

Graph-Rewighted Behavior Cloning with Human Interventions for Precise Robotic Manipulation

Anonymous Author(s)

Affiliation

Address

email

Abstract: Long-horizon precise manipulation produces mixed-quality data for post-training vision-language-action (VLA) models: even successful demonstrations and intervention-assisted rollouts may contain detours, hesitations, failed attempts, and imperfect recoveries. The key challenge is to identify which action chunks actually advance the task, rather than uniformly cloning or discarding imperfect trajectories. We propose Graph-Rewighted Behavior Cloning (GR-BC), an offline iterative post-training framework that builds a task-conditioned goal-reachability graph over demonstrations and iteratively collected rollouts, including both successful and failed episodes with human interventions. Reverse graph search computes minimal remaining steps to task success within the dataset support, and each action chunk is weighted by how much its execution reduces this potential. This consequence-based weighting reuses recoverable segments from failed trajectories, leverages interventions as recovery paths, and down-weights locally regressive chunks even inside successful trajectories. New rollouts are incorporated by appending them to the graph and rerunning search, without training a reward model, action-conditioned critic, or progress predictor. Across four precise manipulation tasks and three VLA backbones, GR-BC improves single-round supervised fine-tuning and reaches 91% success after a few offline iterations, without trajectory comparisons or progress labels.

Keywords: Vision-Language-Action Models, Human-in-the-Loop Learning, Robotic Manipulation

1 Introduction

Post-training vision-language-action (VLA) models [1, 2, 3, 4, 5, 6, 7, 8] for long-horizon precise manipulation rarely start with ideally clean expert datasets. Demonstrations collected by human experts can contain suboptimal or undesired behaviors, such as hesitations, overshoots, alignment detours, or recoveries from an earlier failed attempt. Intervention-assisted rollouts [9] are even more heterogeneous, mixing autonomous failures with delayed expert corrections and recoveries. Conversely, failure rollouts are not uniformly useless as they can also traverse goal-reachable states or contain short trajectory segments that steer the system back toward a viable path. Thus, the core challenge is essentially chunk-level credit assignment: whether an action chunk is worth imitating depends on how much task progress it induces when executed from the current state, rather than the final outcome of the entire trajectory. This motivates our central question: how can we effectively extract these progress-making action chunks from mixed-quality data, especially across iterative human-in-the-loop deployment rounds, to enhance sample efficiency for VLA post-training?

Existing post-training and progress estimation approaches struggle to provide this fine-grained signal within the mixed-quality data. Standard supervised fine-tuning (SFT) naively treats every action chunk as a positive target, thereby inadvertently cloning undesired behaviors from expert demonstrations or intervention-assisted rollouts. Many progress estimation methods implicitly assume that task progress evolves monotonically along successful trajectories, which hardly holds when trajectories contain correction behaviors or pauses. Although recent reward modeling approaches attempt

Submitted to the 10th Conference on Robot Learning (CoRL 2026). Do not distribute.

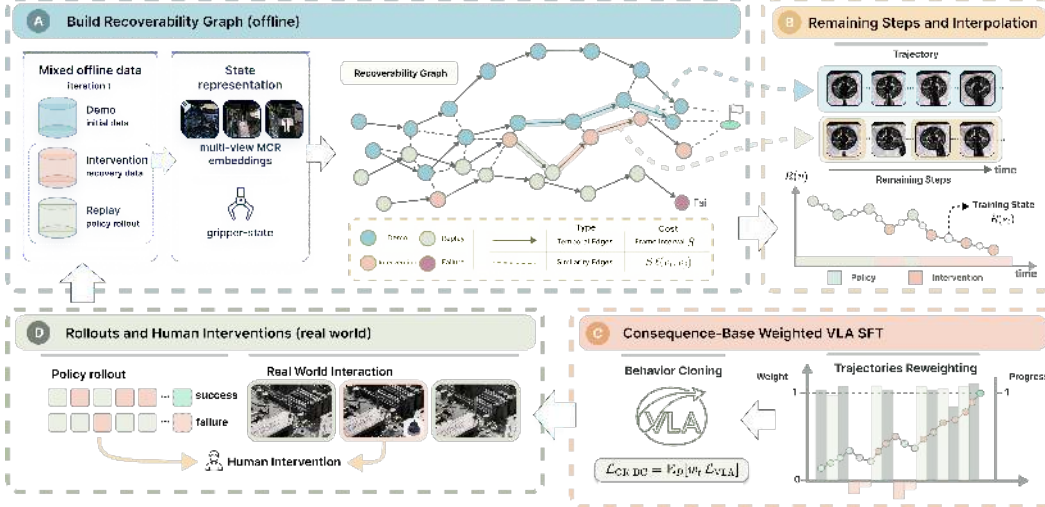


Figure 1: Overview of the GR-BC framework. (A) We construct a task-conditioned goal-reachability graph from mixed-quality offline data. (B) Reverse graph search estimates minimal remaining steps to task success, providing dense potential signals. (C) Action chunks are weighted by these consequence-based potentials for VLA SFT. (D) New real-world rollouts, including failed episodes and human interventions, are iteratively incorporated to refine the graph and policy.

address this by predicting chunk-level progress, they are inevitably bottlenecked by two limitations: (i) an over-reliance on heavy human or VLM annotations, such as dense progress labels, subtask boundaries, and trajectory comparisons [10, 11, 12, 13]; and (ii) the out-of-distribution (OOD) generalization gap inherent in large-scale pre-trained reward models, which often fail to provide reliable zero-shot progress signals when confronted with the precise visual nuances of unseen downstream tasks [14, 15, 16, 17]. While graph-search imitation [18] avoids learning a neural progress predictor, it uses graph search primarily to retrieve high-value neighboring actions for imitation, rather than to quantify how much the action chunk actually executed at a state moves the system toward success.

