FACTORED Q-CHUNKING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning (RL) from demonstrations can teach robots long-horizon manipulation skills despite sparse rewards. Q-chunking, a simple and effective recipe for this setting, has the policy output a short sequence (a *chunk*) of future actions and scores the whole chunk with one critic. However, it commits every joint of the robot to the same chunk, even though many robots combine parts that act on different time scales, such as a mobile base carrying two arms, or an arm carrying a dexterous hand. Letting some joints re-plan mid-chunk breaks value learning, because a critic over the full chunk then scores actions that are never executed. We propose Factored Q-Chunking (FQC), which commits a slow part of the action for the full chunk and re-plans a fast part halfway through. FQC gives each part its own critic that scores exactly the actions that part executes. On DexJoCo (arm and hand) and BiGym (base and two arms), FQC substantially improves online fine-tuning over Q-chunking under the same actor and budget. Ablations show that matching each critic to what is actually executed is essential.

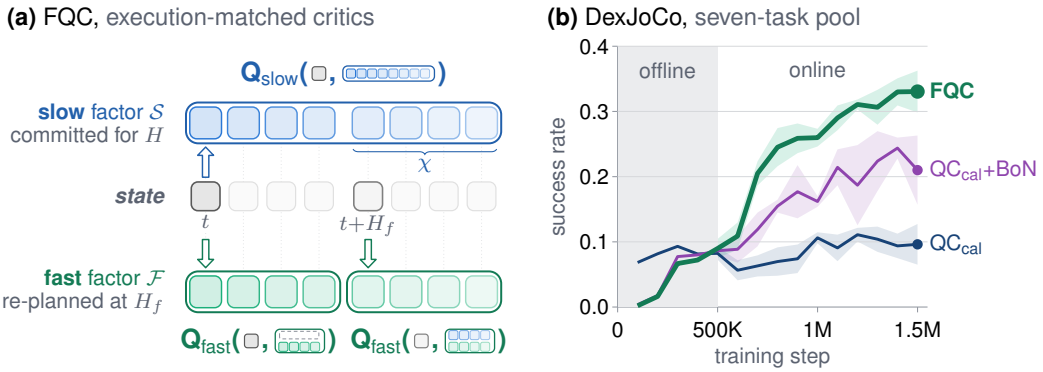


Figure 1: **FQC gives each factor a critic that scores what that factor executes.** (a) $H{=}8$, $H_f{=}4$. Squares are states; the state at t plans both factors and the one at $t{+}H_f$ re-plans only the fast factor (hollow arrows), so the slow factor stays committed for H. Each critic is drawn next to the chunk it scores and bootstraps at the end of that chunk, when its factor next re-plans; at $t{+}H_f$ the fast critic also reads the slow remainder χ (dashed: the empty slot at a slow boundary). Every decision ranks 32 candidates. (b) Seeds 51001–51003, ±1 s.d. bands: QC_{cal} is one critic with FQC's clamp; QC_{cal}+BoN also ranks 32 candidates; five seeds in Table 1.

1 INTRODUCTION

Reinforcement learning (RL) can in principle learn a skill from its reward alone, but on long-horizon manipulation with a sparse reward an agent that explores from scratch rarely reaches it. Offline-to-online RL instead pre-trains on demonstrations and then fine-tunes online; the question is how to turn a few demonstrations into exploration that finds the reward.

Q-chunking (Li et al., 2025a) is a simple and effective recipe for this setting. Its policy outputs a *chunk* of H future actions and executes them all before deciding again, and one *critic*, which estimates the return of an action, scores the whole chunk. Chunks explore coherently, and the value the critic is trained toward (its target) is unbiased: it sums the rewards of exactly the H executed

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

actions and adds the critic's own estimate at the next decision (its *bootstrap*). Recent work re-plans the whole action vector sooner or picks the chunk length per state (Li et al., 2025b; Gireesh et al., 2026; Chen et al., 2026; Shin et al., 2026), but every coordinate (every joint) still shares one *horizon*, the number of steps a decision commits to.

We propose instead to vary the horizon across the action vector. Many robots combine parts that act on different time scales: a mobile base carries two arms, and an arm carries a hand. Picture a base driving toward a dishwasher while the arms correct their grasp of a plate: the base can commit to its path for the whole chunk, whereas the arms should adjust before the chunk ends. Our idea is to keep one part of the action vector committed for the full chunk while the rest re-plans mid-chunk.

Re-planning part of the chunk, however, breaks Q-chunking's value learning. A critic over the full chunk now scores actions that are never executed: the actions it reads for the re-planned part in the second half of the chunk were replaced, so it evaluates a policy other than the one that acts. A critic that instead stops at the re-plan never scores what the committed part does after it.

Factored Q-Chunking (FQC; Figure 1a) resolves this by factoring the critic the same way as the action. The policy commits a *slow factor* (the base, or the arm) for the whole chunk of H steps and re-plans a *fast factor* once, halfway through. Each factor has its own critic, which reads exactly the actions that factor executes and bootstraps where that factor next re-plans, so it never scores an action that is not executed; the other factor's actions executed alongside it averages over. The fast critic also reads χ, the slow actions still committed when it re-plans, which makes its mid-chunk target exact (Proposition 1); in practice χ is optional (Section 5.5). FQC keeps the actor and losses of QC-FQL (Li et al., 2025a; Park et al., 2025a), the Q-chunking variant we compare against, whose actor is a generative model of chunks that executes one sample; FQC instead draws 32 samples at each decision and executes the one its critics score highest (Section 4.3).

Under the same actor, budget, and bound on the critic's bootstrap (the value clamp of Section 4.3, applied to both agents), FQC improves online fine-tuning over Q-chunking on DexJoCo (arm and hand) and BiGym (base and two arms): mean success over the seven DexJoCo and eight BiGym tasks that are not at the floor or ceiling for every method (Section 5.1) rises from 0.102 to 0.327 and from 0.302 to 0.480 (Table 1), and every FQC seed scores above every QC_{cal} seed. The gain appears online, from a comparable start (Figure 2). A two-by-two ablation on both suites (Section 5.3) shows first that the critics must match execution: re-planning the fast factor under one whole-chunk critic, or giving each factor a critic while executing the whole chunk open-loop, does not improve on a single critic that ranks the same 32 candidates. That ranked single critic closes most of the BiGym gap and over half of the DexJoCo gap on its own, but trails FQC on every DexJoCo seed and is within seed spread of it on BiGym. The complementary split (arms slow, or hand slow) and one random partition do about as well (Section 5.5), consistent with a gain from two execution-matched horizons rather than from which part is slow.

2 Related Work

Offline-to-online RL. Offline-to-online RL uses prior data to accelerate online learning. The simplest approach pre-trains with an offline objective and keeps optimizing it online (Kumar et al., 2020; Nair et al., 2020); the resulting values are often pessimistic on actions outside the dataset, which slows exploration. Cal-QL (Nakamoto et al., 2023) calibrates them by lower-bounding values with Monte-Carlo returns, and RLPD (Ball et al., 2023) instead learns online from scratch and treats the offline data as extra replay. FQC works on a different axis: it changes how actions are committed and valued, not how the offline data is used. Its value clamp (Section 4.3) specializes Cal-QL's bound to sparse terminals and is shared with the clamped baseline; we run RLPD and IQL (Kostrikov et al., 2021) at our budget (Appendix E).

Action chunking in RL. Predicting a block $a = (a_0, \dots, a_{H-1})$ and executing it open-loop was popularized in imitation learning (Zhao et al., 2023; Chi et al., 2023); fine-tuning such a policy with RL is a separate line (Ren et al., 2024). Q-chunking (Li et al., 2025a) runs RL directly in the chunked action space: a critic $Q(s, a[0{:}H])$ yields unbiased n-step backups, and chunks give temporally coherent exploration. FQC keeps this recipe's actor and changes how the chunk is executed and valued. The main Q-chunking agent executes the best of N samples of its flow behavior policy under the critic, as in EMaQ and IDQL (Ghasemipour et al., 2021; Hansen-Estruch et al., 2023); its

QC-FQL variant (the FQL actor of Park et al., 2025a), our baseline, executes one distilled sample. FQC ranks 32 samples of that one-step student, and Section 5.4 gives the baseline the same ranking. Tian et al. (2025) condition the critic on short action sequences with n-step returns while keeping a one-step actor. AC3 (Yang et al., 2026) trains a chunked actor-critic with intra-chunk n-step returns, and SEAR (Nagy et al., 2026) and Lee et al. (2026) re-plan early and train the critic on the executed prefix of the chunk; all of them commit every coordinate together. CQN (Seo et al., 2024) selects an action coarse to fine over discretized bins with a critic factored over dimensions, and CQN-AS (Seo & Abbeel, 2025) extends it to action sequences, with results on BiGym and RLBench. There the factoring makes maximization over bins tractable; FQC keeps the chunk continuous and factors it by horizon.

Choosing the horizon. The closest prior work is Decoupled Q-Chunking (DQC; Li et al., 2025b). DQC notes that a policy which must execute a long chunk open-loop can lack reactivity and is hard to model, and decouples the policy's chunk length from the critic's: a long-chunk critic propagates value over many steps, and the policy acts on a shorter partial chunk, scored by a distilled critic that backs up optimistically to the best completion of that partial chunk. FQC shares the goal of keeping the long backup while re-planning sooner, but decouples along the action vector rather than in time: DQC re-plans every coordinate together on the shorter horizon, whereas FQC keeps part of the vector committed while the rest re-plans (Figure 1a). The two also value a re-planned action differently: DQC scores a partial chunk by its best completion, while FQC's fast critic bootstraps where its factor next re-plans, given the slow actions already committed. DQC is evaluated on offline goal-conditioned tasks; we run it at our budget in Section 5.4 and Appendix D. Adaptive Q-Chunking (AQC; Gireesh et al., 2026) trains critics for several chunk lengths and picks one per state by a discount-normalized advantage, using short chunks near contact and long ones in free motion; ACSAC and ACH (Chen et al., 2026; Shin et al., 2026) score several lengths with a transformer critic and pick one per state. These methods vary the horizon across states and FQC across the action vector; the two are complementary, since a per-state selector could choose each factor's horizon.

Multiple control frequencies and temporal abstraction. Closest in structure to FQC, Lee et al. (2020) formalize action variables with different control frequencies as a factored-action MDP with multiple action persistences, show that making a stationary policy action-persistent can be arbitrarily suboptimal, and propose action-persistent policy iteration and actor-critic methods. FQC shares the premise that coordinates can act on different time scales; it commits each factor to a chunk of varying actions rather than holding one action, learns from offline data, and gives each factor a critic at its own horizon. Concurrent options (Rohanimanesh & Mahadevan, 2001) run temporally extended actions on disjoint state variables in parallel. More broadly, hierarchical RL learns temporal abstraction over the whole action: options add a termination condition to a low-level policy (Sutton et al., 1999; Bacon et al., 2017), macro-actions execute open-loop sequences from a given set (Durugkar et al., 2016), and FiGAR and TempoRL learn how long to repeat an action or when to act (Sharma et al., 2017; Biedenkapp et al., 2021). FQC also runs two time scales, but assigns them to disjoint coordinates rather than to levels of a hierarchy, so both factors are trained by standard TD at their own horizons, with no termination condition or high-level policy.

Factored action spaces. Splitting a value over action coordinates, mainly to make maximization tractable in large action spaces, has a long history (Tavakoli et al., 2018; Metz et al., 2017; Seyde et al., 2023). Tang et al. (2022) characterize when an additive decomposition is unbiased, which is why an additive variant, FQC_{add}, is among our comparisons (Section 5.2). Value decomposition in cooperative multi-agent RL (Sunehag et al., 2018; Rashid et al., 2018; Son et al., 2019) trains per-agent utilities through one joint TD target. FQC factors for a different reason: its two heads read different coordinates at different horizons, each with its own TD target, and they rank jointly sampled candidates instead of maximizing each factor separately (Section 4.3).

3 Background

Offline-to-online RL. We consider the same setting as Q-chunking (Li et al., 2025a): an infinite-horizon MDP $(\mathcal{X}, \mathcal{A}, T, r, \gamma)$ with states $s \in \mathcal{X}$ and actions in $\mathcal{A} = \mathbb{R}^D$, equipped with a demonstration dataset $\mathcal{D}$. The agent is pre-trained on $\mathcal{D}$ and then fine-tuned with online interaction. The reward is sparse: $+1$ when the task succeeds, which ends the episode, and 0 otherwise; the terminal is absorb-

ing, so every value lies in $[0, 1]$. We write $R_{t:t+k} = \sum_{i=0}^{k-1} \gamma^i r_{t+i}$ for the k-step discounted return and m_{t+k} for the survival mask, which is zero at the absorbing terminal. Throughout, $\gamma = 0.99$.

