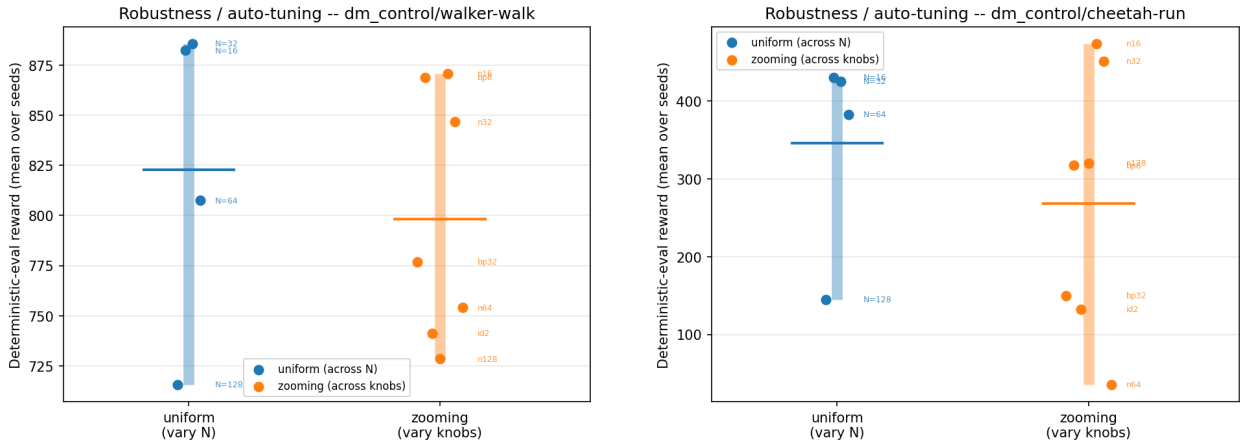


Figure 4: Temporal evolution of ZoomQ's partition on a single action coordinate ($d_A = 1$). Initially the coordinate is a single active box with value estimate θ_0 (shorthand for its entry in the Q-head). Once the box reaches its visit threshold N_{split} it is split, and the two children are placed in the buffering set (hatched) with fresh, non-inherited estimates, while the parent remains active and continues to be selected and played. Once both children have accumulated N_{min} visits of their own (via the redirection mechanism of Section 3.3) they are promoted to active and the parent is retired. Each action coordinate maintains its own such tree.

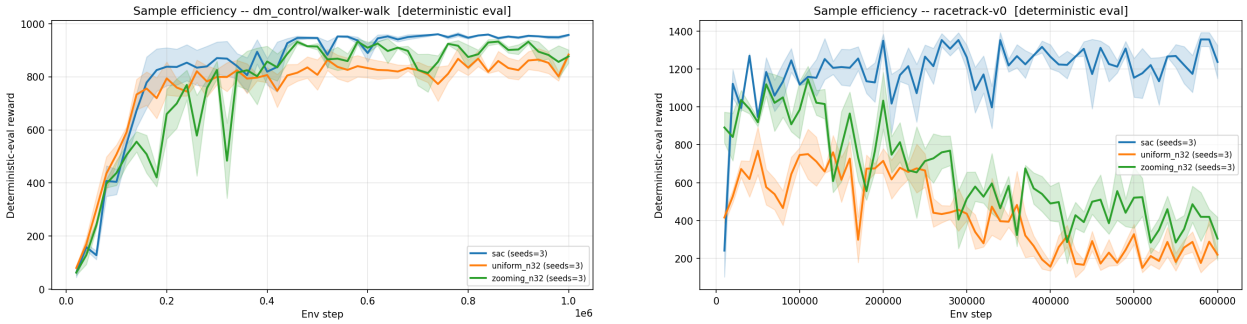
Robustness. Figure 5 compares the spread of Uniform across the action budget N to the spread of ZoomQ across its budget, redirection period $B \in \{8, 16, 32\}$, and initial depth d_0. Each point is one configuration's mean over seeds, and the vertical bar spans its range. The two spreads are comparable in magnitude, with ZoomQ-to-Uniform spread ratios of 0.84 (`walker-walk`), 1.07 (`racetrack-v0`), 1.08 (`cartpole-swingup`), and 1.53 (`cheetah-run`). The redirection period B in particular moves the outcome, consistent with its role in Section 3.4 as the control on how much of the budget is spent. It is the quantity that interpolates between uniform-like behavior at the initial resolution and a deeper partition. We do not claim that B is easier to set than N. We claim only that the failure modes at its two extremes are milder than those of a poorly chosen fixed resolution.