Our core insight is that the quality of an action chunk should be measured by its consequence: specifically, whether its execution reduces the minimal remaining steps to task completion. Crucially, this signal naturally penalizes local regressions inside successful trajectories, while simultaneously up-weighting valid progress-making chunks hidden within an overall failed rollouts.

To this end, we propose GR-BC, a graph-based framework for iterative offline VLA post-training over long-horizon precise manipulation tasks. GR-BC constructs a task-conditioned goal-reachability graph over demonstrations and intervention-assisted rollouts using multi-view MCR [19] embeddings to retrieve candidate visual neighbors and conservative similarity edges to connect dynamically compatible states. Instead of ranking neighbors or training a progress predictor, GR-BC runs reverse graph search to estimate minimal remaining steps and interpolates this potential to dense training states. The policy is then optimized only on physically executed action chunks, with each chunk’s contribution scaled by a precomputed weight proportional to its progress improvement. Because the graph is non-parametric, each offline iteration only appends new rollouts and reruns search, with no reward model, critic, progress predictor, architecture change, or supervised-loss modification.

The contributions are:

- **Graph-Retrieved Progress (GRP):** We propose a non-parametric progress estimation method that constructs a task-conditioned reachability graph and retrieves minimal remaining steps to success, avoiding reward models, critics, progress labels, or VLM annotations.
- **Graph-Reweighted Behavior Cloning (GR-BC):** We introduce a graph-reweighted behavior cloning objective that preserves the standard supervised VLA loss while reweighting each executed action chunk using GRP. This consequence-based reweighting reuses

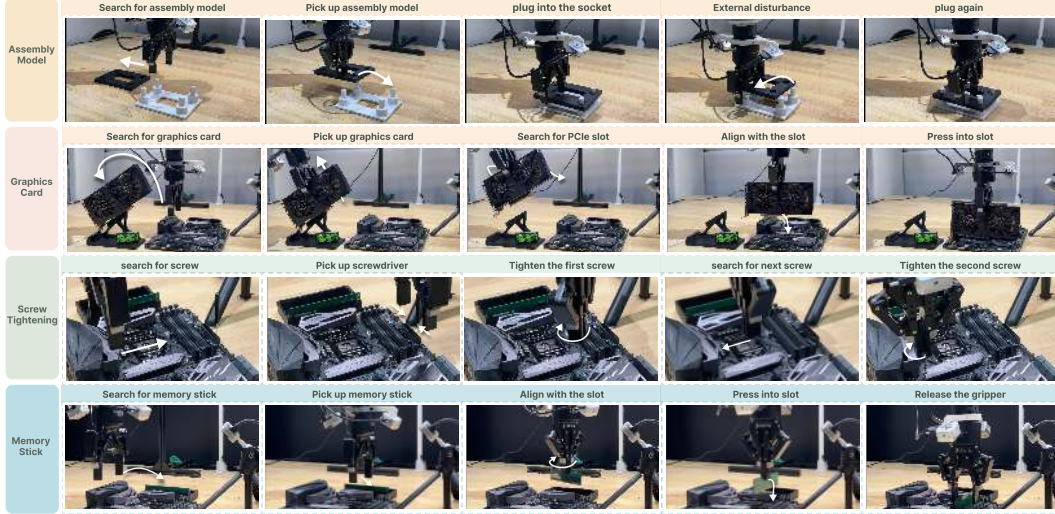


Figure 2: Real-world evaluation tasks. We evaluate GR-BC on four long-horizon precise manipulation tasks requiring precise visual alignment, contact-rich interactions, and recovery from execution errors.

- 71 recovery chunks from failed rollouts while suppressing regressive chunks in successful tra-
 72 jectories.
- 73 • **Iterative Offline Graph-Guided VLA Adaptation:** We validate GR-BC in human-in-the-
 74 loop offline iterative training by updating the reachability graph with intervention-assisted
 75 rollouts and reusing them in subsequent training rounds, achieving over 90% success on
 76 multiple long-horizon precise manipulation tasks with multiple VLA backbones.

77 2 Related Work

78 **Progress Rewards for Robotic Manipulation.** Long-horizon precise manipulation often relies
 79 on dense progress signals to provide supervision beyond sparse task success. Some methods infer
 80 progress from temporal order, goal proximity, or video ranking [20, 21, 22], while others learn stage-
 81 or subtask-aware progress models from segmented demonstrations or language annotations [10,
 82 23, 24]. More recent work uses foundation models or process reward models to estimate visual
 83 task progress from language, videos, or VLM judgments [14, 15, 25, 26, 27, 28, 29]. ARM [11]
 84 is closest to our motivation, replacing absolute progress scores with relative advantage labels for
 85 progressive, regressive, and stagnant behavior. However, these approaches typically require learned
 86 reward models, progress labels, stage annotations, VLM scoring, or temporal-order assumptions.
 87 GRP instead retrieves progress directly from graph reachability and reverse search, without learning
 88 a reward or progress function.