Q-chunking. A chunked policy outputs H actions $a = (a_0, \ldots, a_{H-1}) \in \mathbb{R}^{H \times D}$ at once and executes them open-loop. Q-chunking trains a critic on the whole chunk, $Q(s_t, a[0{:}H])$, with the H-step target

$$y \;=\; R_{t:t+H} \;+\; \gamma^H \, m_{t+H} \, \bar{Q}(s_{t+H},\, \pi(s_{t+H}))\,, \tag{1}$$

where $\bar{Q}$ is a delayed target network and π is the agent's action-selection rule. The critic reads exactly the H actions that produced $R_{t:t+H}$, and the bootstrapped chunk is the next decision the agent takes, so this backup is an unbiased H-step return (Li et al., 2025a, Prop. A.1), unlike an n-step return on a one-step critic. We use the released instantiation QC-FQL (Li et al., 2025a), whose actor is that of FQL (Park et al., 2025a): a flow-matching (Lipman et al., 2023) behavior-cloning teacher is distilled into a one-step student $\mu_\psi(s, z)$ (one flow step per chunk) that maximizes Q, and π executes one student sample. Q-chunking's main agent instead executes the best of N samples of its flow behavior policy under Q.

4 Factored Q-Chunking

FQC executes the two parts of the action vector on different horizons and gives each part a critic that scores exactly what it executes. It keeps Q-chunking's setting, actor, and form of backup; the two changes are (1) *per-factor execution*: part of the action vector is committed for the whole chunk while the rest re-plans mid-chunk (Section 4.1); and (2) *execution-matched critics*: each part has its own critic, which reads exactly what that part executes and bootstraps where that part next re-plans (Section 4.2). Section 4.3 then gives the practical implementation: action selection, the actor, and a value clamp that our baseline shares.

4.1 Per-factor execution

We partition the D action coordinates into a slow factor and a fast factor, with index sets $\mathcal{S} \sqcup \mathcal{F} = \{0, \ldots, D-1\}$, and write $a[0{:}k, \mathcal{I}]$ for the first k steps of a chunk on the coordinates in $\mathcal{I}$. The split is fixed before training. By default it is read off the action layout: the slow factor is the mobile base on BiGym and the arm on DexJoCo, and the fast factor is the rest (the two arms; the hand). The names slow and fast refer to the horizon each factor is given; Section 5.5 swaps the assignment and draws it at random.

FQC executes the two factors on different horizons (Figure 1a). At a *slow boundary* ($t \equiv 0 \bmod H$) the agent selects a full chunk $a \in \mathbb{R}^{H \times D}$. Its slow block $a[0{:}H, \mathcal{S}]$ is committed for all H steps, and its fast block runs for $H_f < H$ steps. At the *mid-chunk re-plan* ($t \equiv H_f \bmod H$) the agent selects again, keeps only the new fast block, and holds the rest of the slow block fixed. We call this committed rest the *slow remainder*: $\chi = a[H_f{:}H, \mathcal{S}]$ at a mid-chunk re-plan and $\chi = \varnothing$ at a slow boundary. Throughout, $H = 8$ and $H_f = 4$, so the fast factor re-plans once per chunk.

4.2 Execution-matched critics

FQC replaces the single critic with two heads. Each reads only its own factor's actions, up to that factor's next re-plan; the fast head also reads the slow remainder:

$$Q_{\text{slow}}(s,\, a[0{:}H, \mathcal{S}])\,, \qquad Q_{\text{fast}}(s,\, \chi,\, a[0{:}H_f, \mathcal{F}])\,. \tag{2}$$

Each head bootstraps where its factor next re-plans:

$$y_{\text{slow}} = R_{t:t+H} + \gamma^H m_{t+H}\, \bar{Q}_{\text{slow}}(s_{t+H},\, \pi(s_{t+H})[0{:}H, \mathcal{S}])\,, \tag{3}$$

$$y_{\text{fast}} = R_{t:t+H_f} + \gamma^{H_f} m_{t+H_f}\, \bar{Q}_{\text{fast}}\big(s_{t+H_f},\, \chi',\, \pi(s_{t+H_f}, \chi')[0{:}H_f, \mathcal{F}]\big)\,, \tag{4}$$

where χ' is the slow remainder in force at $t{+}H_f$: $a[H_f{:}H, \mathcal{S}]$ after a slow boundary and $\varnothing$ after a mid-chunk re-plan. We call this alternation the two-phase fast target. Both heads are trained by squared TD error to these targets.

Why the execution must change with the critic is clearest from the fast factor's backup. Writing $a_{\mathcal{F}} = a[0{:}H_f, \mathcal{F}]$ for the fast block a head reads and a' for the block it bootstraps (target networks and masks omitted):

$$\begin{array}{lll}
\text{Q-chunking} & Q(s_t, a[0{:}H]) \leftarrow R_{t:t+H} + \gamma^{H}\, Q\big(s_{t+H}, a'\big) & a' \text{: next decision} \\
\text{open-loop split} & Q_{\text{fast}}(s_t, a_{\mathcal{F}}) \leftarrow R_{t:t+H_f} + \gamma^{H_f} Q_{\text{fast}}\big(s_{t+H_f}, a'_{\mathcal{F}}\big) & a'_{\mathcal{F}} \text{: never executed} \\
\text{partial re-plan} & Q(s_t, a[0{:}H]) \leftarrow R_{t:t+H} + \gamma^{H}\, Q\big(s_{t+H}, a'\big) & a[H_f{:}H, \mathcal{F}] \text{: replaced} \\
\text{FQC} & Q_{\text{fast}}(s_t, \chi, a_{\mathcal{F}}) \leftarrow R_{t:t+H_f} + \gamma^{H_f} Q_{\text{fast}}\big(s_{t+H_f}, \chi', a'_{\mathcal{F}}\big) & a'_{\mathcal{F}} \text{: next decision}
\end{array} \tag{5}$$

Q-chunking's backup is unbiased because its critic reads every executed action and bootstraps at the next decision. Splitting the critic while still executing the whole chunk open-loop (the open-loop split) breaks both properties: the fast actions $a[H_f{:}H, \mathcal{F}]$ are executed but read by neither head (48 of 128 executed coordinates per chunk on a 16-D BiGym task; Appendix C), and the bootstrapped fast block is a decision the executor never takes, since it plays the committed tail instead. Re-planning the fast factor at H_f restores both: every executed fast action is read, and the bootstrapped block is the re-plan the agent actually makes. Re-planning under the whole-chunk critic instead (the partial re-plan, third line) breaks the first property: the chunk it reads still holds the fast actions $a[H_f{:}H, \mathcal{F}]$ that the re-plan replaced, while the rewards come from their replacements. The slow head needs no change, since its factor is committed for exactly H steps and (3) bootstraps at the next slow boundary. At a slow boundary each head does not read the other factor's actions executed alongside, and averages over them. We formalize the fast backup below.

Proposition 1 (The fast backup is a Bellman backup on an augmented chunk MDP)**.** *Fix the FQC executor, take its decision epochs to be the steps $t \equiv 0 \pmod{H_f}$, and define the augmented state $\tilde{s}_t = (s_t, \chi_t)$. Then:*

(i) the decision-epoch process is a Markov decision process over $\tilde{s}$, and the FQC executor is a stationary policy on it;

(ii) at a mid-chunk re-plan, (4) with Q_{fast} reading $\tilde{s}$ is the exact H_f-step Bellman consistency condition for that policy;

(iii) a fast head that reads s alone converges in the least-squares sense to the χ-marginal $Q^{\text{marg}}(s, a_{\mathcal{F}}) = \mathbb{E}[\, Q(\tilde{s}, a_{\mathcal{F}}) \mid s, a_{\mathcal{F}}]$ under the on-policy distribution of χ given s, and ranking by Q^{marg} instead of Q at a mid-chunk re-plan loses at most $2 \max_{a_{\mathcal{F}}} \text{osc}_{\chi}\, Q(s, \chi, a_{\mathcal{F}})$, where osc_{χ} is the range of Q over χ.

The proof is in Appendix J. Default FQC trains the conditional head of part (ii). Without χ the fast head learns the marginal of part (iii), whose regret bound we measure to be small; the two are within seed noise (Section 5.5), so χ is optional. The two-phase target assumes one mid-chunk re-plan per chunk, hence $H{=}2H_f$; this tie comes from the χ backup, not from running two horizons.

4.3 Practical implementation

Action selection by ranking. Like Q-chunking's main agent, FQC selects actions by ranking samples under its critics, and uses the same rule both to act and as π in (3)–(4); it samples the one-step student. At a slow boundary it draws $N{=}32$ candidates $a^{(k)} \sim \mu_{\psi}(\cdot \mid s)$ and picks $a^{\star} = \arg\max_k J(s, a^{(k)})$, where

$$J(s, a) \;=\; \tfrac{1}{2}[Q_{\text{slow}}(s, a[0{:}H, \mathcal{S}]) + Q_{\text{fast}}(s, \varnothing, a[0{:}H_f, \mathcal{F}])]\,. \tag{6}$$

At a mid-chunk re-plan it draws new candidates, keeps χ fixed, and picks the fast block with the largest $Q_{\text{fast}}(s, \chi, a_{\mathcal{F}})$. Both heads estimate the same discounted return and differ only in the coordinates they read and how far they back up, so their average is well scaled; ranking with either head alone moves the BiGym pool by at most about two points (Appendix N). Released QC-FQL executes one student sample; since ranking alone changes the policy, Section 5.4 gives the single critic the same rule.

Actor. The actor is QC-FQL's, with its flow-matching and distillation losses unchanged; its value term uses J in place of the single critic's Q ($\alpha = 100$, $w = 2$; Appendix A).

Value clamp. Matching execution does not bound the bootstrap. Under the sparse absorbing reward $V^\star(s) \in [0, 1]$, and on a successful demonstration with $k(s)$ steps left the Monte-Carlo return is $\gamma^{k(s)}$; yet at the 100K offline endpoint an earlier unclamped two-head critic, averaged over its two heads on held-out demonstrations, already exceeds 1 near the goal on `dishwasherclosetrays` (median 1.23) and `wallcupboardopen` (2.54). FQC therefore clamps *each* head's bootstrap, not the Q prediction:

$$\hat{V}(s') \;=\; \mathrm{clip}\big(V(s'),\, \underline{v}(s'),\, 1\big), \qquad \underline{v}(s') \;=\; \begin{cases} \gamma^{k(s')} & s' \in \mathcal{D}, \\ 0 & \text{otherwise}, \end{cases} \tag{7}$$

where V is the head's target-network bootstrap in (3) or (4). The floor is Cal-QL's bound (Nakamoto et al., 2023) specialized to sparse success, and requires successful demonstrations. The prefix carries at most the one terminal reward, so bounding $V(s')$ bounds the whole target. The clamped target operator remains a contraction, and clamping never moves the bootstrap away from a value the bounds admit (Lemma 1). QC_{cal} applies the same clamp to the single critic, so the clamp alone does not separate the two (Section 5.4).

Training data. Both heads are trained on a $50/50$ demonstration/replay mixture as two-member ensembles with mean aggregation, matching the released Q-chunking critic. Demonstration chunks are split the same way as executed ones, with χ read off the recorded chunk; since they were not generated by this executor, their targets are off-policy, as in any offline backup. Network inputs and the full execution loop are in Appendix A.

5 Experiments

We evaluate FQC on two sparse-reward manipulation suites whose embodiments combine distinct parts, and answer three questions:

- **(Q1)** Must the critics match execution: does re-planning part of the chunk help under one whole-chunk critic, or splitting the critic without re-planning?
- **(Q2)** How does FQC compare with Q-chunking under the same actor, budget, and value clamp, and how much of the gain comes from ranking candidates, from critic capacity, from a shorter horizon, or from two execution-matched horizons?
- **(Q3)** How do the choice of slow factor and the input χ affect performance?

5.1 Environments and datasets

DexJoCo (Wang et al., 2026) is an arm with a dexterous hand: state-based RL from a few demonstrations, 22-D actions (slow: the 6-DoF arm; fast: the 16-DoF hand), sparse terminal reward, and 500K offline + 1M online steps. Four of its eleven tasks stay at the floor for every agent we ran on them, and we pool the other seven. **BiGym** (Chernyadev et al., 2024) is a mobile base with two arms: 70–80-D state, 16-D actions (slow: the 4 base coordinates; fast: the 12 arm coordinates; 15-D with 3 base coordinates on `moveplate`), 2–64 human demonstrations per task, sparse terminal reward, and 100K offline + 500K online steps. Of its 39 tasks, 26 stay near zero and 5 at or above 0.95 for every method compared, and we pool the other eight. On DexJoCo every method uses the same enlarged flow actor ($1024^{\times 4}$, 30 flow steps). Task lists, the pooling rule, and hyperparameters are in Appendices G and O.

5.2 Comparisons

All agents share the QC-FQL actor (network, flow-matching and distillation losses), optimizer, and environment-step budget; they differ in the critic, in how the chunk is executed, and, for the $N{=}32$ rows, in the action-selection rule. Each comparison isolates one ingredient of FQC (Table 1).