Sample efficiency. Figure 6 plots deterministic-evaluation return against environment steps at a single budget ($N = 32$) over a longer horizon. On `racetrack-v0`, both value-based discretization arms peak near 150k steps and then decline, a value instability common to Uniform and ZoomQ and therefore not attributable to the adaptive partition. SAC remains stable and higher throughout.



(a) walker-walk **(b)** cheetah-run

Figure 5: Robustness: spread of UNIFORM across the action budget N versus ZOOMQ across its hyperparameters (budget N, redirection period $B \in \{8, 16, 32\}$, initial depth d_0). Each point is one configuration's mean over seeds, and the vertical bar spans its range.



(a) walker-walk **(b)** racetrack-v0

Figure 6: Sample efficiency at N=32: deterministic-evaluation return versus environment steps. On `racetrack-v0`, both value-based discretization arms peak near 150k steps and then decline, while SAC remains stable and higher.

B.1 Compute environment

Hardware. All experiments were run on a single workstation with one NVIDIA RTX 4000 Ada Generation GPU (20 GB of VRAM), an Intel Core Ultra 9 285K CPU with 24 cores and 24 threads, and 46 GiB of system memory. Training was performed on the GPU.

Operating system and software. The workstation runs Ubuntu 24.04.2 LTS with kernel 6.17.0-1025-oem and NVIDIA driver 570.211.01. The installed CUDA toolkit is release 12.4, and the driver supports CUDA 12.8. Experiments use Python 3.11.15 with PyTorch 2.11.0, NumPy 1.26.4, `dm_control` 1.0.40, MuJoCo 3.8.0, Gymnasium 1.2.3, and `highway-env` 1.10.2, in a Conda-managed environment. A full environment specification accompanies the code supplement.

Number of runs. The reported studies comprise 180 training runs, three seeds (42, 43, 44) for each configuration across the four tasks. Of these, 108 belong to the efficiency sweep (four action budgets per task for both discretizations, together with SAC), 36 to the robustness sweep over the

buffering period B and the initial depth d_0, and 36 to the sample-efficiency study.

Determinism. At the start of each run we seed PyTorch, NumPy, and Python's `random` module from the run seed. We do not seed the environment random number generator, and we do not enable deterministic GPU kernels, leaving `torch.use_deterministic_algorithms` and cuDNN determinism at their defaults. Runs are therefore statistically rather than bit-wise reproducible. A fixed seed fixes network initialization together with the replay and exploration sampling, but not the environment's stochasticity or low-level kernel nondeterminism. All reported results are aggregates over seeds.

Runtime. Per-run wall-clock time and total compute were not recorded for the experiments reported here.

C Proofs of Theorem 4 and Proposition 5

This appendix proves the two results of Section 4 for the tabular action-only box-zooming optimistic-Q algorithm with per-state tables. Lemma 11, the action-only zooming reduction, is the new technical content. Lemmas 8, 10, and 13 adapt arguments of (Cao and Krishnamurthy, 2020) to state-indexed box tables, which collapses the doubling factor to $\Lambda = 2$. Appendix C.1 fixes notation, Appendices C.2–C.5 prove the lemmas, Appendix C.6 proves Proposition 5, and Appendix C.7 proves Theorem 4. Proposition 5 is proved before Theorem 4 and its proof does not use Theorem 4, so the appeal to it inside Appendix C.7 is not circular.

Throughout, $c_3, c_4, \ldots$ denote universal positive constants whose values may differ between occurrences, while c_1 is the fixed split-threshold constant of (3), and $\tilde{\mathcal{O}}$ suppresses factors logarithmic in $S, K, H, 1/\delta$.

C.1 Setup and notation

Fix $\delta \in (0,1)$ and $K \geq 1$. The state space is finite with $|\mathcal{S}| = S$. Every state s carries its own dyadic box tree over $\mathcal{A}$, with active set $\mathcal{P}_k(s)$, buffering set $\mathcal{B}_k(s)$, and the invariants of Appendix A holding separately for each s. Because the cells are half-open, the leaf map

$$\mathrm{leaf}_k^s(a) \;:=\; \text{the unique } C \in \mathcal{P}_k(s) \text{ with } a \in C$$

is well defined for every $a \in \mathcal{A}$, including on split boundaries.