89 **Weighted Behavior Cloning for VLA Post-training.** Many RL-based VLA post-training meth-
 90 ods improve downstream task performance while preserving the generalization inherited from pre-
 91 trained VLA backbones through constrained updates [30, 31, 32, 33, 34]. Within this broader
 92 paradigm, weighted behavior cloning improves policy learning from heterogeneous data by assign-
 93 ing higher likelihood weights to actions estimated to be more valuable. Many weighted imitation
 94 and offline RL methods derive training weights from estimated values or advantages [35, 36, 37, 38].
 95 Recent robot and VLA post-training methods similarly reweight behavior cloning using reward, ad-
 96 vantage, intervention, or off-policy value estimates [10, 11, 39, 40, 41, 42]. Unlike prior weighted
 97 BC methods, GR-BC does not estimate action values, rewards, or advantages. Instead, it derives
 98 weights from the observed reduction in minimal remaining steps, directly measuring the conse-
 99 quence of executed action chunks.

3 Preliminaries

Problem Formulation. We study interactive imitation learning as an iterative offline VLA post-training procedure. Given an iteration round r , the mixed-quality data pool $\mathcal{D}^r$ is defined as:

$$\mathcal{D}^r = \mathcal{D}_{\text{demo}} \cup \mathcal{D}_{\text{rollout}}^{1:r}, \quad \mathcal{D}_{\text{rollout}}^{1:r} = \mathcal{D}_{\text{success-rollout}}^{1:r} \cup \mathcal{D}_{\text{fail-rollout}}^{1:r}. \quad (1)$$

Here, rollouts are labeled by their final task outcomes and may contain human interventions at arbitrary moments. Each trajectory $\tau = (s_0, a_0, \dots, s_n, a_n) \in \mathcal{D}^r$ is annotated with metadata including a language instruction g , the overall outcome, and intervention timestamps. The state s_t aggregates three RGB views (o_t^1, o_t^2, o_t^3), proprioception, and a binary gripper state q_t^{grip} . While s_t forms the basis for graph node construction, the VLA policy π_ϕ is optimized to predict future action chunks $a_{t:t+N}$ conditioned on (s_t, g) . As trajectory-level outcomes do not guarantee localized action optimality, our goal is to assign a fine-grained quality weight to each observed chunk.

Offline Weighted VLA Post-training. We use weighted supervised post-training over real-observed chunks:

$$\mathcal{L}(\phi) = \mathbb{E}_{(s_t, g, a_{t:t+N}) \sim \mathcal{D}^r} [w_t \mathcal{L}_{\text{VLA}}(\pi_\phi, s_t, g, a_{t:t+N})], \quad (2)$$

where $w_t \geq 0$ is a sample-level weight (recovering standard SFT when $w_t = 1$), and $\mathcal{L}_{\text{VLA}}$ denotes a standard imitation objective, such as behavioral cloning, denoising, or flow-matching [43]. GR-BC adopts this interface by deriving w_t from GRP; thus, the graph evaluates topological state potentials instead of actions (Fig. 1), leaving VLA training strictly supervised by the original dataset chunks.

4 Methodology

As illustrated in Fig. 1, GR-BC first builds a task-conditioned goal-reachability graph over mixed-quality trajectories (Sec. 4.1), retrieves graph-retrieved progress from reverse-search remaining-step potentials (Sec. 4.2), and uses the resulting progress improvement to reweight action chunks for iterative VLA post-training (Sec. 4.3).

4.1 Goal-reachability Graph Construction

To leverage recoverable segments within mixed-quality historical rollouts, we structure the cumulative offline dataset $\mathcal{D}^\nabla$ into a task-conditioned goal-reachability graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.

Node set $\mathcal{V}$: To maintain graph sparsity, trajectories are uniformly subsampled every k frames while strictly retaining all terminal and successful states: $\mathcal{V} = \{s_t \mid t = 0, k, 2k, \dots\} \cup \mathcal{V}_{\text{terminal}} \cup \mathcal{V}_{\text{success}}$. Each node $v_i \in \mathcal{V}$ aggregates synchronized multi-view visual embeddings $Z(v_i) = \{z_c(v_i)\}_{c=1}^3$, where $z_c(v_i) = f_{\text{MCR}}(o_c(v_i))$ via a frozen manipulation-centric encoder [19], as shown in Fig. 3.

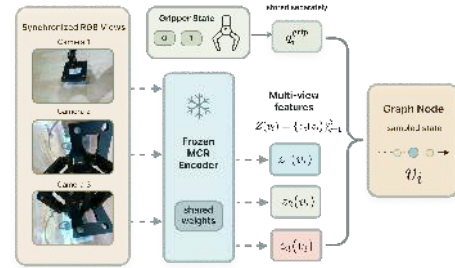


Figure 3: Multi-view node encoding for graph construction.

Crucially, the binary gripper state q_i^{grip} is maintained separately from $Z(v_i)$ to act as a hard structural filter during subsequent edge construction, preventing the soft alignment of dynamically incompatible states.