QC is QC-FQL, executing one student sample: on BiGym the released, unclamped agent; on DexJoCo we trained only the clamped agent, so QC there is an earlier batch of QC_{cal}. **QC_{cal}** adds FQC's value clamp; it is the baseline under the same clamp. **QC_{cal}+BoN** also ranks 32 student candidates with the single critic, both to act and in the target, as FQC does; it isolates ranking from factoring. The

Table 1: **With the same ranking, a design does well only when its critics match its execution.** N: candidate chunks ranked per decision (1: the released QC-FQL sample). The top block crosses one or two critics with one or two execution horizons (Section 5.3). Both pools, protocol of Section 5.2, per task in Appendix B: mean $\pm$ 1 s.d. over five seeds (per seed in Appendix M). Grey: seeds 51001–51003, mean only. ‡ eval-only on the $H{=}8$ QC_{cal} checkpoints; trained on DexJoCo. § No unclamped QC was trained on DexJoCo: this cell is an earlier batch of the clamped QC_{cal} agent. —: not run.

variant	what it changes	N	DexJoCo (7)	BiGym (8)
critics × execution horizons, all ranking 32 candidates				
QC_{cal}+BoN	one critic, one horizon	32	0.231 ± 0.038	0.440 ± 0.067
open-loop split	two critics, one horizon	32	0.205 ± 0.015	0.346 ± 0.055
partial re-plan	one critic, two horizons	32	0.237 ± 0.031	0.362 ± 0.076
FQC	two critics, two horizons	32	$\mathbf{0.327 \pm 0.014}$	$\mathbf{0.480 \pm 0.076}$
Q-chunking: one critic, one horizon, one sample				
QC	QC-FQL, no clamp on BiGym	1	$0.083 \pm 0.017^{\S}$	0.292 ± 0.011
QC_{cal}	+ FQC's clamp	1	0.102 ± 0.007	0.302 ± 0.027
QC_{cm}	+ capacity-matched critic	1	0.117 ± 0.009	0.318 ± 0.060
QC_{cal}+replan 4	every coordinate re-planned every 4 steps	1	0.065 ± 0.013	$0.279 \pm 0.054^{\ddagger}$
QC_{cal}, $H{=}4$	retrained at horizon 4	1	—	0.060 ± 0.014
QC, $H{=}4$	retrained at horizon 4	1	—	0.047 ± 0.036
other two-horizon variants				
FQC-cadence	keeps the executor; no clamp, no χ	32	0.295 ± 0.028	0.370
FQC_{add}	additive fast head (Section 5.2)	32	0.308 ± 0.033	0.400 ± 0.042
$\text{FQC}_{\setminus\chi}$	no χ	32	0.340 ± 0.018	0.432 ± 0.047
complementary split	hand slow / arms slow	32	0.314 ± 0.026	0.482 ± 0.039
random partition	one fixed draw ($\lvert\mathcal{S}\rvert$-matched on DexJoCo)	32	0.292 ± 0.031	0.477 ± 0.062

open-loop split trains FQC's two critics and ranks the same candidates but executes the whole chunk open-loop (second line of (5)). The **partial re-plan** is its mirror: QC_{cal}+BoN's critic, clamp, and ranking with FQC's executor, which at the mid-chunk re-plan ranks full chunks with the one critic; its buffer stores the chunk proposed at the slow boundary (third line of (5)). With QC_{cal}+BoN and FQC, these two cross one or two critics with one or two execution horizons. $\textbf{FQC}_{\setminus\chi}$ drops only χ from the fast head. $\textbf{FQC}_{\text{add}}$ is an additive variant that differs from FQC in three places: its fast head does not read χ, its fast target is single-phase, and its mid-chunk bootstrap averages the values of two candidate continuations instead of using the one the executor plays; it is therefore not the χ ablation. **FQC-cadence** keeps FQC's executor without the clamp or χ. Single-horizon controls isolate capacity and a shorter shared horizon: $\textbf{QC}_{\text{cm}}$ has a four-member instead of a two-member critic ensemble, approximately matching FQC's two heads in parameter count, $\textbf{QC}_{\text{cal}}$**+replan** 4 re-plans every coordinate every 4 steps, and $H{=}4$ retrains at horizon 4. The **complementary split** swaps $\mathcal{S}$ and $\mathcal{F}$, and the **random partition** is one fixed draw. We also run **DQC** (Li et al., 2025b) at our budget (Section 5.4), and same-budget RLPD and IQL (Appendix E).

Protocol. Every cell is a 200-episode evaluation. Tables report the mean $\pm$ 1 s.d. over five training seeds (51001–51005) of the per-seed pooled mean: on DexJoCo the mean of the last five online evaluations, on BiGym the 600K checkpoint. Grey cells use seeds 51001–51003. We call the switch from offline to online training the handover.

5.3 Must the critics match execution?

The top block of Table 1 crosses one or two critics with one or two execution horizons; all four designs share the actor, the clamp, and the ranking of 32 candidates. Re-planning the fast factor pays off only with two critics: over the open-loop split it adds 0.122 on DexJoCo and 0.134 on BiGym, but over QC_{cal}+BoN, with one critic, $+0.006$ and -0.077. Splitting the critic without the re-plan does not help either: the open-loop split sits below QC_{cal}+BoN on both suites. The difference between the two re-planning gains is positive on all five seeds of both suites (paired t-test, $p = 0.013$ on DexJoCo and $p = 0.008$ on BiGym; Appendix C), as (5) predicts. On BiGym, deploying the QC_{cal}+BoN

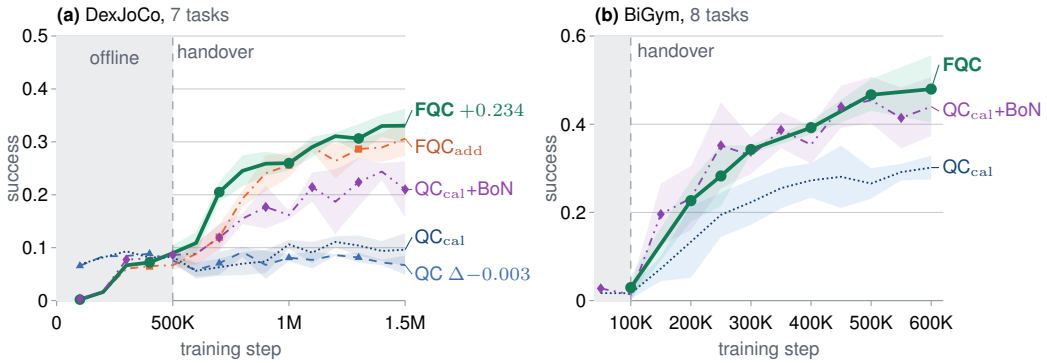


Figure 2: **The gap opens during online fine-tuning.** Pooled success during training; grey: offline pre-training; dashed rule: the handover. (a) DexJoCo, seven tasks: every agent starts online from a comparable offline checkpoint; QC and QC_{cal} are two batches of the same clamped agent; seeds 51001–51003. (b) BiGym, eight-task pool, five seeds (three for FQC and QC_{cal} before 600K). ± 1 s.d. bands; per task in Appendix B.

checkpoints with the partial re-plan already gives 0.369, so there most of the loss appears when the whole-chunk critic ranks re-planned candidates.

5.4 How does FQC compare with Q-chunking?

FQC improves on Q-chunking on both suites: 0.327 against 0.102 for QC_{cal} on DexJoCo, and 0.480 against 0.302 on BiGym (Table 1; learning curves in Figure 2; per task in Appendix B). On both suites every FQC seed scores above every QC_{cal} seed (exact two-sided Mann–Whitney test, $p = 0.008$, the smallest attainable with five seeds per arm), and the IQM intervals are disjoint (Appendix L); FQC's seed spread on BiGym is larger. Against the single critic given FQC's ranking (QC_{cal}+BoN) the margin holds on every DexJoCo seed and is within seed spread on BiGym (below). On DexJoCo the earlier batch of the same clamped agent, QC, reaches 0.083; the two batches show the run-to-run variation on this suite. Over all 39 BiGym tasks FQC is 0.227 and QC_{cal} 0.191; the other 31 tie (Appendix G).

The gain appears during online fine-tuning. On DexJoCo, QC leads for most of the offline phase (pooled 0.066–0.089) while FQC is still rising from 0.002, and the two arrive at the handover indistinguishable (FQC 0.090 ± 0.015, QC 0.076 ± 0.008). The million online steps then add $+0.234$ to FQC and -0.003 to QC, which falls below its own offline checkpoint on `click_mouse`, `pick_bucket` and `fold_glasses` while FQC shows no comparable drop (Appendix B). On BiGym, FQC likewise separates from QC_{cal} after the handover. Every curve carries the clamp, so the clamp does not produce the rise.

Ranking accounts for part of the gain. In the middle block of Table 1, every single-horizon variant that executes one student sample — long or short, clamped or not (unclamped only on BiGym), larger or not — sits near or below QC_{cal}. Extra critic capacity (QC_{cm}) does not produce the FQC margin; re-planning every coordinate every 4 steps (eval-only on BiGym, trained on DexJoCo) stays with the single-sample variants; and retraining at horizon 4 collapses the BiGym pool, since one short horizon for every coordinate backs up over fewer steps. Giving the single critic FQC's ranking is the largest single step on both suites: QC_{cal}+BoN reaches 0.231 on DexJoCo and 0.440 on BiGym.

Two execution-matched horizons add to ranking on DexJoCo. With the same ranking, FQC stays above QC_{cal}+BoN on all five DexJoCo seeds, by more than the two batches of the clamped single critic differ, and every variant in the bottom block does so seed for seed (FQC-cadence, without the clamp, on four of five). In Figure 2(a) QC_{cal}+BoN also climbs online, and FQC pulls away from it after the first online evaluation. On BiGym FQC and the complementary and random splits sit above it within seed spread, with FQC ahead on three of five seeds, while FQC_{add} and $\text{FQC}_{\setminus\chi}$ do not; there the effect of two horizons depends on the task rather than on the split: on

dishwasherclosetrays FQC and the complementary and random splits all lead it by a wide margin, and on removesandwich all three trail it.

Other methods. DQC (Li et al., 2025b) re-plans every coordinate together on the shorter horizon with its own critic and value losses, and ranks 32 candidates. At this paper's budget it reaches 0.049 ± 0.034 on the BiGym pool (five seeds); that is not the budget it was designed for, and at its own offline-heavy budget we have one seed on four tasks (Appendix D). The equal-budget test of re-planning every coordinate is QC_{cal}+replan 4. Same-budget RLPD and IQL sit below QC on both suites (Appendix E).

5.5 How do the split and χ affect performance?

Which factor is slow. The complementary runs swap $\mathcal{S}$ and $\mathcal{F}$ and change nothing else; the random partition is one fixed draw, $|\mathcal{S}|$-matched on DexJoCo. On DexJoCo the default sits slightly above the complementary swap, which sits above the random split, and every assignment is still carried by click_mouse, water_plant and hammer_nail. On BiGym the three assignments are within seed spread of one another and of QC_{cal}+BoN. On DexJoCo all three stay above it seed for seed, consistent with the gain coming from two execution-matched horizons rather than from which part is slow; the embodiment default adds a little there. Per task, the default split leads on water_plant, hammer_nail and dishwasherclose; on click_mouse and dishwasherclosetrays, the tasks with the largest gains, the swap does at least as well. Each suite has one random draw, so this does not show that every split works.

The input χ. $\text{FQC}_{\setminus\chi}$ drops only χ from the fast head and keeps the two-phase target, the clamp, and the executor. At five seeds it is within one seed-s.d. of FQC on BiGym and ahead of it on DexJoCo, so χ is not resolved from the rest of the package. This agrees with the saved BiGym critics: the fast head's value moves with χ by a few percent of how much it moves with the fast block (Table 12; demonstration states, one checkpoint), so χ rarely changes the $\arg\max$. We therefore treat χ as optional; the headline FQC keeps it because Proposition 1(ii) is exact for it.

6 Discussion

We propose Factored Q-Chunking (FQC), which commits a slow part of the robot's action for the whole chunk, re-plans a fast part halfway through, and gives each part a critic that scores exactly the actions it executes, so every bootstrapped continuation is a decision the agent takes. With the same actor and value clamp, this improves online fine-tuning over Q-chunking on an arm with a dexterous hand (pooled success 0.102 to 0.327) and on a mobile base with two arms (0.302 to 0.480), and a two-by-two ablation shows that the critics must match execution: neither re-planning part of the chunk nor splitting the critic helps on its own. We argue for treating the commitment horizon of action chunking as a per-coordinate choice, with value learning matched to what is executed.

Several challenges remain. First, the split and the horizons are fixed before training: we test the embodiment split, its swap, and one random draw per suite (Section 5.5), and one pair of horizons, $H{=}8$ and $H_f{=}4$, which the two-phase χ target ties ($H{=}2H_f$). Learning the split, or choosing each factor's horizon per state as adaptive chunking does for the whole vector, is a natural next step. Second, the slow commitment has no termination condition, so tasks in which the base or the arm must react within H steps are outside its scope. Third, we ran FQC only with the QC-FQL actor and with ranking. Finally, the evidence is concentrated: the DexJoCo margin is carried by three of the seven pooled tasks, FQC does not lift the 26 jointly floored BiGym tasks (Appendix F), a sign test over tasks is not significant (Appendix L), two horizons add to ranking on DexJoCo but not measurably on BiGym, and at five seeds χ is not resolved from the rest of the package (Section 5.5).