At step h of episode k the algorithm is at state s_h^k and plays a_h^k. Write $C_h^k = \mathrm{leaf}_k^{s_h^k}(a_h^k)$ for the selected active box. If a buffering redirection fires, every $(H{+}1)$th selection of a box whose children are buffering, let $\hat{C}_h^k \in \mathcal{B}_k(s_h^k)$ be the redirected child, and otherwise $\hat{C}_h^k := C_h^k$. The *update counter* $n^k(s,C)$ counts the number of times $(s_h^{k'}, \hat{C}_h^{k'}) = (s,C)$ over $k' < k$, and the *selection counter* $\tilde{n}^k(s,C)$ counts the number of times $(s_h^{k'}, C_h^{k'}) = (s,C)$. These differ by the redirected plays. Splitting and promotion are driven by $n^k(s,C)$ and are applied to each state independently.

The state-indexed estimate. For each pair (s,C) the algorithm maintains an optimistic estimate $\hat{Q}_h^k(s,C)$ by the Jin-style update with learning rate $\alpha_t = (H+1)/(H+t)$. Writing $t = n^k(s,C)$ and letting $k_1 < \cdots < k_t$ be the episodes on which (s,C) was the updated pair, the estimate unrolls to

$$\hat{Q}_h^k(s,C) \;=\; \alpha_t^0\, H \;+\; \sum_{i=1}^{t} \alpha_t^i \left(r_h^{k_i} + V_{h+1}^{k_i}(s_{h+1}^{k_i}) + b_i \right), \tag{9}$$

where $\alpha_t^0 = \prod_{j=1}^{t}(1-\alpha_j)$ and $\alpha_t^i = \alpha_i \prod_{j=i+1}^{t}(1-\alpha_j)$ are the Jin weights, $V_{h+1}^k(s') := \max_{C' \in \mathcal{P}_k(s')} \hat{Q}_{h+1}^k(s', C')$, and $b_i = \Theta\big(\sigma\sqrt{H\log(4SHK/\delta)/i}\big)$ is the bonus of Assumption 3. The point of indexing by state is that every episode k_i entering (9) satisfies $s_h^{k_i} = s$: the samples averaged into $\hat{Q}_h^k(s, C)$ all come from the same state, and only the action varies within C. Because $\alpha_1 = 1$ we have $\alpha_t^0 = 0$ for $t \geq 1$, so the first visit overwrites the H-initialization.

The associated surplus is

$$\beta_t^{\mathrm{opt}}(s, C) \;:=\; \sup_{a \in C} \Big[\hat{Q}_h^k(s, C) - Q_h^\star(s, a)\Big]_+ \tag{10}$$

when $n^k(s, C) = t$, where $[x]_+ = \max(x, 0)$.

The clipping operator. Following (Simchowitz and Jamieson, 2019, Section 2) we write

$$\mathrm{clip}\big[\mu \;\big|\; \nu\big] \;:=\; \mu \cdot \mathbf{1}\{\mu \geq \nu\}, \tag{11}$$

so the operator zeroes a surplus that falls strictly below the threshold ν and returns it unchanged otherwise. It does not cap μ at ν. Consequently a box contributes to the clipped sum only at those visit counts t for which $\beta_t^{\mathrm{opt}}(s, C) \geq \nu$, and the way to bound the contribution of a large-gap box is to show that the set of such t is empty or small, not to replace β_t^{opt} by ν.

Finally, the recursive value-error decomposition of (Jin et al., 2018, Section 3), applied as in (Cao and Krishnamurthy, 2020, Eq. 3) with the relevant ball replaced by the state-indexed leaf map, gives

$$\begin{aligned} V_h^k(s) - Q_h^\star(s, a) \;\leq\; & \mathbf{1}[n^k(s, C) = 0] \cdot H \\ & + \sum_{i=1}^{n^k(s,C)} \alpha_{n^k(s,C)}^i \Big(V_{h+1}^{k_i} - V_{h+1}^\star\Big)(s_{h+1}^{k_i}) \\ & + \beta_{n^k(s,C)}^{\mathrm{opt}}(s, C) \end{aligned} \tag{12}$$

for any a with $C = \mathrm{leaf}_k^s(a)$. The sum bound $\sum_{t=i}^{\infty} \alpha_t^i \leq 1 + 1/H$ is (Jin et al., 2018, Lemma 4.1).