Temporal edges $\mathcal{E}$. To encode physically executed transitions, directed edges are established between consecutive intra-trajectory nodes as $v_i \rightarrow v_{i+k}$ with cost $c(v_i, v_{i+k}) = k$, ensuring that the cumulative path cost naturally quantifies the elapsed physical frames along the observed historical rollouts.

Cross-trajectory similarity edges $\mathcal{E}_{\text{sim}}$. To bridge independent trajectories and establish goal-reachability for failed or intervened rollouts, we introduce undirected similarity edges $\mathcal{E}_{\text{sim}}$ governed by multi-view consensus, gripper-state consistency, and local distance adaptability. We first construct a candidate pair set $\mathcal{C}_{\text{mv}}$ by intersecting the mutual K -nearest neighbors (K -NN)

145 queried within each camera view’s MCR feature space, effectively eliminating spurious single-view
 146 matches. For each candidate pair, the raw feature distance is normalized by a local resolution metric
 147 to ensure a consistent physical interpretation across the embedding space.

148 For camera c , let $\mathcal{N}(v)$ denote the immediate temporal neighbors of node v . We define the local
 149 scale as:

$$\sigma_c(v) = \frac{1}{|\mathcal{N}(v)|} \sum_{q \in \mathcal{N}(v)} \|z_c(v) - z_c(q)\|_2, \quad (3)$$

150 which captures how visually distant a single transition around v on the graph is.

151 As illustrated in Fig. 4, the multi-view adaptive edge length is then normalized by the maximum
 152 relative distance across all camera views:

$$\ell(v_i, v_j) = \max_{c \in \{1,2,3\}} \frac{\|z_c(v_i) - z_c(v_j)\|_2}{\epsilon_c + (\sigma_c(v_i) + \sigma_c(v_j))/2}, \quad (4)$$

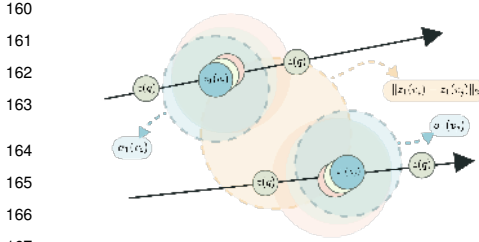
153 where ϵ_c is a small constant for numerical stability.

154 Combining these three criteria, the similarity edge set is formulated as

$$\mathcal{E}_{\text{sim}} = \{(v_i, v_j) \in C_{\text{mv}} \mid q_i^{\text{grip}} = q_j^{\text{grip}} \wedge \ell(v_i, v_j) \leq \alpha_{\text{edge}}\}, \quad (5)$$

155 with cost proportional to its adaptive length $c(v_i, v_j) = k \cdot \ell(v_i, v_j)$, where α_{edge} is a pre-defined
 156 threshold. Since temporal edges have cost k , this normalized similarity cost preserves the relative
 157 penalty of cross-trajectory bridges while still allowing failure and intervention states to reconnect to
 158 success paths when only similarity evidence suffices.

159 4.2 Remaining Steps and Interpolation



168 **Figure 4: Cross-trajectory similarity computation.** We compute the pair-wise feature
 169 distance between states v_i and v_j , normalized by their local adaptive scales $\sigma_c(v)$, to
 170 establish connectivity that reflects consistent
 171 physical state compatibility.

172 $\min_{\pi: v \rightarrow g_*} \sum_{e \in \pi} c(e)$. Nodes with no path to g_* are assigned $R_{\text{rec}}(v) = \infty$, resulting in a zero
 173 training weight. This constraint enforces a conservative data-reuse policy, filtering out ungrounded,
 174 suboptimal behaviors while salvaging locally successful segments.

177 **Dense Interpolation:** To bridge the gap between sparse graph nodes and the dense temporal super-
 178 vision required by VLA training, any state s_t situated between two adjacent trajectory nodes v_i and
 179 v_{i+k} (where $i \leq t \leq i+k$) is evaluated via linear interpolation:

$$\lambda_t = \frac{t-i}{k}, \quad \hat{R}_{\text{rec}}(s_t) = (1 - \lambda_t)R_{\text{rec}}(v_i) + \lambda_t R_{\text{rec}}(v_{i+k}). \quad (6)$$

180 This procedure identically extends to any future horizon state s_{t+N} , ensuring dense alignment for
 181 the chunk-level loss function detailed in Sec. 4.3.

182 4.3 Weighted SFT and Iterative Offline Update

183 Leveraging the dense remaining-step estimates, we formalize a chunk-level reweighting mechanism
 184 and establish the weighted SFT objective for iterative offline policy optimization.