AI USE STATEMENT

In this work we used generative AI tools (Cursor agents) to help implement the method and interpret results against the evaluation logs. We have not used generative AI tools to develop the theoretical framing, formulate the mathematical claims, write the proofs, generate synthetic datasets, or translate the manuscript; qualitative data analysis does not apply. Additionally, we used generative AI tools to draft and edit the paper, to create and modify the TikZ/pgfplots figures, to write and edit training, evaluation, and table-recompute code, and to search and format literature. Every reported number was recomputed from the raw evaluation logs; AI-assisted code and prose were checked by the authors before inclusion. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The algorithm is specified in Appendix A; the proofs are in Appendix J; shared hyperparameters are in Appendix O. The DexJoCo and BiGym task inventories are Appendix G. The supplementary material contains the anonymized training and evaluation code of both suites, the agent file and flags of every arm in Table 1, the evaluation protocol commands, and the scripts and per-cell table with which every printed number is recomputed from the raw evaluation logs. We do not include a public repository URL in this anonymous submission.

REFERENCES

Agarwal, R., Schwarzer, M., Castro, P. S., Courville, A., & Bellemare, M. G. (2021). *Deep Reinforcement Learning at the Edge of the Statistical Precipice*. NeurIPS 2021.

Bacon, P.-L., Harb, J., & Precup, D. (2017). *The Option-Critic Architecture*. AAAI Conference on Artificial Intelligence.

Ball, P. J., Smith, L., Kostrikov, I., & Levine, S. (2023). *Efficient Online Reinforcement Learning with Offline Data*. International Conference on Machine Learning (ICML).

Biedenkapp, A., Rajan, R., Hutter, F., & Lindauer, M. (2021). *TempoRL: Learning When to Act*. International Conference on Machine Learning (ICML).

Chen, Q., Zhao, J., Zhou, H., et al. (2026). *ACSAC: Adaptive Chunk Size Actor-Critic with Causal Transformer Q-Network*. arXiv:2605.11009.

Chernyadev, N., Backshall, N., Ma, X., Lu, Y., Seo, Y., & James, S. (2024). *BiGym: A Demo-Driven Mobile Bi-Manual Manipulation Benchmark*. Conference on Robot Learning (CoRL).

Chi, C., Feng, S., Du, Y., et al. (2023). *Diffusion Policy: Visuomotor Policy Learning via Action Diffusion*. Robotics: Science and Systems (RSS).

Durugkar, I. P., Rosenbaum, C., Dernbach, S., & Mahadevan, S. (2016). *Deep Reinforcement Learning with Macro-Actions*. arXiv:1606.04615.

Ghasemipour, S. K. S., Schuurmans, D., & Gu, S. S. (2021). *EMaQ: Expected-Max Q-Learning Operator for Simple Yet Effective Offline and Online RL*. International Conference on Machine Learning (ICML).

Gireesh, N., Ju, Y., & Wang, H. (2026). *Adaptive Q-Chunking for Offline-to-Online Reinforcement Learning*. arXiv:2605.05544.

Hansen-Estruch, P., Kostrikov, I., Janner, M., Kuba, J. G., & Levine, S. (2023). *IDQL: Implicit Q-Learning as an Actor-Critic Method with Diffusion Policies*. arXiv:2304.10573.

Kostrikov, I., Nair, A., & Levine, S. (2021). *Offline Reinforcement Learning with Implicit Q-Learning*. International Conference on Learning Representations (ICLR 2022).

Kumar, A., Zhou, A., Tucker, G., & Levine, S. (2020). *Conservative Q-Learning for Offline Reinforcement Learning*. Advances in Neural Information Processing Systems (NeurIPS).

Lee, J., Lee, B.-J., & Kim, K.-E. (2020). *Reinforcement Learning for Control with Multiple Frequencies*. Advances in Neural Information Processing Systems (NeurIPS).

Lee, J., Cho, G., Yi, H., Im, J., Park, G., & Kim, S. (2026). *Receding-Horizon Execution for Action Chunking in Offline-to-Online Reinforcement Learning*. DEMO Workshop 2026.

Li, Q., Zhou, Z., & Levine, S. (2025a). *Reinforcement Learning with Action Chunking*. Advances in Neural Information Processing Systems (NeurIPS).

Li, Q., Park, S., & Levine, S. (2025b). *Decoupled Q-Chunking*. International Conference on Learning Representations (ICLR 2026).

Lipman, Y., Chen, R. T. Q., Ben-Hamu, H., et al. (2023). *Flow Matching for Generative Modeling*. International Conference on Learning Representations (ICLR).

Metz, L., Ibarz, J., Jaitly, N., & Davidson, J. (2017). *Discrete Sequential Prediction of Continuous Actions for Deep RL*. arXiv:1705.05035.

Nagy, C. F. M., Celik, O., Gospodinov, E., Seligmann, F., Liao, W., Kaushik, A., & Neumann, G. (2026). *SEAR: Sample Efficient Action Chunking Reinforcement Learning*. arXiv:2603.01891.

Nair, A., Gupta, A., Dalal, M., & Levine, S. (2020). *AWAC: Accelerating Online Reinforcement Learning with Offline Datasets*. arXiv:2006.09359.

Nakamoto, M., Zhai, Y., Singh, A., Sobol Mark, M., Ma, Y., Finn, C., Kumar, A., & Levine, S. (2023). *Cal-QL: Calibrated Offline RL Pre-Training for Efficient Online Fine-Tuning*. NeurIPS 2023.

Park, S., Li, Q., & Levine, S. (2025a). *Flow Q-Learning*. International Conference on Machine Learning (ICML).

Park, S., Frans, K., Eysenbach, B., & Levine, S. (2025b). *OGBench: Benchmarking Offline Goal-Conditioned RL*. International Conference on Learning Representations (ICLR).

Rashid, T., Samvelyan, M., de Witt, C. S., Farquhar, G., Foerster, J., & Whiteson, S. (2018). *QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning*. International Conference on Machine Learning (ICML).

Ren, A. Z., Lidard, J., Ankile, L. L., et al. (2024). *Diffusion Policy Policy Optimization*. International Conference on Learning Representations (ICLR 2025).

Rohanimanesh, K., & Mahadevan, S. (2001). *Decision-Theoretic Planning with Concurrent Temporally Extended Actions*. Conference on Uncertainty in Artificial Intelligence (UAI).

Seo, Y., Uruç, J., & James, S. (2024). *Continuous Control with Coarse-to-fine Reinforcement Learning*. Conference on Robot Learning (CoRL).

Seo, Y., & Abbeel, P. (2025). *Coarse-to-fine Q-Network with Action Sequence for Data-Efficient Reinforcement Learning*. Advances in Neural Information Processing Systems (NeurIPS).

Seyde, T., Werner, P., Schwarting, W., Gilitschenski, I., Riedmiller, M., Rus, D., & Wulfmeier, M. (2023). *Solving Continuous Control via Q-learning*. International Conference on Learning Representations (ICLR).

Sferrazza, C., Huang, D.-M., Lin, X., Lee, Y., & Abbeel, P. (2024). *HumanoidBench: Simulated Humanoid Benchmark for Whole-Body Locomotion and Manipulation*. Robotics: Science and Systems (RSS).

Sharma, S., Lakshminarayanan, A. S., & Ravindran, B. (2017). *Learning to Repeat: Fine Grained Action Repetition for Deep Reinforcement Learning*. International Conference on Learning Representations (ICLR).

Shin, Y., Chae, J., Kim, S., et al. (2026). *Adaptive Action Chunking via Multi-Chunk Q Value Estimation*. arXiv:2605.10044.

Son, K., Kim, D., Kang, W. J., Hostallero, D., & Yi, Y. (2019). *QTRAN: Learning to Factorize with Transformation for Cooperative Multi-Agent Reinforcement Learning*. International Conference on Machine Learning (ICML).

Sunehag, P., Lever, G., Gruslys, A., et al. (2018). *Value-Decomposition Networks For Cooperative Multi-Agent Learning Based On Team Reward*. International Conference on Autonomous Agents and MultiAgent Systems (AAMAS).

Sutton, R. S., Precup, D., & Singh, S. (1999). *Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning*. Artificial Intelligence, 112(1–2), 181–211.

Tang, S., Makar, M., Sjoding, M., Doshi-Velez, F., & Wiens, J. (2022). *Leveraging Factored Action Spaces for Efficient Offline Reinforcement Learning in Healthcare*. NeurIPS 2022.

Tavakoli, A., Pardo, F., & Kormushev, P. (2018). *Action Branching Architectures for Deep Reinforcement Learning*. AAAI Conference on Artificial Intelligence.

Tian, D., Celik, O., & Neumann, G. (2025). *Chunking the Critic: A Transformer-Based Soft Actor-Critic with N-Step Returns*. International Conference on Learning Representations (ICLR 2026).

Wang, H., Zhao, W., Wang, X., Huang, S., Lin, H., Zheng, B., Xu, R., Wang, G., Mu, Y., Wang, H., Fan, L., Li, H., Zhang, Z., & Tan, T. (2026). *DexJoCo: A Benchmark and Toolkit for Task-Oriented Dexterous Manipulation on MuJoCo*. arXiv:2605.16257.

Yang, J., Zhu, B., Chen, J., & Jiang, Y.-G. (2026). *Actor-Critic for Continuous Action Chunks: A Reinforcement Learning Framework for Long-Horizon Robotic Manipulation with Sparse Reward*. AAAI Conference on Artificial Intelligence.

Zhao, T. Z., Kumar, V., Levine, S., & Finn, C. (2023). *Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware*. Robotics: Science and Systems (RSS).

A Implementation details

Networks. Each critic is the released Q-chunking value network: concatenate the observation with a flattened action block, then a $512^{\times 4}$ MLP with LayerNorm. Q_{slow} takes $\dim(s) + H|\mathcal{S}|$ inputs and Q_{fast} takes $\dim(s) + H_f|\mathcal{S}| + 1 + H_f|\mathcal{F}|$ (Figure 3): $\chi = a[H_f{:}H, \mathcal{S}]$ is always an $H_f \times |\mathcal{S}|$ slot, a scalar $c \in \{0, 1\}$ marks whether that slot is committed, and $a_{\mathcal{F}}$ is the H_f-step fast block. On BiGym ($\dim(s) = 70$, $|\mathcal{S}| = 4$, $|\mathcal{F}| = 12$) that is 102 and $70 + 16 + 1 + 48 = 135$; on DexJoCo `hammer_nail` ($\dim(s) = 38$, $|\mathcal{S}| = 6$, $|\mathcal{F}| = 16$) it is 86 and $38 + 24 + 1 + 64 = 127$. At a slow boundary $\chi = \varnothing$ is encoded by zero-filling the slot and setting $c = 0$; at a mid-chunk re-plan the slot holds the committed slow remainder and $c = 1$. This is one network of fixed width, not two nets and not the student's proposed slow tail. Neither head sees the other factor's columns, so neither has an action-gradient on them; the disjointness is structural. The flow teacher and one-step student are those of QC-FQL, trained with the same flow-matching and distillation losses, with the single critic's Q in the student objective replaced by J from (6):

$$\mathcal{L}_{\text{actor}}(\psi, \xi) = \mathcal{L}_{\text{flow-BC}}(\xi) + \alpha\, \mathcal{L}_{\text{distill}}(\psi, \xi) - w \cdot \mathbb{E}_{s,z}\big[J\big(s, \mu_\psi(s, z)\big)\big]\,, \tag{8}$$

with $\alpha = 100$ and $w = 2$.

Execution, more completely. Let $\mu_\psi(s, z) \in \mathbb{R}^{H \times D}$ denote the one-step student. At a slow boundary:

1. Draw 32 candidates from the student, score them by (6), and keep $a^\star$.
2. Commit $a^\star[0{:}H, \mathcal{S}]$. Execute $\big(a^\star[0{:}H_f, \mathcal{S}],\ a^\star[0{:}H_f, \mathcal{F}]\big)$ for H_f steps.
3. At s_{t+H_f}, draw 32 new candidates, freeze the committed slow remainder χ, and $\arg\max$ over $Q_{\text{fast}}(s, \chi, \cdot)$. Execute the new fast sub-block together with the remaining slow sub-block $a^\star[H_f{:}H, \mathcal{S}]$.