C.2 The optimistic-Q surplus bound

Lemma 8 (Optimistic-Q surplus) *Under Assumptions 1 and 3, with probability at least $1 - \delta/2$, for all $h \in [H]$, $k \in [K]$, states s, and active boxes $C \in \mathcal{P}_k(s)$ with $n^k(s, C) = t \geq 1$,*

$$\beta_t^{\mathrm{opt}}(s, C) \;\leq\; \underbrace{c_3\, \sigma \sqrt{\frac{H \log(4SHKM/\delta)}{t}}}_{\text{statistical}} \;+\; \underbrace{L_A \cdot s(C)}_{\text{action-Lipschitz bias}} . \tag{13}$$

Proof: Fix h, k, s, C and let $t = n^k(s, C) \geq 1$ with update episodes $k_1 < \cdots < k_t$, all of which satisfy $s_h^{k_i} = s$ by the definition of the update counter. Since $\alpha_t^0 = 0$, subtracting $Q_h^\star(s, a) = \sum_{i=1}^{t} \alpha_t^i Q_h^\star(s, a)$

from (9) and applying the Bellman identity $Q_h^\star(s, a_h^{k_i}) = r_h^{k_i} + (P_h V_{h+1}^\star)(s, a_h^{k_i})$ termwise gives

$$\begin{aligned}
&\hat{Q}_h^k(s, C) - Q_h^\star(s, a) \\
&\quad = \underbrace{\sum_{i=1}^{t} \alpha_t^i \big(V_{h+1}^{k_i} - V_{h+1}^\star\big)(s_{h+1}^{k_i})}_{(A)\text{: downstream value error}} \\
&\quad + \underbrace{\sum_{i=1}^{t} \alpha_t^i \, \zeta_h^{k_i}}_{(B)\text{: value-target noise}} + \sum_{i=1}^{t} \alpha_t^i \, b_i \\
&\quad + \underbrace{\sum_{i=1}^{t} \alpha_t^i \big[Q_h^\star(s, a_h^{k_i}) - Q_h^\star(s, a)\big]}_{(C)\text{: action-Lipschitz bias}},
\end{aligned}$$

where $\zeta_h^{k_i} := V_{h+1}^\star(s_{h+1}^{k_i}) - (P_h V_{h+1}^\star)(s, a_h^{k_i})$.

Every occurrence of $Q_h^\star$ in (C) is evaluated at the single state s. This is the step at which state indexing is used. Had one table been shared across states, the same manipulation would leave a residual $\sum_i \alpha_t^i [Q_h^\star(s_h^{k_i}, a) - Q_h^\star(s, a)]$, which Assumption 1 cannot bound because it compares $Q^\star$ across different states.

Term (A) is not part of the surplus. It is carried separately in (12) and discharged by the induction on h in Appendix C.7.

Bounding (B). The sequence $\{\zeta_h^{k_i}\}_i$ is a martingale-difference sequence with respect to $\mathcal{G}_h^{k_i}$, conditionally σ-sub-Gaussian by Assumption 3. Azuma–Hoeffding for weighted sub-Gaussian differences (Wainwright, 2019, Theorem 2.19) gives

$$\mathbb{P}\left(\Big| \sum_{i=1}^{t} \alpha_t^i \, \zeta_h^{k_i} \Big| \geq u \right) \;\leq\; 2 \exp\left(- \frac{u^2}{2\sigma^2 \sum_{i=1}^{t} (\alpha_t^i)^2} \right).$$

The Jin quadratic-weight bound $\sum_{i=1}^{t} (\alpha_t^i)^2 \leq 2H/t$ (Jin et al., 2018, Lemma 4.1(b)) is the only place H enters the statistical rate. Setting $u = \sigma\sqrt{4H \log(4SHKM/\delta)/t}$ and union-bounding over $h \in [H]$, $k \in [K]$, and the $M \leq KH$ state-box pairs that can ever be visited yields the stated bound with probability at least $1 - \delta/4$.

The bonus and optimism. The bonus is designed so that $\sum_{i=1}^{t} \alpha_t^i b_i = \Theta(\sigma\sqrt{H \log(4SHK/\delta)/t})$, as in (Jin et al., 2018, Lemma 4.3). On the event above it dominates (B), certifying optimism and bounding the bonus-plus-noise contribution at the same order.

Bounding (C). The pair (s, C) is updated only on plays at state s whose action lies in C, so $a_h^{k_i} \in C$, and with $a \in C$ the half-open box has $\|a_h^{k_i} - a\|_\infty \leq s(C)$ by the definition of the scale. Assumption 1 bounds each summand by $L_A s(C)$, and $\sum_i \alpha_t^i = 1$ carries this through the weighted average.