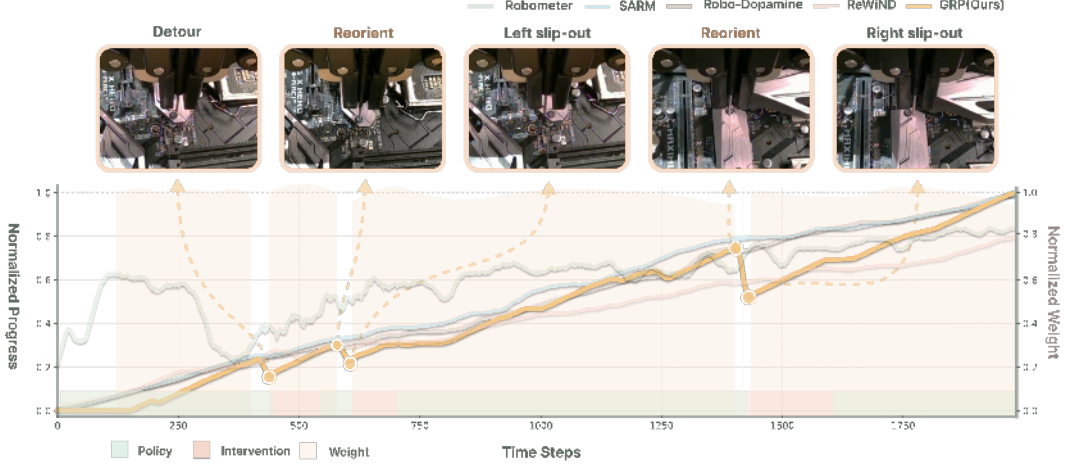


Figure 5: **Comparison of progress estimation and transition weights.** GRP (ours) provides more consistent progress signals than reward modeling baselines (SARM, Robometer, Robo-Dopamine, ReWiND). Our consequence-based weights effectively isolate task-advancing segments from local regressions or failed behaviors.

Chunk Consequence Evaluation: To eliminate task-dependent scale variances across trajectories of varying lengths, the dense potential $\hat{R}_{\text{rec}}(s)$ is first normalized into a bounded progress score $p(s) \in [0, 1]$:

$$p(s) = \text{clip} \left(1 - \frac{\hat{R}_{\text{rec}}(s)}{R_{\text{max}}}, 0, 1 \right), \quad (7)$$

where R_{max} denotes an upper percentile of finite R_{rec} values across the data pool to guard against extreme outliers. For an action chunk $a_{t:t+N-1}$ executed at state s_t , its trajectory impact over a lookahead horizon N is then evaluated via two complementary metrics: $\Delta_t = p(s_{t+N}) - p(s_t)$, $d_t = p(s_t) - \min_{\tau \in [t, t+N]} p(s_\tau)$, where the net progress Δ_t measures endpoint progress, and the regression depth d_t exposes transient intra-chunk violations. Explicitly tracking d_t prevents the policy from cloning unstable, highly sub behaviors that superficially terminate with a positive net progress Δ_t .

Consequence-Modulated Optimization: Chunks exhibiting stable forward progress are selectively prioritized via a step-wise transition weight w_t :

$$w_t = \begin{cases} \text{clip}(1 + \beta_w \tanh(\gamma \Delta_t), w_{\min}, w_{\max}), & d_t \leq \eta \wedge \Delta_t > \delta_{\min}, \\ 0, & \text{otherwise,} \end{cases} \quad (8)$$

where β_w and γ calibrate the progress bonus amplitude and sensitivity, bounded by $[w_{\min}, w_{\max}]$. Crucially, the zero-weight branch masks suboptimal gradients while retaining the corresponding states for graph connectivity. GR-BC then applies the weighted VLA objective in Sec. 3 using consequence-derived weights, with the compact form $\mathcal{L}_{\text{GR-BC}} = \mathbb{E}_{\mathcal{D}}[w_t \mathcal{L}_{\text{VLA}}]$, where w_t acts as a sample-level gradient multiplier without modifying the VLA architecture.

Iterative Offline Update: At round r , new rollouts $\mathcal{D}_{\text{rollout}}^r$ are integrated into the data pool. GR-BC then increments the graph with the new rollouts and re-executes the reverse Dijkstra search to refresh w_t for round $r + 1$. Crucially, this pipeline shifts the computational bottleneck from costly neural network retraining to efficient graph lookups.

5 Experiments

In this section, we answer three questions:

- **Q1:** Does GRP provide a more accurate progress signal for precise manipulation?
- **Q2:** Does GR-BC improve VLA post-training on mixed-quality human-in-the-loop rollouts?

210 • **Q3:** How do different graph-construction choices influence the quality of graph-retrieved
 211 progress?

212 We evaluate GR-BC on four real-world manipulation tasks shown in Fig. 2. Additional implemen-
 213 tation details are provided in Appendix A.

Table 1: Progress-model interfaces and evaluation results.

Method	$\Delta_{SF} \uparrow$	$\rho_{dev} \uparrow$	Class. SE $\uparrow$	Class. FE $\uparrow$	Multi-cam.	Failure data	Progress labels
ReWiND [29]	0.003	0	30/30	1/30	—	—	—
SARM [10]	0.24	0.48	26/30	14/30	—	—	✓
Robo-Dopamine [15]	0.29	0.59	29/30	8/30	✓	—	✓
RoboMeter [13]	0.159	0.69	30/30	6/30	—	✓	✓
GRP (ours)	0.97	0.92	30/30	30/30	✓	✓	—