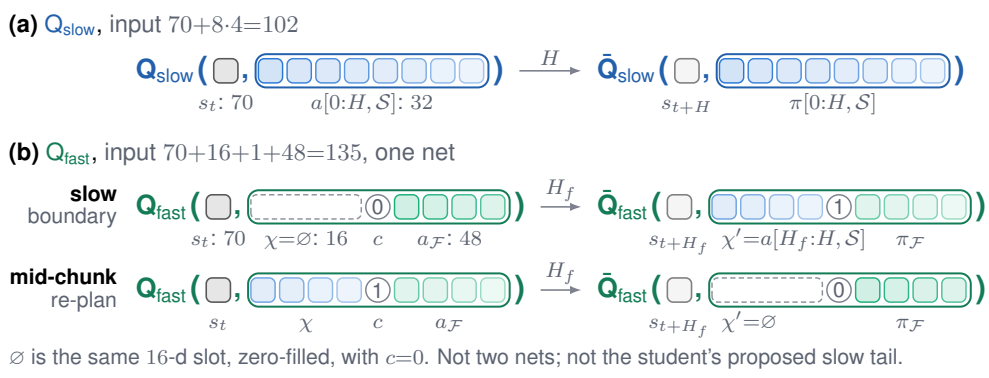


Figure 3: **What each critic receives, including the $t{+}\Delta$ bootstrap.** Same capsule language as Figure 1: each bead is one timestep. The released Q-chunking MLP concatenates s with a flattened action vector; factoring is a choice of which slice to hand it. Q_{slow} reads the slow factor over H (70+32=102). Q_{fast} always reads a fixed 70+16+1+48=135 vector: $\chi{=}a[H_f{:}H, \mathcal{S}]$ is a $4{\times}4$ slot, $c \in \{0, 1\}$ marks whether that slot is committed, and $a_{\mathcal{F}}$ is the H_f-step fast block. At a slow boundary $\chi{=}\varnothing$ is zeros and $c{=}0$; the target fills χ' with the committed slow tail. At a mid-chunk replan the slots swap. One network, one width.

The open-loop split is the same pipeline with step 3 deleted: the original $a^\star[H_f{:}H, \mathcal{F}]$ is executed open-loop, which is the unscored block in Figure 4 (Appendix C). QC_{cal}+BoN runs step 1 with its single critic in place of J, executes the whole chunk open-loop for H steps, and uses the same rule as π in its target. The partial re-plan runs all three steps with its single critic: step 1 as QC_{cal}+BoN, and in step 3 it overwrites each candidate's first $H{-}H_f$ slow rows with χ and ranks the full chunks by $Q(s, a[0{:}H])$; its buffer stores the chunk proposed in step 1. QC and QC_{cal} execute one student sample, as released.

B Per-task results

Table 2 and Table 3 give the per-task cells behind the pooled columns of Table 1.

The gain is concentrated. On DexJoCo the margin is carried by `click_mouse`, `water_plant` and `hammer_nail`; on the other four pooled tasks every agent sits near the floor, and QC is ahead on `bimanual_microwave_cook`. On BiGym FQC is ahead of QC_{cal} on five of the eight tasks, ties on `moveplate` and `flipsandwich`, and is behind only on `dishwasherloadcups`, near the floor.

task	**FQC**	FQC_{add}	QC_{cal}+BoN	partial re-plan	open-loop split	QC
`click_mouse`	$\mathbf{0.837 \pm 0.074}$	0.821 ± 0.087	0.576 ± 0.116	0.578 ± 0.100	0.509 ± 0.064	0.024 ± 0.036
`water_plant`	$\mathbf{0.759 \pm 0.061}$	0.679 ± 0.065	0.425 ± 0.122	0.516 ± 0.050	0.435 ± 0.060	0.425 ± 0.162
`hammer_nail`	$\mathbf{0.546 \pm 0.125}$	0.476 ± 0.156	0.399 ± 0.154	0.399 ± 0.063	0.342 ± 0.053	0.000 ± 0.000
`pick_bucket`	0.095 ± 0.024	0.096 ± 0.011	$\mathbf{0.118 \pm 0.018}$	0.096 ± 0.021	0.090 ± 0.017	0.023 ± 0.015
`pinch_tongs`	0.030 ± 0.020	0.056 ± 0.013	$\mathbf{0.069 \pm 0.029}$	0.043 ± 0.006	0.038 ± 0.016	0.028 ± 0.021
`fold_glasses`	0.012 ± 0.002	0.009 ± 0.004	$\mathbf{0.013 \pm 0.007}$	0.012 ± 0.004	0.009 ± 0.001	$\mathbf{0.013 \pm 0.007}$
`bimanual_microwave_cook`	0.012 ± 0.008	0.017 ± 0.004	0.020 ± 0.004	0.018 ± 0.007	0.013 ± 0.005	$\mathbf{0.068 \pm 0.048}$
pooled (7)	$\mathbf{0.327 \pm 0.014}$	0.308 ± 0.033	0.231 ± 0.038	0.237 ± 0.031	0.205 ± 0.015	0.083 ± 0.017

Table 2: DexJoCo (500K+ 1M), last-five online mean of 200-episode evals, mean $\pm$ s.d. over five seeds. QC is QC-FQL with the enlarged actor and FQC's clamp; on DexJoCo it is the same agent as QC_{cal}, and no unclamped QC was trained. QC_{cal}+BoN gives that clamped critic FQC's ranking of 32 candidates; the partial re-plan and the open-loop split complete the two-by-two of Table 1 (Appendix C). A second five-seed batch of the clamped agent (QC_{cal}, pooled 0.102) and the other variants are in Table 1.

DexJoCo. Giving the clamped single critic FQC's ranking of 32 candidates (QC_{cal}+BoN) recovers over half of the gap to FQC. Every seed's seven-task mean favours FQC and FQC_{add} over QC. The four unpooled tasks are at the floor for the three agents we ran on them (Appendix G). On

three pooled tasks QC's online success falls below its offline checkpoint (seeds 51001–51003: `click_mouse` 0.128 → 0.034, `pick_bucket` 0.070 → 0.021, `fold_glasses` 0.060 → 0.013; the `fold_glasses` offline value is from a separate offline-only run of the same configuration, and the run's own checkpoint gives the same picture), and FQC shows no comparable drop: on `click_mouse` both dip within the first 100K online steps (QC to 0.023, FQC 0.218 → 0.122), and of the two only FQC recovers, to 0.851. QC_{cal} (dotted in Figure 2(a)), a second batch of the same clamped agent, also stays far below FQC. Offline checkpoints at 100K–500K and every $\sim$ 100K online checkpoint are evaluated for 200 episodes on these seven tasks. Fifteen DexJoCo runs were trained twice with the same configuration, once on each of our two machines (FQC, FQC_{add} and QC on `hammer_nail`, `fold_glasses` and `bimanual_microwave_cook`, seeds 51003–51005). For each task we read every arm's five seeds from the one machine that holds five complete seeds for every arm on that task: for `hammer_nail` the machine that trained its first three seeds, and for the other two tasks the machine whose copies are complete. Reading the other complete copies instead moves no pooled mean by more than 0.006 and changes no comparison.

	dsTrays	dsClose	wcOpen	dtOpen	rmSand	flipSand	moveplate	loadCups	**pool (8)**
FQC	**0.784**	**0.376**	0.804	**1.000**	0.042	0.000	0.811	0.021	**0.480**
	±0.305	±0.279	±0.147	±0.000	±0.028	±0.000	±0.050	±0.021	±0.076
QC_{cal}+BoN	0.300	0.225	**0.859**	0.993	**0.254**	0.000	**0.859**	0.029	0.440
	±0.339	±0.394	±0.101	±0.007	±0.126	±0.000	±0.025	±0.024	±0.067
partial re-plan	0.343	0.318	0.361	**1.000**	0.024	0.000	0.846	0.008	0.362
	±0.390	±0.409	±0.130	±0.000	±0.030	±0.000	±0.054	±0.007	±0.076
open-loop split	0.649	0.201	0.127	0.973	0.076	0.000	0.738	0.003	0.346
	±0.289	±0.197	±0.134	±0.040	±0.148	±0.000	±0.036	±0.004	±0.055
QC_{cal}	0.095	0.034	0.479	0.903	0.015	0.000	0.815	**0.073**	0.302
	±0.111	±0.046	±0.239	±0.128	±0.018	±0.000	±0.057	±0.024	±0.027
QC	0.019	0.021	0.622	0.880	0.004	**0.001**	0.763	0.028	0.292
	±0.034	±0.029	±0.121	±0.179	±0.007	±0.002	±0.055	±0.023	±0.011

Table 3: BiGym eight-task pool at 600K; 200 episodes, mean with ± 1 s.d. over five seeds beneath. FQC and both QC_{cal} rows share the clamp; QC_{cal}+BoN adds FQC's ranking of 32 candidates, and the partial re-plan and the open-loop split complete the two-by-two (Appendix C); QC is the released QC-FQL agent, unclamped. Short labels abbreviate the task names of Table 9; per-task learning curves in Figure 5.

BiGym. Every BiGym cell uses the same evaluation protocol (Appendix K). Tasks on which every compared method stays near zero (26) or at or above 0.95 (5) are not pooled; Appendix G names them and gives the one-percent rule with its two borderline cases.

C CRITICS × EXECUTION HORIZONS

The four ranked designs in the top block of Table 1 share the actor, the clamp, and the ranking of 32 candidates, and differ only in the critic and the executor. QC_{cal}+BoN has one critic over the whole chunk and executes it open-loop. The open-loop split uses (2) with the same H-step executor: one decision, no mid-chunk re-plan, and ranking by J (each head's target draws its own candidates). The partial re-plan keeps the single critic and uses FQC's executor (Appendix A); its buffer stores the chunk proposed at the slow boundary, so its critic reads fast actions that were replaced. FQC matches both factors. The two mismatched designs are controls, not methods: each keeps one of FQC's two changes. In the open-loop split, the factored critics read FQC's two horizons; a factored critic matched to a single open-loop horizon, with both heads reading all H steps, is not run. Write H_k for the number of steps factor k executes between decisions and H_k^Q for the number of steps its critic reads. Table 4 is the resulting support, and Table 5 the paired contrasts.

Concretely, on a 16-D task the open-loop split executes

$$|\mathcal{F}| \cdot (H - H_f) \;=\; 12 \times 4 \;=\; 48 \quad \text{of} \quad H \cdot D = 128 \text{ coordinates} \tag{9}$$

that enter neither head (red block in Figure 4). Ranking therefore compares candidates on their first four arm steps only. Per-factor execution closes that gap.

method	factor	executes	critic reads	
QC_{cal}+BoN	all 16 coords	8	8	matched
open-loop split	base $\times 4$	8	8	matched
open-loop split	**arms** $\times 12$	8	4	$2\times$ **mismatch**
partial re-plan	base $\times 4$	8	8	matched
partial re-plan	**arms** $\times 12$	4+4	8	**reads replaced block**
FQC	base $\times 4$	8	8	matched
FQC	**arms** $\times 12$	4+4	4	matched

Table 4: Executor horizon versus critic horizon, per factor, on a 16-D BiGym task. In the open-loop split the fast critic is defined on H_f while the executor still runs H; in the partial re-plan the one critic reads all H fast steps of the proposed chunk, of which the last $H-H_f$ were replaced.

	DexJoCo (7)			BiGym (8)		
contrast	difference	positive	p	difference	positive	p
FQC − open-loop split	$+0.122$	$5/5$	0.0003	$+0.134$	$5/5$	0.020
partial re-plan − QC_{cal}+BoN	$+0.006$	$3/5$	0.83	-0.077	$0/5$	0.009
interaction (first minus second)	$+0.116$	$5/5$	0.013	$+0.211$	$5/5$	0.008
FQC − partial re-plan	$+0.090$	$5/5$	0.007	$+0.117$	$5/5$	0.009
open-loop split − QC_{cal}+BoN	-0.026	$1/5$	0.15	-0.094	$1/5$	0.093

Table 5: Paired contrasts in the two-by-two block of Table 1, on per-seed pooled means (five seeds). The first two rows are the gain from re-planning the fast factor with two critics and with one; "positive" counts seeds with a positive difference; p: two-sided paired t-test.

DexJoCo training of the mismatched designs. On DexJoCo the open-loop split and the partial re-plan were trained after the other arms, with the same flags, budget, and evaluation, on a revision of the training loop that tracks each environment's position in its chunk. Before that revision an environment that reset could execute up to seven actions left over from its previous episode, so the fix touches only data collection and, if anything, favours these two arms.

Deployment-only probe. On BiGym, deploying the 600K QC_{cal}+BoN checkpoints with the partial re-plan, without retraining, gives 0.369 ± 0.080 (200 episodes, five seeds), close to the trained partial re-plan (0.362): the whole-chunk critic already ranks re-planned candidates poorly at deployment.

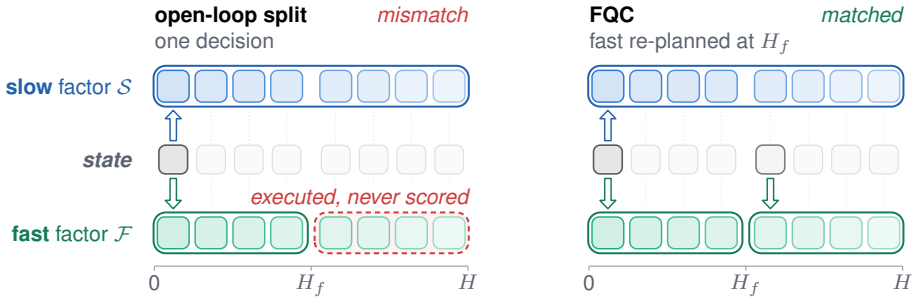


Figure 4: **Matched horizon.** The critic reads exactly what the executor runs. The open-loop split makes one decision. The slow factor is scored over the whole chunk; the fast critic scores only the first H_f. The dashed interval is executed and never scored. FQC opens a second fast decision and scores each H_f block over exactly the steps it executes. The slow factor stays matched to H. Hollow arrows mark decisions.