Combining and taking the positive part gives (13). □

Remark 9 *Under rewards in $[0, 1]$ and horizon H, the next-state value $V_{h+1}^\star$ ranges over $[0, H]$, so the value-target noise ζ is bounded by H and is therefore sub-Gaussian at scale $\sigma = \Theta(H)$, not $\Theta(1)$. Substituting $\sigma = \Theta(H)$ into (13) gives a statistical term of order $H^{3/2}\sqrt{\log(\cdot)/t}$. We carry σ explicitly and substitute only at the end.*

C.3 The visit counter bound

Lemma 10 (Counter bound) *For any $h \in [H]$, $k \in [K]$, state s, and active box $C \in \mathcal{P}_k(s)$,*

$$n^k(s,C) \;\le\; \tilde{n}^k(s,C) \;\le\; (2H+3)\, c_1 \cdot s(C)^{-2} \;=:\; N_{\max}(C). \tag{14}$$

For buffering boxes $C \in \mathcal{B}_k(s)$ we have $n^k(s,C) < N_{\min}(C)$ and $\tilde{n}^k(s,C) = 0$.

Proof: This is the action-only analog of (Cao and Krishnamurthy, 2020, Lemma 9) with $\Lambda = 2$ and $d_{\max} = 1$, applied within a single state's table.

For $C \in \mathcal{B}_k(s)$, buffering boxes are not selected, so $\tilde{n}^k(s,C) = 0$, and the promotion threshold moves C out of $\mathcal{B}_k(s)$ once $n^k(s,C) \ge N_{\min}(C)$.

For $C \in \mathcal{P}_k(s)$, atomic promotion means C stops being played once both children are promoted. Before splitting, C is played at most $N_{\text{split}}(C) = c_1 s(C)^{-2}$ times by (3). After splitting, C continues to be played while its children buffer. Each child has scale at least $s(C)/2$ and is promoted once it reaches $N_{\min} = \frac{1}{4}c_1(s(C)/2)^{-2} = c_1 s(C)^{-2}$ own-visits. Each own-visit consumes one out of every $H{+}1$ plays of C, and there are $\Lambda = 2$ children, so C is played at most $2(H+1)c_1 s(C)^{-2}$ further times. Summing gives (14). □

C.4 The action-only zooming reduction

Lemma 11 (Action-only zooming reduction) *Fix $h \in [H]$, a state s, and a scale $r \in (0,1]$. Call a box C of scale $s(C) = r$ in the tree of state s* small-gap *if*

$$\min_{a \in C} \mathrm{gap}_h(s,a) \;<\; \frac{4(H+1)\max(1,L_A)}{d_{\max}} \cdot r. \tag{15}$$

Then every small-gap box satisfies $C \cap \mathcal{A} \subseteq \mathcal{A}^\star_{h,r}(s)$, and the number of small-gap boxes of scale r in the tree of state s is at most

$$N_r^{\text{pack}}\big(\mathcal{A}^\star_{h,r}(s)\big) \;\le\; \sup_{s' \in \mathcal{S}} N_r^{\text{pack}}\big(\mathcal{A}^\star_{h,r}(s')\big) \;\le\; c \cdot r^{-z_A}. \tag{16}$$

Summing over states, the number of small-gap boxes of scale r across all trees is at most $Sc\,r^{-z_A}$.

Proof: Fix s and let C be a small-gap box of scale r in the tree of s. Choose $a_0 \in C$ attaining the minimum in (15), so $\mathrm{gap}_h(s,a_0) < \frac{4(H+1)\max(1,L_A)}{d_{\max}} r$. For any $a \in C$ we have $\|a - a_0\|_\infty \le s(C) = r$, so Assumption 1 gives

$$\begin{aligned}
\mathrm{gap}_h(s,a) \;&=\; V_h^\star(s) - Q_h^\star(s,a) \\
&\le\; \mathrm{gap}_h(s,a_0) + L_A r \\
&<\; \big(\tfrac{4(H+1)\max(1,L_A)}{d_{\max}} + L_A\big) r \;\le\; \big(\tfrac{4(H+1)\max(1,L_A)}{d_{\max}} + 2L_A\big) r,
\end{aligned}$$

the defining condition of $\mathcal{A}^\star_{h,r}(s)$. Hence $C \cap \mathcal{A} \subseteq \mathcal{A}^\star_{h,r}(s)$.

Both a_0 and the enclosing box belong to the tree of the same state s, and the displayed inequality compares $Q_h^\star$ only at s. Because no comparison across states occurs, Assumption 1 suffices in place of a joint Lipschitz condition.