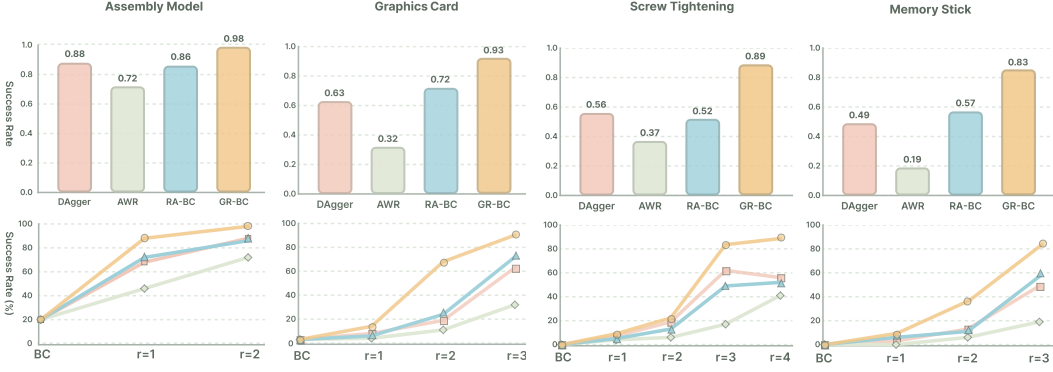


Figure 6: Policy improvement across offline post-training rounds. GR-BC achieves more consistent gains than DAgger, AWR, and RA-BC on precise assembly tasks.

214 5.1 Progress Estimation Evaluation

215 As visualized in Fig. 5, we evaluate GRP against representative progress estimation baselines, in-
 216 cluding ReWiND [29], SARM [10], Robo-Dopamine [15], and RoboMeter [13], on mixed-quality
 217 trajectories collected across policy-training iterations. Table 1 summarizes their data interfaces and
 218 evaluation results. Additional details of the progress-function evaluation protocol are provided in
 219 Appendix B.

220 **Evaluation protocol.** We evaluate each method under its intended supervision setting rather than
 221 enforcing a unified training protocol. Training and validation data are constructed according to
 222 each method’s original assumptions, while evaluation trajectories remain fixed. To avoid injecting
 223 additional subjective labels, we do not annotate any partial-success states or per-frame progress
 224 labels. Method-specific assumptions are summarized in Table 1. For each task, we evaluate 30
 225 successful (SE) and 30 failed (FE) episodes. Following the SARM classification protocol [10], we
 226 report trajectory-level classification results over successful and failed episodes. We also report the
 227 success–failure progress gap Δ_{SF} , computed as the difference between final normalized progress on
 228 SE and FE episodes. Finally, we select 100 manually localized deviation windows from successful
 229 trajectories. These windows capture locally regressive behaviors, including teleoperation mistakes
 230 and pre-intervention deviations. We define ρ_{dev} as the fraction of deviation windows whose start-
 231 to-end progress drop exceeds a threshold:

$$\rho_{dev} = \frac{1}{|\mathcal{W}_{dev}|} \sum_{w=(t_s, t_e) \in \mathcal{W}_{dev}} \mathbb{I}[p(s_{t_s}) - p(s_{t_e}) > \delta_{dev}], \quad (9)$$

232 where t_s and t_e denote the manually localized start and failure frame of window w , and δ_{dev} is the
 233 minimum progress drop required for detection. Higher ρ_{dev} indicates better sensitivity to locally
 234 regressive manipulation behavior.

5.2 Policy Improvement Evaluation

We use X-VLA [44] as the backbone for all policy-learning experiments, as its efficiency enables multiple offline post-training rounds. All methods are warm-started from the same BC checkpoint trained on successful demonstrations. We compare GR-BC with three baselines: **Dagger**, which continues supervised training on aggregated intervention rollouts [45]; **AWR**, which reweights chunks using action-level advantages following ALOE [35, 40]; and **RA-BC**, which reweights data using a reward model following SARM [10]. Additional details of the policy improvement evaluation are provided in Appendix C.

Policy improvement through iterative offline post-training. As shown in Fig. 6, GR-BC achieves the highest final success rate on all four tasks and improves more consistently across offline post-training rounds. Dagger benefits from additional intervention data but still clones autonomous errors and recovery actions uniformly. AWR is sensitive to inaccurate action-value estimates, which can overweight harmful chunks, while RA-BC is limited by the granularity of its learned progress model when local contact errors or misalignments occur. In contrast, GR-BC weights each executed chunk by its graph-retrieved progress change, suppressing regressive behaviors while prioritizing clean execution and recovery segments. (see Fig. 5 for signal-level evidence of this weighting capability).

5.3 Ablation Study

We ablate graph construction on the Assembly Model dataset to evaluate its effect on graph-retrieved progress.

Table 2 reports four diagnostics: failure reachability measures whether failed-rollout nodes can reach the success sink, zero-weight ratio measures how many chunks are suppressed, pre/normal compares weights on pre-intervention autonomous segments against normal segments, and mismatch edges count gripper-inconsistent similarity links. Temporal edges alone cannot connect failed rollouts to the success sink, while cross-trajectory similarity edges greatly improve failure reachability but may introduce incompatible shortcuts. Our gripper-consistent graph removes these mismatches while preserving similar reachability, indicating better graph precision without sacrificing useful recovery connections. We further provide a hyperparameter analysis in Appendix D.

Table 2: Graph ablation on the Assembly Model task.

Variant	Failure reach. $\uparrow$	Zero-wt. ratio $\downarrow$	Pre/normal $\downarrow$	Mismatch edges $\downarrow$
Temporal only	0.0%	62.3%	1.02	–
w/o gripper	97.8%	26.3%	0.46	15.8%
Ours	98.3%	25.0%	0.42	0.0%

6 Limitations

GR-BC is limited by the quality of its dataset-supported reachability graph. Incorrect visual neighbors can create shortcut paths and distort remaining-step estimates, while overly conservative edge filters may miss valid recovery bridges. The current gripper-consistency and multi-view checks reduce common mismatch cases, but they do not fully certify physical reachability. In addition, GR-BC has limited generalization to states outside the support of the collected data. Future work should study stronger reachability tests, adaptive graph updates, and broader deployment across more embodiments and task families.