D MATCHED-BUDGET DQC

Decoupled Q-Chunking (Li et al., 2025b) (Algorithm 1: Q_ϕ MSE from V, Q^P expectile distill, V quantile from Q^P, BC flow on the short chunk, ranking 32 candidates under Q^P, $q_{\text{agg}}=\min$, replan every $h_a=4$) at the paper's BiGym budget 100K+ 500K, five seeds, 200 episodes, is 0.049 ± 0.034 on the eight-task pool. That is the matched-budget comparison; it is not the budget the method was

designed for, and it is not pooled with Table 1. The equalized receding-horizon control in the main ladder is QC_{cal}+replan 4.

On DexJoCo the same algorithm at 500K offline + 1M online, three seeds, 200-episode eval at 1.5M, is `click_mouse` 0.335/0.345/0.080 (mean 0.253), `hammer_nail` 0.110/0.100/0.250 (mean 0.153), `water_plant` 0.220/0.120/0.100 (mean 0.147); nine-cell mean 0.184. These cells are not pooled with Table 2.

Official budget. Table 6 reports the official 1M offline + 500K online protocol on `wallcupboardopen`, `moveplate`, `dishwasherclosetrays`, and `drawertopopen`, one seed (51001), 200 episodes, same h_a. These numbers are not pooled with the main tables.

task	1.5M final	best (step)
`wallcupboardopen`	0.010	0.010 (900K)
`moveplate`	0.000	0.010 (1.1M)
`dishwasherclosetrays`	0.480	0.595 (1.2M)
`drawertopopen`	0.090	0.090 (1.5M)
pooled (4)	0.145	0.176

Table 6: Official-budget DQC, seed 51001, 200 episodes, replan h_a=4. Not pooled with the main tables. Matched-budget five-seed DQC on the eight-task pool is 0.049.

The extra budget raises `dishwasherclosetrays` relative to the matched-budget five-seed mean (0.228) but does not recover `wallcupboardopen` or `moveplate`. The equalized eight-task receding-horizon analogue is QC_{cal}+replan 4 in Table 1 (0.279).

E SAME-BUDGET NON-QC NEIGHBORS

RLPD and IQL (Ball et al., 2023; Kostrikov et al., 2021) are public offline-to-online recipes at the paper's env-step budget, not FLOP-matched, and not QC variants. BiGym uses native open-loop H=8 evaluation (200 episodes at 600K); DexJoCo uses the last-five online mean of Table 2. Both sit below QC, so they do not explain the FQC–QC gap and are not columns of Tables 2 or 3.

suite	method	protocol	pool
BiGym (8)	RLPD	open-loop H=8, 200 ep	0.215 ± 0.055
	IQL	same	0.062 ± 0.017
	QC (released)	Table 3	0.292
DexJoCo (7)	RLPD	last-five online	0.002 ± 0.002
	IQL	same	0.001 ± 0.000
	QC	Table 2	0.083

Table 7: Same-budget non-QC neighbors, five seeds. Data and horizon matched; not FLOP-matched.

A collector-reset control that re-samples the committed slow remainder at every fast replan is harmful on the eight-task pool (0.451 vs. FQC 0.509 on seeds 51001–51003) and is not a method column of Table 1.

F FLOORED TASKS

Three tasks sit at the floor for every method — `pickbox` (0.000), `flipcup` ($\leq$ 0.003), `flipcutlery` ($\leq$ 0.005). Instrumented rollouts of the 600K checkpoints (3 tasks $\times$ 3 methods $\times$ 3 seeds $\times$ 50 episodes) put the cause well upstream of anything FQC changes. The gripper is in contact with the target object on 0.01–1.4% of steps, and in most episodes never touches it at all (Table 8).

Evaluating the offline endpoint of the eight-task pool (Figure 2(b), the 100K point) puts it near but not at the floor: five of the eight tasks are non-zero on at least one seed, and `dishwasherclosetrays` reaches 0.46 on one seed before any online interaction, while `drawertopopen` is exactly 0

	% of steps in contact			% of episodes ever in contact		
task	QC	open-loop	FQC†	QC	open-loop	FQC†
`flipcup`	1.37	0.63	1.20	15.3	14.7	24.7
`flipcutlery`	0.25	0.37	0.04	4.7	8.0	9.3
`pickbox`	0.01	0.11	0.18	1.3	2.7	3.3

Table 8: Contact on the floored tasks, 1350 episodes. † these rollouts predate the clamp of Section 4.3 and were collected on FQC-cadence; FQC in Table 9 scores 0.000, 0.003 and 0.000 on these three tasks, still at the floor.

there and 1.000 after online training. A behaviour-cloning bound therefore cannot distinguish "the demonstrations are the ceiling" from "solvable, but only online": essentially all of the pooled capability is created online, but not literally all of it. What would make these three tasks informative is reach-and-grasp (contact-conditioned shaping, more demonstrations, or a scripted grasp primitive). `pickbox` would then be the sharpest test of the fast head's blindness to the committed slow plan (Proposition 1(iii)), being the most strongly bimanual task in the suite.

The other 23 jointly floored BiGym tasks have no contact log; they are named in Table 9.

G TASK INVENTORIES

Relation to each benchmark's published numbers. Readers will want to compare our absolute rates against the numbers the two suites report, so we state why that comparison does not hold. BiGym (Chernyadev et al., 2024) reports demonstration-driven baselines averaged over its full task set, with imitation methods (ACT, Diffusion Policy) well above the reinforcement-learning ones (DrQ-v2, IQL, AWAC, CQN); DexJoCo (Wang et al., 2026) reports *only* imitation baselines and no reinforcement-learning baseline at all, with the strongest around 50% averaged over its suite. Three things differ from our setting: those numbers average over the whole benchmark while we pool a subset chosen by a floored/saturated rule; several of them are pixel-based while every agent here is state-based; and the imitation baselines are trained on far larger demonstration sets than the 2–64 episodes per task used here. Our absolute rates therefore measure value models against each other under one budget, not progress against those published systems, and we do not claim otherwise.

BiGym, 39 tasks. Table 9 is the 39-task sweep behind Table 3. The main text pools only the eight rows marked *pool*. Each algorithm has its own column. Pool rows of FQC, QC_{cal}, and released QC are the five-seed cells of Table 3; the eight pool cells of FQC_{add} are the same five seeds (eight-task mean 0.400 ± 0.042 in Table 1). Every other cell is seeds 51001–51003. Sat and floor FQC_{add} cells are the additive variant of Section 5.2. Released unclamped QC is complete on all 39 tasks. Averaging all 39 rows, FQC is 0.227 and QC_{cal} is 0.191. The five saturated tasks are at or near 1 for both, and the floor tasks near 0 for both, so the whole difference comes from the eight pool tasks.

DexJoCo. Table 10 is all eleven tasks. The main-text pool is the seven tasks that are not at the floor. On `bimanual_photograph`, FQC, FQC_{add} and QC were run under the protocol of Table 2 for seeds 51001–51003, and every evaluation of every run is 0. `bimanual_assembly`, `bimanual_hanoi`, and `bimanual_unlock_ipad` were not run under that protocol; their rows are three-seed runs of the same agents with an extended observation that adds dynamic object state, whose last-five means are at most 0.001. The other variants of Table 1 were not run on these four tasks. With these rows counted as 0, an eleven-task mean is seven-elevenths of the corresponding seven-task mean.

task	FQC	FQC_{add}	QC_{cal}	QC	status
dishwasherclosetrays	0.784	0.516	0.095	0.019	pool
wallcupboardopen	0.804	0.754	0.479	0.622	pool
drawertopopen	1.000	1.000	0.903	0.880	pool
moveplate	0.811	0.684	0.815	0.763	pool
dishwasherclose	0.376	0.135	0.034	0.021	pool
removesandwich	0.042	0.083	0.015	0.004	pool
flipsandwich	0.000	0.018	0.000	0.001	pool
dishwasherloadcups	0.021	0.008	0.073	0.028	pool
drawersallclose	1.000	1.000	1.000	1.000	sat
drawertopclose	1.000	1.000	1.000	1.000	sat
reachtarget	1.000	1.000	1.000	1.000	sat
reachtargetdual	0.985	1.000	1.000	1.000	sat
wallcupboardclose	1.000	1.000	1.000	1.000	sat
cupboardscloseall	0.000	0.000	0.000	0.000	floor
cupboardsopenall	0.000	0.000	0.000	0.000	floor
dishwasherloadcutlery	0.000	0.000	0.000	0.000	floor
dishwasherloadplates	0.000	0.000	0.002	0.000	floor
dishwasheropen	0.000	0.000	0.000	0.002	floor
dishwasheropentrays	0.000	0.000	0.000	0.000	floor
dishwasherunloadcups	0.013	0.002	0.012	0.000	floor
dishwasherunloadcupslong	0.000	0.000	0.000	0.000	floor
dishwasherunloadcutlery	0.000	0.000	0.000	0.000	floor
dishwasherunloadcutlerylong	0.000	0.000	0.000	0.000	floor
dishwasherunloadplates	0.000	0.000	0.000	0.000	floor
dishwasherunloadplateslong	0.000	0.000	0.000	0.000	floor
drawersallopen	0.000	0.000	0.000	0.000	floor
flipcup	0.000	0.000	0.002	0.003	floor
flipcutlery	0.003	0.000	0.005	0.002	floor
groceriesstorelower	0.000	0.000	0.000	0.000	floor
groceriesstoreupper	0.000	0.000	0.000	0.000	floor
movetwoplates	0.003	0.002	0.005	0.005	floor
pickbox	0.000	0.000	0.000	0.000	floor
putcups	0.002	0.000	0.000	0.000	floor
saucepantohob	0.000	0.000	0.000	0.000	floor
stackblocks	0.000	0.000	0.000	0.000	floor
storebox	0.000	0.000	0.000	0.000	floor
storekitchenware	0.000	0.000	0.000	0.000	floor
takecups	0.000	0.000	0.000	0.000	floor
toastsandwich	0.000	0.000	0.000	0.000	floor

Table 9: All 39 BiGym tasks at 600K, 200 episodes. Pool rows of FQC, FQC_{add}, QC_{cal}, and released QC: five seeds; all other cells: seeds 51001–51003. One algorithm per column. FQC: χ-conditioned agent of Table 3. FQC_{add}: additive variant of Section 5.2 (eight-task mean 0.400). Status *pool* is the eight-task mean 0.480 vs. 0.302 (QC_{cal}); the 39-row means are 0.227 vs. 0.191. *sat* = every compared method ≥ 0.95. *floor* = every compared method ≤ 0.01, with two borderline cases: dishwasherunloadcups (FQC 0.013, QC_{cal} 0.012) is floored with the other unload tasks, and flipsandwich, above 0.01 for FQC_{add} only (0.018), is kept in the pool as a contact task of the eight. QC is released QC-FQL. QC_{cal}+BoN was run on the eight pool tasks only (Table 3).

task	**FQC**	FQC_{add}	QC	status
click_mouse	0.837	0.821	0.024	pool
water_plant	0.759	0.679	0.425	pool
hammer_nail	0.546	0.476	0.000	pool
pick_bucket	0.095	0.096	0.023	pool
pinch_tongs	0.030	0.056	0.028	pool
fold_glasses	0.012	0.009	0.013	pool
bimanual_microwave_cook	0.012	0.017	0.068	pool
bimanual_photograph	0	0	0	floor
bimanual_assembly	0^*	0^*	0^*	floor
bimanual_hanoi	0^*	0^*	0^*	floor
bimanual_unlock_ipad	0^*	0^*	0^*	floor

Table 10: All 11 DexJoCo tasks, last-five online mean. Pool rows: five seeds, the means of Table 2 (s.d. there). bimanual_photograph: seeds 51001–51003 under the protocol of Table 2; every evaluation is 0. *Seeds 51001–51003 with an extended (dynamic-object) observation; these tasks were not run with the stock observation. The other variants of Table 1 were not run on the four floor tasks.

H Displacement from behavior cloning

The actor (8) is the released QC-FQL flow-BC teacher distilled into a one-step student, shared by FQC and QC. On demonstration states that student sits 3–4% of action-norm from its teacher for both methods (Table 11). That displacement is the known behaviour of this FQL actor, not a factored-critic effect: the two methods match, and it does not explain Section 5.4, where the clamped single critic falls below its offline checkpoint on three tasks and FQC shows no comparable drop. The measurement is one seed on demonstration rather than on-policy states, and action-space distance is a proxy for behavioral difference.

	student − BC teacher (% of action norm)	score spread (median, rel.)	arg max vs. random (dist. to demo)
QC, `wallcupboardopen`	**4.2**%	0.037	+0.072
FQC-cadence, `wallcupboardopen`	**4.1**%	0.033	+0.032
FQC-cadence, `dishwasherclosetrays`	**3.1**%	0.009	+0.012

Table 11: Shared FQL one-step student vs. its BC teacher, demonstration states at 600K (seed 51001, 150 states). Both arms displace 3–4% of action-norm. Measured on FQC-cadence; QC uses the same actor class.