Boxes of a common scale r within one tree have pairwise disjoint interiors. Under axis cycling every side of such a box lies in $[r/2, r]$, so distinct centers are at pairwise ℓ_∞ distance at least $r/2$, with equality only when $d_A > 1$; on the interval trees of Section 3.1 the separation is exactly r.

Selecting one center from each small-gap box therefore produces an $(r/2)$-separated subset of $\mathcal{A}^\star_{h,r}(s)$, so the number of such boxes is at most $N^{\mathrm{pack}}_{r/2}(\mathcal{A}^\star_{h,r}(s)) \le c\, 2^{z_A} r^{-z_A}$ by Definition 2, uniformly in s. We absorb 2^{z_A} into c. Summing the per-state bound over the S trees gives the final claim. □

Remark 12 (Why the count is taken per state) *The bound is applied separately within each state's tree and only then summed, making the slice-wise packing number $N^{\mathrm{pack}}_r(\mathcal{A}^\star_{h,r}(s))$ the correct object. A single partition shared across states would contain boxes whose witnesses a_0 belong to different states, and such boxes are contained only in $\bigcup_s \mathcal{A}^\star_{h,r}(s)$. The packing number of that union is not bounded by $\sup_s N^{\mathrm{pack}}_r(\mathcal{A}^\star_{h,r}(s))$: if m states have r-separated state-specific optimal actions, each slice has packing number $\mathcal{O}(1)$ while the union has packing number m. Per-state tables avoid this by never forming the union, at the cost of the factor S.*

C.5 The amortizing lemma

Lemma 13 (Amortizing the buffering schedule) *This is (Cao and Krishnamurthy, 2020, Lemma 8). Fix $i \ge H^2$ and $T_0 \ge 0$. Let $\tilde{\alpha}^i_t$ be the sequence formed from $\{\alpha^i_t\}_{t\ge i}$ by repeating every Hth term starting at $t = T_0$. Then*

$$\sum_{t=i}^{\infty} \tilde{\alpha}^i_t \;\le\; \left(1 + \frac{2}{H}\right)^2. \tag{17}$$

Proof: Proved in (Cao and Krishnamurthy, 2020, Appendix A). The argument is combinatorial and depends only on the $H+1$ redirection cadence, which is identical for boxes and balls and is unaffected by state indexing. □

C.6 Proof of Proposition 5

Write $M_K := \sum_{s\in\mathcal{S}}(|\mathcal{P}_K(s)| + |\mathcal{B}_K(s)|)$ for the total number of boxes ever instantiated. We bound the number of boxes at each dyadic scale in two independent ways and take the smaller at each scale.

The visit budget. A box of scale r is created when a parent of scale r or $2r$ splits, and by (3) a box of scale ρ splits only after accumulating $c_1\rho^{-2}$ own-visits. Each visit consumes one of the KH environment steps available over K episodes, and each split produces $\Lambda = 2$ children. Hence the number of boxes of scale r created across all states is at most

$$m(r) \;\le\; \frac{8KH}{c_1}\, r^2. \tag{18}$$

This holds for every box, irrespective of its gap.

The geometric budget. Within one state's tree, boxes of a common scale r are disjoint subsets of $[0,1]^{d_A}$, so there are at most $\mathcal{O}(r^{-d_A})$ of them, giving $m(r) \le \mathcal{O}(Sr^{-d_A})$ across states. If in addition the box is small-gap in the sense of (15), Lemma 11 sharpens this to $m(r) \le Sc\, r^{-z_A}$.

Balancing. Taking the minimum of (18) and the geometric budget at each scale, the two are equal at

$$r_\star \;=\; \left(\tfrac{c_1 S}{8KH}\right)^{\frac{1}{d_A+2}}, \qquad m(r_\star) \;=\; \mathcal{O}\Big(S^{\frac{2}{d_A+2}}(KH)^{\frac{d_A}{d_A+2}}\Big),$$

and the minimum is increasing in r below $r_\star$ and decreasing above it, so every scale contributes at most $m(r_\star)$. Boxes have scale $2^{-\lfloor \ell/d_A \rfloor}$ at depth ℓ, and the counter bound (Lemma 10) with the KH budget caps the reachable depth at $\mathcal{O}(d_A \log(KH))$, so there are $\mathcal{O}(\log(KH))$ distinct scales. Summing,

$$M_K \;\le\; \tilde{\mathcal{O}}\Big(S^{\frac{2}{d_A+2}}(KH)^{\frac{d_A}{d_A+2}}\Big). \tag{19}$$