7 Conclusion

We introduced GR-BC, an offline iterative framework for VLA post-training. By constructing a task-conditioned goal-reachability graph, we derive Graph-Retrieved Progress (GRP) to estimate minimal remaining steps, enabling consequence-based weighting for action chunks. This approach effectively prioritizes task-advancing segments while reusing recovery behaviors from failures. Empirical results confirm that GRP provides a robust, critic-free progress signal for VLA refinement in precise robotic manipulation.

References

- [1] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.
- [2] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023.
- [3] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, et al. π_0 : A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.
- [4] K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. R. Equi, C. Finn, N. Fusai, M. Y. Galliker, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In *9th Annual Conference on Robot Learning*, 2025.
- [5] J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.
- [6] A. Zhai, B. Liu, B. Fang, C. Cai, E. Ma, E. Yin, H. Wang, H. Zhou, J. Wang, L. Shi, et al. Igniting vlms toward the embodied space. *arXiv preprint arXiv:2509.11766*, 2025.
- [7] W. Wu, F. Lu, Y. Wang, S. Yang, S. Liu, F. Wang, Q. Zhu, H. Sun, Y. Wang, S. Ma, et al. A pragmatic vla foundation model. *arXiv preprint arXiv:2601.18692*, 2026.
- [8] Y. Zhang, J. Zhao, C. Fan, F. Yan, T. Li, H. Tang, S. Fu, X. Wu, Q. Weng, W. Zhang, et al. Prts: A primitive reasoning and tasking system via contrastive representations. *arXiv preprint arXiv:2604.27472*, 2026.
- [9] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer. Hg-dagger: Interactive imitation learning with human experts. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8077–8083. IEEE, 2019.
- [10] Q. Chen, J. Yu, M. Schwager, P. Abbeel, Y. Shentu, and P. Wu. Sarm: Stage-aware reward modeling for long horizon robot manipulation. *arXiv preprint arXiv:2509.25358*, 2025.
- [11] Y. Mao, Z. Yu, W. Mao, Y. Li, Q. Hu, Z. Lan, M. Zhu, and H. Chen. Arm: Advantage reward modeling for long-horizon manipulation. *arXiv preprint arXiv:2604.03037*, 2026.
- [12] Y. Feng, H. Shi, H. Li, X. Guo, Y. Wang, C. Zhang, J. Zhang, X. Zhang, J. Tang, and J. Zhang. Procvlm: Learning procedure-grounded progress rewards for robotic manipulation. *arXiv preprint arXiv:2605.08774*, 2026.
- [13] A. Liang, Y. Korkmaz, J. Zhang, M. Hwang, A. Anwar, S. Kaushik, A. Shah, A. S. Huang, L. Zettlemoyer, D. Fox, et al. Robometer: Scaling general-purpose robotic reward models via trajectory comparisons. *arXiv preprint arXiv:2603.02115*, 2026.
- [14] T. Lee, A. Wagenmaker, K. Pertsch, P. Liang, S. Levine, and C. Finn. Roboreward: General-purpose vision-language reward models for robotics. *arXiv preprint arXiv:2601.00675*, 2026.
- [15] H. Tan, S. Chen, Y. Xu, Z. Wang, Y. Ji, C. Chi, Y. Lyu, Z. Zhao, X. Chen, P. Co, et al. Robo-dopamine: General process reward modeling for high-precision robotic manipulation. *arXiv preprint arXiv:2512.23703*, 2025.
- [16] S. Chen, C. Harrison, Y.-C. Lee, A. J. Yang, Z. Ren, L. J. Ratliff, J. Duan, D. Fox, and R. Krishna. Topreward: Token probabilities as hidden zero-shot rewards for robotics. *arXiv preprint arXiv:2602.19313*, 2026.