I Additional experimental figures

The pooled learning curves are Figure 2 in the main text; Figure 5 draws the same curves per task.

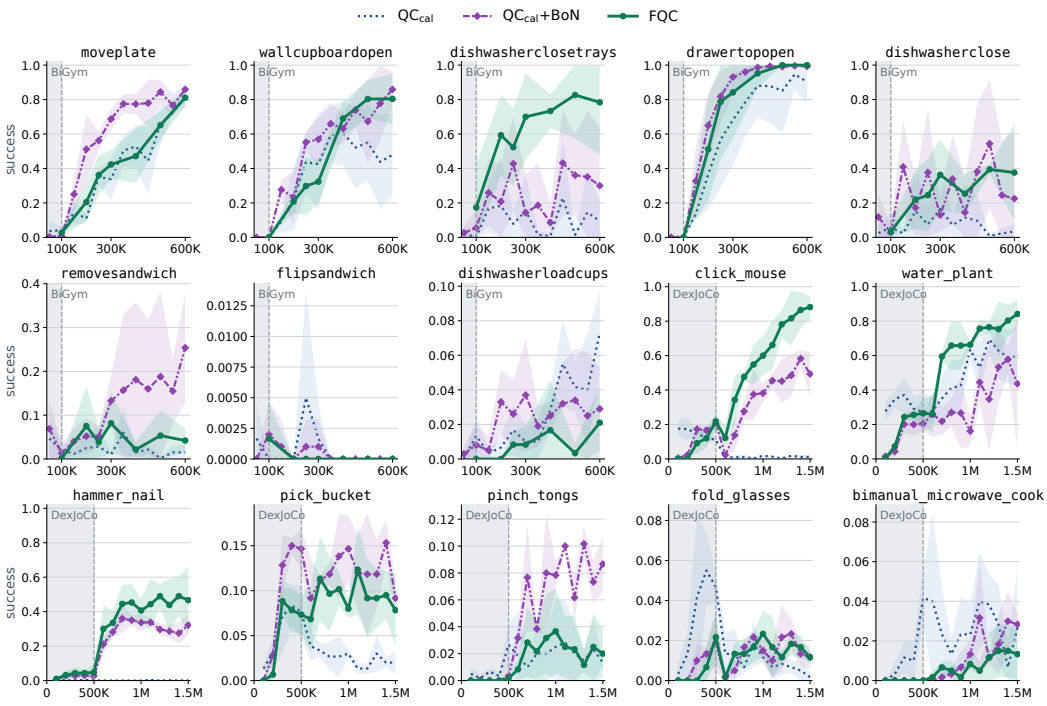


Figure 5: **Per-task learning curves.** QC$_{cal}$, QC$_{cal}$+BoN and FQC on the eight pooled BiGym tasks (first eight panels) and the seven pooled DexJoCo tasks (last seven), drawn as in Figure 2: mean over seeds with ±1 s.d. bands; grey: offline pre-training; dashed rule: the handover. BiGym: five seeds for QC$_{cal}$+BoN; FQC and QC$_{cal}$ three before 600K and five at 600K. DexJoCo: seeds 51001–51003. Each panel has its own vertical scale.

Figure 6 is a third-suite probe, not a result. At one seed and ten episodes per point the resolution is 0.1, so we report only the direction: on OGBench the factored value model wins the three navigation tasks that move and does not inherit QC's `cube-triple`. We plot only the two horizon-matched

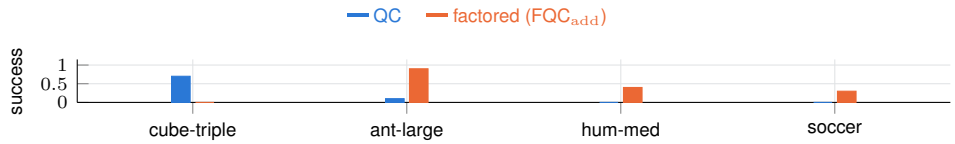


Figure 6: **OGBench, the tasks that move.** Official 500K+ 500K, **one seed**, last 10-episode evaluation at 1M (Park et al., 2025b). Both arms use the suite default $H{=}5$ ($H_f{=}2$); we plot only these two because they are the pair that is horizon-matched. The χ-conditioned agent cannot be compared here: its two-phase backup requires $H{=}2H_f$, so it would run at $H{=}4$, and the paper clamp is invalid on OGBench's dense negative returns. Jointly saturated and jointly floored tasks of the 19-task matrix are omitted. At ten episodes the resolution is 0.1, so this is a generality probe, not a measurement: read the direction, not the values. QC keeps `cube-triple`; the factored value model wins the three navigation tasks.

arms; the χ-conditioned agent would have to run at $H{=}4$ against the suite default $H{=}5$, and its clamp is invalid on OGBench's dense negative returns, so no χ comparison is available here. The factored agent also ranks 32 candidates while QC executes one sample, so this probe does not separate ranking from factoring.

J Proofs

J.1 Proof of Proposition 1

(i). Between consecutive decision epochs the system is driven by a fixed rule of the current augmented state and the decision. At a slow boundary ($\chi = \varnothing$) the decision is a full chunk $a \in \mathbb{R}^{H\times D}$; the executor plays $\big(a[0{:}H_f, \mathcal{S}], a[0{:}H_f, \mathcal{F}]\big)$ for H_f environment steps and stores $\chi' = a[H_f{:}H, \mathcal{S}]$. At a mid-chunk epoch the decision is a fast block $a_{\mathcal{F}} \in \mathbb{R}^{H_f\times|\mathcal{F}|}$; the executor plays $(\chi, a_{\mathcal{F}})$ for H_f steps and the next epoch is a slow boundary ($\chi' = \varnothing$). In both cases the distribution of the next augmented state depends only on $(\tilde{s}, \text{decision})$, through the environment kernel and this bookkeeping, so the decision-epoch process is an MDP over $\tilde{s}$ with an alternating epoch type readable from χ. The FQC executor selects each decision by a fixed rule of $\tilde{s}$ (candidates drawn from $\mu_\psi(\cdot \mid s)$, slow remainder frozen when $\chi \neq \varnothing$, ranked by the deployed score), hence it is a stationary policy on this MDP. □

(ii). At a mid-chunk epoch with state $\tilde{s} = (s, \chi)$, playing $a_{\mathcal{F}}$ yields the reward prefix $R_{t:t+H_f}$ and the next epoch state $(s_{t+H_f}, \varnothing)$, at which the deployed rule takes the decision whose fast block is $\pi(s_{t+H_f}, \varnothing)[0{:}H_f, \mathcal{F}]$. The one-epoch Bellman consistency condition for the deployed policy is therefore exactly (4) with the critic reading $(\tilde{s}, a_{\mathcal{F}})$: every executed fast coordinate is part of the decision, every executed slow coordinate is part of $\tilde{s}$ (the remainder χ), and the bootstrapped continuation is the decision actually taken at the next epoch. At a slow boundary the decision is the whole chunk and the fast head reads only its fast block, so there it fits the average of the target over the slow block selected with it, as in part (iii). Without per-factor execution no decision occurs at $t + H_f$, and the target evaluates a continuation the executor never plays (second line of (5)). □

(iii). This is the χ-omitted fast head (FQC$_{\setminus\chi}$ in Table 1), not FQC$_{\text{add}}$ (Section 5.2). Squared-loss regression of the targets on inputs $(s, a_{\mathcal{F}})$ converges, over the training distribution, to the conditional mean of the target given those inputs. Conditioning on $\tilde{s} = (s, \chi)$ refines this to $Q(\tilde{s}, a_{\mathcal{F}})$; omitting χ yields $Q^{\text{marg}}(s, a_{\mathcal{F}}) = \mathbb{E}\big[Q(\tilde{s}, a_{\mathcal{F}}) \mid s, a_{\mathcal{F}}\big]$ with χ drawn from its conditional distribution under the data-generating policy. For the regret bound, write $\varepsilon(a_{\mathcal{F}}) = \operatorname{osc}_\chi Q(s, \chi, a_{\mathcal{F}})$ over the support of $\chi \mid s$. Since $Q^{\text{marg}}(s, a_{\mathcal{F}})$ is an average of $Q(s, \cdot, a_{\mathcal{F}})$ over that support, $|Q(s, \chi, a_{\mathcal{F}}) - Q^{\text{marg}}(s, a_{\mathcal{F}})| \le \varepsilon(a_{\mathcal{F}})$ for every χ in the support. Let $\hat{a} = \arg\max Q^{\text{marg}}$ and $a^* = \arg\max_a Q(s, \chi, a)$ over the same candidate set. Then $Q(s, \chi, \hat{a}) \ge Q^{\text{marg}}(s, \hat{a}) - \varepsilon(\hat{a}) \ge Q^{\text{marg}}(s, a^*) - \varepsilon(\hat{a}) \ge Q(s, \chi, a^*) - \varepsilon(a^*) - \varepsilon(\hat{a}) \ge Q(s, \chi, a^*) - 2\max_a \varepsilon(a)$. □

Lemma 1 (Valid clipping is never a step away from the target)**.** *Let $V^\dagger$ be any value function satisfying the bounds pointwise, $\underline{v}(s') \le V^\dagger(s') \le 1$. Then (a) the target operator with the clipped bootstrap (7) remains a γ^{H_f}-contraction in $\|\cdot\|_\infty$; (b) for every s', $|\text{clip}(V(s'), \underline{v}(s'), 1) - V^\dagger(s')| \le |V(s') - V^\dagger(s')|$, so clipping never moves the bootstrap away from any value the bounds admit; and*

(c) on demonstration states the floor $\gamma^{k(s')}$ is the Monte-Carlo return of the demonstrating policy, which lower-bounds $V^\star$ under the sparse absorbing reward — the Cal-QL bound (Nakamoto et al., 2023) specialized to sparse success, so the bounds are valid whenever the demonstrations succeed.

Is the bound vacuous? A measurement. Under the paper's reward model $V^\star \in [0, 1]$, so the trivial bound on one-step selection regret is 1 and part (iii) says nothing unless the oscillation is far below 0.5. The quantity is cheap to evaluate from a saved critic, so we evaluate it. For each of 64 demonstration states we draw proposals from the deployed one-step student, read off the committed slow remainders $\chi^{(j)} = a^{(j)}[H_f{:}H, \mathcal{S}]$ and the fast blocks $a^{(k)}_{\mathcal{F}}$, and evaluate $Q_{\text{fast}}(s, \chi^{(j)}, a^{(k)}_{\mathcal{F}})$ on the product grid of remainders and fast blocks.

task (seed 51001 unless noted)	$\max_a \text{osc}_\chi\, Q$ (median)	bound $2 \max \text{osc}$	spread over $a_{\mathcal{F}}$ at fixed χ
`moveplate`	0.0007	0.0015	0.0154
`moveplate` (s51002)	0.0005	0.0011	0.0116
`wallcupboardopen`	0.0011	0.0022	0.0417
`dishwasherclosetrays`	0.0006	0.0012	0.0065
`drawertopopen`	0.0004	0.0008	0.0101
`dishwasherclose`	0.0007	0.0014	0.0119

Table 12: Measured oscillation of the χ-conditioned fast head, 600K checkpoints, 64 demonstration states. Computed from the saved 600K critics on 64 demonstration states.

Two readings, and we take both. The bound is *not* vacuous: at $2 \max \text{osc} \approx 10^{-3}$ it is three orders of magnitude below the trivial bound, so Proposition 1(iii) does constrain the χ-omitted head's selection regret here. But the same measurement says the constraint is loose in the direction that matters for us: the value moves 4–11% as much when the committed slow plan changes as when the fast block does, so conditioning on χ should rarely change the $\arg\max$. Dropping only χ is $\text{FQC}_{\setminus\chi}$ in Table 1. The measurement is over demonstration states with χ drawn from the deployed proposal rule rather than from a rollout distribution, so it is a proxy for the on-policy oscillation, not that quantity itself.

J.2 Proof of Lemma 1

(a). $x \mapsto \text{clip}(x, \ell, u)$ is 1-Lipschitz for any fixed $\ell \le u$, so for two bootstrap functions V_1, V_2 the clipped targets differ by at most $\gamma^{H_f} \|V_1 - V_2\|_\infty$ pointwise; the operator inherits the γ^{H_f} modulus of the unclipped one. □

(b). $\text{clip}(\cdot, \ell(s'), 1)$ is the projection onto the interval $[\ell(s'), 1]$, which by assumption contains $V^\dagger(s')$. Projection onto an interval can only decrease the distance to any point of the interval: if $V(s')$ lies inside, the clip is the identity; if outside, the clip moves $V(s')$ to the nearest endpoint, which lies weakly between $V(s')$ and $V^\dagger(s')$. □

(c). On a successful demonstration state s' with $k(s')$ steps to the absorbing $+1$ terminal, the demonstrating policy's Monte-Carlo return is exactly $\gamma^{k(s')}$, and $V^\star(s') \ge V^{\text{demo}}(s') = \gamma^{k(s')}$. The ceiling holds because the episode return is at most one terminal $+1$: $V^\star(s') \le 1$. Both bounds require the demonstration to actually succeed. □

K Provenance of the BiGym columns

Every BiGym cell in the main tables and this appendix uses the shared 200-episode protocol of Section 5.2: evaluation seeds 53000–53199, the 600K checkpoint, and one deployment setting per method.