Repeating the balance with the sharpened small-gap count Scr^{-z_A} in place of $\mathcal{O}(Sr^{-d_A})$ gives $r_\star = (c_1 cS/8KH)^{1/(z_A+2)}$ and shows that the number of small-gap boxes ever instantiated is at most

$$M_K^{\text{near}} \;\le\; \tilde{\mathcal{O}}\Big(S^{\frac{2}{z_A+2}}(KH)^{\frac{z_A}{z_A+2}}\Big). \tag{20}$$

Both exponents on K are strictly below 1, so the realized partition is sublinear in the interaction budget and is unaffected by N_{tot} once N_{tot} exceeds (19). □

Remark 14 *The naive argument that no box finer than $r_0 = K^{-1/(z_A+2)}$ can gather its $\Theta(r^{-2})$ required visits is false: $r_0^{-2} = K^{2/(z_A+2)} \ll KH$ whenever $z_A > 0$. The visit budget alone does not forbid fine boxes. What limits the partition is the product of the visit budget with the geometric count, computed by the balance above.*

C.7 Proof of Theorem 4

The structure follows (Cao and Krishnamurthy, 2020, Appendix A.4–A.6), with Lemma 8 replacing their surplus lemma, Lemma 10 replacing their Lemma 9 with $\Lambda = 2$, and Lemma 11 replacing the joint-zooming step.

Step 1: per-step decomposition. Iterating (12) over h and applying Lemma 13 to absorb the buffering redirections gives

$$\begin{aligned}\mathrm{Reg}(K) \le \Big(1+\tfrac{2}{H}\Big)^3 \sum_{h,k}\Big[H\mathbf{1}[n^k(s_h^k, C_h^k)=0] + \xi_h^{k+1} \\ + \mathrm{clip}\Big(\beta_{n^k}^{\mathrm{opt}}(s_h^k, C_h^k) \;\Big|\; \tfrac{\mathrm{gap}_h(s_h^k,a_h^k)}{H+1}\Big)\Big],\end{aligned} \tag{21}$$

with $(1+2/H)^3 \le 8$ for $H \ge 2$. The martingale term obeys

$$\sum_{h,k}\xi_h^{k+1} \;\le\; 2\sqrt{2H^3K\log(4HK/\delta)} \tag{22}$$

with probability at least $1-\delta/4$ (Sinclair et al., 2019, Lemma E.9).

Step 2: the first-visit term. For each triple (h,s,C) there is at most one episode at which $n^k(s,C)=0$, so

$$\sum_{h,k} H\mathbf{1}[n^k(s_h^k, C_h^k)=0] \;\le\; H^2 M_K,$$

which Proposition 5 bounds by $\tilde{\mathcal{O}}(H^2 S^{2/(d_A+2)}(KH)^{d_A/(d_A+2)})$. This is sublinear in K. Proposition 5 is proved without reference to Theorem 4, so no circularity arises.

Step 3: the clipped surplus, per box. Reorganize the clipped sum by state-box pair. Fix h and a pair (s, C) of scale r, and let $\mathrm{gap}_h(s, C) := \min_{a \in C} \mathrm{gap}_h(s, a)$.

If (s, C) is not small-gap, that is $\mathrm{gap}_h(s, C) \geq \frac{4(H+1)\max(1, L_A)}{d_{\max}} r$, then by (13) and Lemma 10 the surplus at visit count $t \geq N_{\min}(C)$ satisfies

$$\beta_t^{\mathrm{opt}}(s, C) \;\leq\; c_3 \sigma \sqrt{\tfrac{H \log(4SHK/\delta)}{t}} + L_A r \;<\; \frac{\mathrm{gap}_h(s, C)}{H+1}$$

for all $t \geq N_{\min}(C) = \Theta(r^{-2})$, once the constants in N_{split} and in the bonus are chosen so that the statistical term is at most $L_A r$ at $t = N_{\min}(C)$: the surplus is then at most $2L_A r$, while the clip threshold is at least $4\max(1, L_A) r / d_{\max} > 2L_A r$. The factor $\max(1, L_A)$ is what makes this hold for every $L_A > 0$; Assumption 1 permits $L_A = \Theta(H)$. By the definition (11) the clip is then identically zero at every such t, so a large-gap box contributes nothing. Used correctly, the operator annihilates the contribution rather than capping it at the threshold.