- [17] Y. Wu, W. Yuan, A. Qi, V. Guizilini, J. Mao, and Y. Wang. Large reward models: Generalizable online robot reward generation with vision-language models. *arXiv preprint arXiv:2603.16065*, 2026.
- [18] Z.-H. Yin and P. Abbeel. Offline imitation learning through graph search and retrieval. *arXiv preprint arXiv:2407.15403*, 2024.
- [19] G. Jiang, Y. Sun, T. Huang, H. Li, Y. Liang, and H. Xu. Robots pre-train robots: Manipulation-centric robotic representation from large-scale robot datasets. In *International Conference on Learning Representations*, volume 2025, pages 81885–81905, 2025.
- [20] Z. Wu, W. Lian, V. Unhelkar, M. Tomizuka, and S. Schaal. Learning dense rewards for contact-rich manipulation tasks. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6214–6221. IEEE, 2021.
- [21] Y. Lee, A. Szot, S.-H. Sun, and J. J. Lim. Generalizable imitation learning from observation via inferring goal proximity. *Advances in neural information processing systems*, 34:16118–16130, 2021.
- [22] D. Yang, D. Tjia, J. Berg, D. Damen, P. Agrawal, and A. Gupta. Rank2reward: Learning shaped reward functions from passive video. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2806–2813. IEEE, 2024.
- [23] T. Mu, M. Liu, and H. Su. Drs: Learning reusable dense rewards for multi-stage tasks. *arXiv preprint arXiv:2404.16779*, 2024.
- [24] C. Kim, M. Heo, D. Lee, H. Lee, J. Shin, J. Lim, and K. Lee. Subtask-aware visual reward learning from segmented demonstrations. In *International Conference on Learning Representations*, volume 2025, pages 15142–15165, 2025.
- [25] K.-H. Hung, P.-C. Lo, J.-F. Yeh, H.-Y. Hsu, Y.-T. Chen, and W. Hsu. Victor: Learning hierarchical vision-instruction correlation rewards for long-horizon manipulation. In *International Conference on Learning Representations*, volume 2025, pages 17550–17577, 2025.
- [26] Y. J. Ma, V. Kumar, A. Zhang, O. Bastani, and D. Jayaraman. Liv: Language-image representations and rewards for robotic control. In *International Conference on Machine Learning*, pages 23301–23320. PMLR, 2023.
- [27] M. Alakuijala, R. McLean, I. Woungang, N. Farsad, S. Kaski, P. Marttinen, and K. Yuan. Video-language critic: Transferable reward functions for language-conditioned robotics. *arXiv preprint arXiv:2405.19988*, 2024.
- [28] Y. J. Ma, J. Hejna, C. Fu, D. Shah, J. Liang, Z. Xu, S. Kirmani, P. Xu, D. Driess, T. Xiao, et al. Vision language models are in-context value learners. In *International Conference on Learning Representations*, volume 2025, pages 33984–34009, 2025.
- [29] J. Zhang, Y. Luo, A. Anwar, S. A. Sontakke, J. J. Lim, J. Thomason, E. Biyik, and J. Zhang. Rewind: Language-guided rewards teach robot policies without new demonstrations. *arXiv preprint arXiv:2505.10911*, 2025.
- [30] A. Wagenmaker, M. Nakamoto, Y. Zhang, S. Park, W. Yagoub, A. Nagabandi, A. Gupta, and S. Levine. Steering your diffusion policy with latent space reinforcement learning. *arXiv preprint arXiv:2506.15799*, 2025.
- [31] W. Xiao, H. Lin, A. Peng, H. Xue, T. He, Y. Xie, F. Hu, J. Wu, Z. Luo, L. Fan, et al. Self-improving vision-language-action models with data generation via residual rl. *arXiv preprint arXiv:2511.00091*, 2025.

- [32] Y. Li, X. Ma, J. Xu, Y. Cui, Z. Cui, Z. Han, L. Huang, T. Kong, Y. Liu, H. Niu, et al. Gr-rl: Going dexterous and precise for long-horizon robotic manipulation. *arXiv preprint arXiv:2512.01801*, 2025.
- [33] Y. Chen, S. Tian, S. Liu, Y. Zhou, H. Li, and D. Zhao. Conrft: A reinforced fine-tuning method for vla models via consistency policy. *arXiv preprint arXiv:2502.05450*, 2025.
- [34] Y. Wang, X. Li, P. Xie, P. Yang, B. Nie, Y. Cai, Q. Zhang, C. Qu, J. Wu, J. Song, et al. Learning while deploying: Fleet-scale reinforcement learning for generalist robot policies. *arXiv preprint arXiv:2605.00416*, 2026.
- [35] X. B. Peng, A. Kumar, G. Zhang, and S. Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.
- [36] A. Nair, A. Gupta, M. Dalal, and S. Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.
- [37] I. Kostrikov, A. Nair, and S. Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.
- [38] X. Chen, Z. Zhou, Z. Wang, C. Wang, Y. Wu, and K. Ross. Bail: Best-action imitation learning for batch deep reinforcement learning. *Advances in Neural Information Processing Systems*, 33:18353–18363, 2020.
- [39] H. Liu, S. Nasiriany, L. Zhang, Z. Bao, and Y. Zhu. Robot learning on the job: Human-in-the-loop autonomy and learning during deployment. *The International Journal of Robotics Research*, 44(10-11):1727–1742, 2025.
- [40] R. Yang, H. Wang, C. Liu, X. Yan, Y. Wang, X. Du, S. Yue, Y. Liu, C. Zhang, L. Qi, et al. Aloe: Action-level off-policy evaluation for vision-language-action model post-training. *arXiv preprint arXiv:2602.12691*, 2026.
- [41] P. Intelligence, A. Amin, R. Aniceto, A. Balakrishna, K. Black, K. Conley, G. Connors, J. Darpinian, K. Dhabalia, J. DiCarlo, et al. $\pi_{0.6}^*$: A VLA that learns from experience. *arXiv preprint arXiv:2511.14759*, 2025.
- [42] Z. Su, W. Kong, H. Dong, and H. Dong. Ig-rft: An interaction-guided rl framework for vla models in long-horizon robotic manipulation. *arXiv preprint arXiv:2602.20715*, 2026.
- [43] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.
- [44] J. Zheng, J. Li, Z. Wang, D. Liu, X. Kang, Y. Feng, Y. Zheng, J. Zou, Y. Chen, J. Zeng, et al. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*, 2025.
- [45] S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.