In Figure 2(b), QC_{cal}+BoN and QC_{cal} are evaluated every 50K steps and FQC every 100K steps and at 250K. QC_{cal}+BoN has five seeds throughout; FQC and QC_{cal} have five at 600K, the cells of Table 1, and three before, because their seeds 51004 and 51005 kept only the 600K checkpoint. The tables add seeds 51004 and 51005 for FQC, $\text{FQC}_{\setminus\chi}$, FQC_{add} (0.400 ± 0.042), QC_{cal} (0.302 ± 0.027), QC_{cm}, released QC (0.292 ± 0.011), QC_{cal}+replan 4 (0.279 ± 0.054), the open-loop split (0.346 ± 0.055), the complementary and random partitions, and the two horizon-4 rows. QC_{cal}+BoN (0.440 ± 0.067) and the partial re-plan (0.362 ± 0.076) were run for seeds 51001–51005 on the eight pool tasks only.

FQC-cadence remains seeds 51001–51003. Every DexJoCo entry of Table 1 is last-five online on the same seven-task pool, five seeds.

L AGGREGATE STATISTICS

Five seeds resolve the differences the paper claims and not the ones it leaves open. Table 13 compares FQC with each arm on the per-seed pooled means of Table 15, treating each arm's five training seeds as independent samples. Against $\mathrm{QC}_{\mathrm{cal}}$ on both suites and against $\mathrm{QC}_{\mathrm{cal}}$+BoN on DexJoCo, every FQC seed scores above every seed of the other arm, which gives the smallest p an exact two-sided Mann–Whitney test can reach with five seeds per arm (0.008); two or three seeds per arm would already give 80% power at the observed difference. FQC's 4-point lead over $\mathrm{QC}_{\mathrm{cal}}$+BoN on BiGym would need about 50 seeds per arm, and its differences from the other two-horizon variants 11 to 30; the paper claims none of these. For a 10-point difference at the measured spread, 80% power needs about 2–3 seeds per arm on DexJoCo and 7–10 on BiGym.

suite	FQC vs.	difference	Welch p	Mann–Whitney p	seeds for 80% power
DexJoCo (7)	$\mathrm{QC}_{\mathrm{cal}}$	$+0.225$	$< 10^{-6}$	0.008	2
	$\mathrm{QC}_{\mathrm{cal}}$+BoN	$+0.096$	0.003	0.008	3
	$\mathrm{FQC}_{\mathrm{add}}$	$+0.020$	0.271	0.421	28
	$\mathrm{FQC}_{\setminus\chi}$	-0.012	0.277	0.310	30
BiGym (8)	$\mathrm{QC}_{\mathrm{cal}}$	$+0.178$	0.004	0.008	3
	$\mathrm{QC}_{\mathrm{cal}}$+BoN	$+0.040$	0.406	0.222	52
	$\mathrm{FQC}_{\mathrm{add}}$	$+0.080$	0.084	0.095	11
	$\mathrm{FQC}_{\setminus\chi}$	$+0.047$	0.278	0.151	30

Table 13: Two-sided tests of FQC against each arm on per-seed pooled means, five seeds per arm (Table 15). Last column: seeds per arm for 80% power at $\alpha = 0.05$, at the observed difference and the measured spread. Of these rows the paper claims only the three with Mann–Whitney $p = 0.008$; the two-by-two contrasts are paired over seeds (Table 5).

We also report the interquartile mean over all (task, seed) cells, i.e. the mean of the middle 50% of cells (DexJoCo has 35 cells, so the two boundary cells enter with weight $1/4$; BiGym's 40 cells need no weighting), with a stratified bootstrap interval (seeds resampled within task, 20k draws), which is the aggregation Agarwal et al. (2021) recommend for exactly this regime. The rliable implementation instead trims a whole number of cells (8 of 35 per side); on DexJoCo it gives FQC 0.254 $[0.227, 0.281]$, $\mathrm{FQC}_{\mathrm{add}}$ 0.234 $[0.200, 0.267]$ and QC 0.024 $[0.018, 0.035]$, which leaves every comparison below unchanged. On BiGym the two conventions coincide. Both pools are dominated by cells at the floor, which is why the IQM sits below the pooled mean.

suite	arm	pooled mean $\pm$ s.d.	IQM	95% CI on IQM
DexJoCo (7)	**FQC**	0.327 ± 0.014	0.244	$[0.215, 0.272]$
	$\mathrm{FQC}_{\mathrm{add}}$	0.308 ± 0.033	0.224	$[0.189, 0.260]$
	QC	0.083 ± 0.017	0.024	$[0.017, 0.034]$
	$\mathrm{QC}_{\mathrm{cal}}$	0.102 ± 0.007	0.018	$[0.015, 0.023]$
BiGym (8)	**FQC**	0.480 ± 0.076	0.466	$[0.389, 0.544]$
	$\mathrm{FQC}_{\setminus\chi}$	0.432 ± 0.047	0.378	$[0.327, 0.434]$
	$\mathrm{QC}_{\mathrm{cal}}$	0.302 ± 0.027	0.174	$[0.121, 0.225]$

Table 14: Interquartile mean (boundary cells fractionally weighted; see text) and stratified bootstrap intervals on the two headline pools, five seeds. FQC's interval is disjoint from QC and $\mathrm{QC}_{\mathrm{cal}}$ on DexJoCo and from $\mathrm{QC}_{\mathrm{cal}}$ on BiGym. It is *not* disjoint from $\mathrm{FQC}_{\mathrm{add}}$ on DexJoCo or from $\mathrm{FQC}_{\setminus\chi}$ on BiGym.

Paired over seeds, all five seeds' pooled means favour FQC over QC and $\mathrm{QC}_{\mathrm{cal}}$ on DexJoCo and over $\mathrm{QC}_{\mathrm{cal}}$ and $\mathrm{QC}_{\mathrm{cm}}$ on BiGym, with no per-seed reversal (Appendix M). Against $\mathrm{QC}_{\mathrm{cal}}$+BoN, FQC is ahead on all five DexJoCo seeds and on three of five BiGym seeds. A sign test over tasks is non-significant on both suites ($5/7$, $p = 0.45$ each), which is what one should expect: most tasks in each pool are at the floor for every method, so they carry no sign, and tasks are not independent

replicates of one effect. We report it so the reader does not have to guess. These numbers are computed from the same per-(task, seed) cells as Tables 2 and 15.

M PER-SEED POOLED MEANS

Table 15 lists each seed's pooled mean for the arms of Table 1 that have five seeds; the tables of the main text pool seeds 51001–51005. FQC is ahead of QC_{cal} and QC_{cm} on every seed of both suites, and of QC_{cal}+BoN on all five DexJoCo seeds and three of five BiGym seeds. It is ahead of $\text{FQC}_{\setminus\chi}$ on four of five BiGym seeds and one of five DexJoCo seeds, and of the complementary and random BiGym partitions on three and two of five.

suite	arm	51001	51002	51003	51004	51005	pooled 51001–51005
BiGym (8)	FQC	0.504	0.522	0.501	0.526	0.345	0.480 ± 0.076
	FQC_{add}	0.416	0.331	0.391	0.429	0.432	0.400 ± 0.042
	$\text{FQC}_{\setminus\chi}$	0.482	0.378	0.459	0.455	0.388	0.432 ± 0.047
	QC	0.308	0.299	0.280	0.288	0.286	0.292 ± 0.011
	QC_{cal}	0.331	0.298	0.321	0.297	0.262	0.302 ± 0.027
	QC_{cal}+replan 4	0.226	0.334	0.301	0.219	0.318	0.279 ± 0.054
	QC_{cm}	0.286	0.325	0.250	0.412	0.316	0.318 ± 0.060
	QC_{cal}+BoN	0.464	0.496	0.502	0.382	0.355	0.440 ± 0.067
	open-loop split	0.283	0.322	0.379	0.422	0.323	0.346 ± 0.055
	partial re-plan	0.334	0.428	0.446	0.344	0.259	0.362 ± 0.076
	complementary	0.441	0.522	0.512	0.440	0.494	0.482 ± 0.039
	random	0.403	0.421	0.527	0.537	0.496	0.477 ± 0.062
DexJoCo (7)	FQC	0.325	0.343	0.305	0.330	0.335	0.327 ± 0.014
	FQC_{add}	0.316	0.279	0.269	0.330	0.346	0.308 ± 0.033
	$\text{FQC}_{\setminus\chi}$	0.334	0.348	0.337	0.315	0.364	0.340 ± 0.018
	QC	0.064	0.080	0.074	0.088	0.109	0.083 ± 0.017
	QC_{cal}	0.093	0.105	0.108	0.096	0.109	0.102 ± 0.007
	QC_{cm}	0.100	0.120	0.118	0.123	0.122	0.117 ± 0.009
	QC_{cal}+BoN	0.248	0.201	0.191	0.232	0.285	0.231 ± 0.038
	open-loop split	0.197	0.192	0.214	0.195	0.227	0.205 ± 0.015
	partial re-plan	0.229	0.216	0.253	0.282	0.207	0.237 ± 0.031
	QC_{cal}+replan 4	0.076	0.081	0.057	0.056	0.055	0.065 ± 0.013
	FQC-cadence	0.331	0.288	0.304	0.297	0.254	0.295 ± 0.028
	complementary	0.281	0.301	0.313	0.331	0.347	0.314 ± 0.026
	random	0.285	0.268	0.260	0.332	0.316	0.292 ± 0.031

Table 15: Per-seed pooled means, 200-episode evals; BiGym at 600K, DexJoCo last-five online. The last column is the mean $\pm$ 1 s.d. reported in the main text. On DexJoCo, QC and QC_{cal} are two independent batches of the same clamped agent.

N RANKING BY ONE HEAD

On the saved 600K FQC critics of seeds 51001–51003 we change only the deployed ranking rule: at a slow boundary candidates are ranked by Q_{slow} alone or by Q_{fast} alone instead of J; the mid-chunk rule, the critics, and the actor are unchanged. Training still used J. Table 16 is the result: either head alone lands within about two points of J on the pool. Whether candidates are ranked at all matters more (QC_{cal} against QC_{cal}+BoN in Table 1).

task	Q_{fast} only	Q_{slow} only	J
`dishwasherclosetrays`	0.835	0.835	0.860
`wallcupboardopen`	0.853	0.823	0.883
`drawertopopen`	1.000	1.000	1.000
`moveplate`	0.815	0.802	0.817
`dishwasherclose`	0.355	0.473	0.452
`removesandwich`	0.025	0.025	0.042
`flipsandwich`	0.000	0.000	0.000
`dishwasherloadcups`	0.017	0.028	0.020
pooled (8)	0.487	0.498	0.509

Table 16: Deploy-time ranking rule on the BiGym pool, same trained FQC critics, seeds 51001–51003, 200 episodes.

O HYPERPARAMETERS

	value
discount γ	0.99
slow / QC chunk length H	8
fast chunk length H_f	4
flow-BC distillation weight α	100
value weight w in (8)	2
critic ensemble	2, mean aggregation
batch size	256
demo / replay mixture	50/50
offline + online steps (BiGym)	100K + 500K
offline + online steps (DexJoCo)	500K + 1M
training seeds (BiGym / DexJoCo)	5 / 5 (51001–51005; curve seeds in the figure captions)
evaluation episodes (BiGym)	200, seeds 53000–53199
evaluation episodes (DexJoCo)	200 per checkpoint (100K offline grid and $\sim$ 100K online); the table statistic is the last-five online mean
OGBench eval	10 episodes, seed 51001; plotted arms $H{=}5$ (suite default)
actor (DexJoCo only, both methods)	$1024^{\times 4}$ flow actor, 30 steps
critic MLP	$512^{\times 4}$ + LayerNorm
optimizer / actor-critic lrs	as in released QC-FQL
candidates ranked per decision N	32: FQC and its variants, open-loop split, partial re-plan, QC_{cal}+BoN, DQC; 1: QC, QC_{cal}, QC_{cm}, replan 4, $H{=}4$

Table 17: Hyperparameters shared with released QC-FQL unless noted. FQC does not tune a separate learning rate, ensemble size, or α.

HumanoidBench (Sferrazza et al., 2024) is a natural next embodiment — legs, torso, arms and hands offer more than two candidate subsets, and the action space is large enough to test more than two horizons and more than one random split — but it ships no demonstrations, while every component of this recipe (flow-BC teacher, distilled student, offline phase, balanced demonstration/replay sampling) presumes a demonstration set. Porting it is a second study.