If instead (s, C) is small-gap, the clip may leave the surplus intact, and we bound it by summing (13) over t from $N_{\min}(C)$ to $N_{\max}(C) = \mathcal{O}(Hr^{-2})$:

$$\begin{aligned}\sum_{t=N_{\min}}^{N_{\max}} \beta_t^{\mathrm{opt}}(s, C) &\leq c_4 \sigma \sqrt{H \log(\cdot)} \int_{N_{\min}}^{N_{\max}} \tfrac{dn}{\sqrt{n}} + L_A r N_{\max} \\ &= \mathcal{O}\big(\sigma H \log^{1/2}(\cdot)\, r^{-1}\big) + \mathcal{O}\big(Hr^{-1}\big),\end{aligned}$$

using $\sqrt{N_{\max}} = \mathcal{O}(\sqrt{H}/r)$. With $\sigma = \Theta(H)$ (Remark 9) the first term dominates and the per-box total is $\tilde{\mathcal{O}}(H^2 r^{-1})$.

Step 4: summing over scales and balancing. By Lemma 11 there are at most Scr^{-z_A} small-gap boxes of scale r across states, and there are H stages, so the scale-r contribution is $\tilde{\mathcal{O}}(H \cdot Sr^{-z_A} \cdot H^2 r^{-1}) = \tilde{\mathcal{O}}(SH^3 r^{-(z_A+1)})$. Following (Cao and Krishnamurthy, 2020, proof of Lemma 15), sum over dyadic scales $r \in \{r_0, 2r_0, \ldots, 1\}$ and add the contribution of scales finer than the cutoff r_0, which is bounded by $HKr_0/d_{\max}$. The total clipped surplus is

$$\tilde{\mathcal{O}}\Big(SH^3 r_0^{-(z_A+1)} + HKr_0\Big),$$

and setting $r_0 = (SH^2/K)^{1/(z_A+2)}$ balances the two terms, giving

$$\sum_{h,k} \mathrm{clip}(\cdot) \;\leq\; \tilde{\mathcal{O}}\Big(H\big(SH^2\big)^{\frac{1}{z_A+2}} K^{\frac{z_A+1}{z_A+2}}\Big). \tag{23}$$

Step 5: collecting. Substituting (22), Step 2, and (23) into (21),

$$\begin{aligned}\mathrm{Reg}(K) \;\leq\; &\tilde{\mathcal{O}}\Big(H\big(SH^2\big)^{\frac{1}{z_A+2}} K^{\frac{z_A+1}{z_A+2}}\Big) \\ &+ \tilde{\mathcal{O}}\Big(H^2 S^{\frac{2}{d_A+2}} (KH)^{\frac{d_A}{d_A+2}}\Big) + \tilde{\mathcal{O}}\Big(\sqrt{H^3 K \log(HK/\delta)}\Big).\end{aligned}$$

The three exponents on K are $\frac{z_A+1}{z_A+2}$, $\frac{d_A}{d_A+2}$, and $\frac{1}{2}$, all strictly below 1, so the bound is sublinear. Since $\frac{z_A+1}{z_A+2} \geq \frac{1}{2}$ for $z_A \geq 0$, the martingale term never dominates. The first-visit term is dominated by the zooming term exactly when $\frac{d_A}{d_A+2} \leq \frac{z_A+1}{z_A+2}$, which rearranges to $2(z_A+2) \geq d_A + 2$, i.e. $d_A \leq 2z_A + 2$; this holds unconditionally when $d_A \leq 2$. Otherwise the term is retained as the second additive term of Theorem 4. □

Remark 15 (Where the linear term went) *An earlier form of this argument bounded the number of instantiated boxes crudely by $\mathcal{O}(K)$, turning the first-visit term into $\tilde{\mathcal{O}}(HK)$. That term is linear in K and is not dominated by $K^{(z_A+1)/(z_A+2)}$, since the latter exponent is strictly below 1. Proposition 5 is what removes it, by showing that the realized partition is sublinear.*

D Note on the Driving Benchmark

We use `racetrack-v0` from `highway-env` (Leurent, 2018) as a testbed because it is lightweight, reproducible, and pairs a one-dimensional continuous steering action with a 25-dimensional kinematic observation, the asymmetry the action-only restriction is designed for. Autonomous driving is not the focus of this paper and we make no claims about driving performance.

Two features of the driving literature motivate the design rather than the application. First, the observation space is high-dimensional, rendering joint state–action partitioning of the kind in (Cao and Krishnamurthy, 2020) impractical and motivating the restriction to the action space. Second, in lane-keeping and cornering the optimal steering distribution concentrates on narrow, locally distinct ranges (Kiran et al., 2021), the structure a small action zooming dimension formalizes. Readers interested primarily in driving applications should consult (Kiran et al., 2021) and (Hua et al., 2025) rather than this appendix.