VLMs can supervise **offline RL**, but their **feedback** must consider **sub-trajectories,** be **non-Markovian**, and be interpreted as a component in a **simple** algorithm—such as a weight in weighted regression—**rather than** as a **reward**.

OfflineRLAIF

Jacob Beck

Piloting VLM Feedback for RL via SFO

Motivation

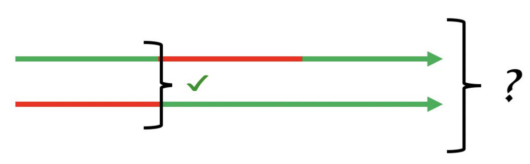
Vision Language Model (VLM) feedback
The absence of large-scale control data prevents training a general RL foundation model. Still, we can leverage existing VLMs for supervision.

Offline RL from AI Feedback (Offline RLAIF)
VLMs struggle to differentiate random trajectories at initialization. Offline RL can include trajectories that are easier for VLMs to differentiate.

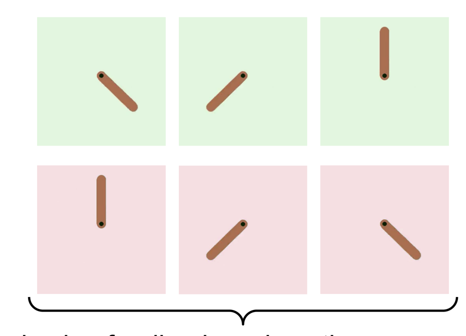
Challenges with Offline RLAIF
1) **Full-trajectory evaluation** exacerbates stitching issues
2) VLMs are not trained to **understand continuous control data**
3) **Feedback propagation** is unstable even with ground truth rewards

Conclusions

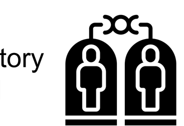
1) Sub-Trajectories Matter
Full-trajectory preferences decrease VLM calls, but are uninformative and worsen stitching issues, so sub-sampling trajectories is critical



2) Non-Markovian Feedback is Crucial
VLMs do not natively understand control data, so visual cues over time are needed to assess progress



3) Simplicity Outperforms Complexity
A filtered and weighted behavior cloning approach (SFBC) surpasses complex RL-based methods



Sub-Trajectory Filtered Behavioral Cloning (SFBC)

Existing work, such as **RL-VLM-F** (Wang et al., 2024) and **Clip-based rewards** (Baumli et al., 2023, Rocamonde et al., 2024), evaluates **online RL**, uses a **Markovian** reward, and investigates how to **elicit reward** from VLMs.

In contrast, **this study** evaluates **offline**, leverages **non-Markovian** feedback, and investigates **how best to use the feedback** (not just as reward).

1) We divide trajectories into disjoint and equal length **sub-trajectories**: $\tau_i = \left(s_{i\cdot k}, a_{i\cdot k}, s_{i\cdot k+1}, a_{i\cdot k+1}, \ldots, s_{(i+1)\cdot k}\right)$ with segment length k

2) We prompt an LLM to evaluate each sub-trajectory with a Markov and **non-Markov prompt**, and define the feedback as a combination:

$$\left.\begin{aligned} P_{\text{Markov}}(\tau_i) &= 1 - P(\text{“no”}|\text{Markov Prompt}) \\ P_{\text{Non-Markov}}(\tau_i) &= 1 - P(\text{“no”}|\text{Non-Markov Prompt}) \end{aligned}\right\} P_{VLM}(\tau_i) = \min\left(1, P_{\text{Markov}}(\tau_i) + P_{\text{Non-Markov}}(\tau_i)\right)$$

3) We **behaviorally clone** weighted sub-trajectories, and introduce *retrospective filtering*, assuming a failed sub-trajectory may result from preceding failure:

$$\mathcal{D}_{SFBC} = \{(s_t, a_t, \tau_i) \mid \tau_i \in \mathcal{D},\ (s_t, a_t) \in \tau_i,\ PVLM(\tau_i) \geq \alpha,\ P_{VLM}(\tau_{i+1}) \geq \alpha\} \quad \mathcal{L}_{SFBC} = -\mathbb{E}_{(s_t, a_t, \tau_i) \sim \mathcal{D}_{SFBC}}\left[P_{VLM}(\tau_i) \log \pi_\theta(a_t|s_t)\right]$$

Results

We evaluate on Pendulum-v1 across 15 seeds using GPT-4o. The dataset consists of 500 trajectories, with 300 steps from an expert policy and 300 from a failure policy, stitched in a random order. Sub-trajectory length (k) = 100. We subsample frames by 20x. Threshold (α) = 0.1.

Method	Success Rate (%)	Std. Error (%)	Mean Return	Std. Error
BC Naive	33	12	-4716	790
TD3+BC (GT)	27	11	-5131	814
VLM BC (Full-Trajectory)	13	9	-5234	578
AWAC (GT)	0	0	-7840	308
SF-BC (Ours)	**73**	11	**-1585**	518
VLM+TD3+BC	27	11	-5013	649
S-DPO	0	0	-6859	181
No Filtering	40	13	-4164	883
Markov Prompt Only	40	13	-4229	869
No Weighting	33	12	-3459	604
No Retrospective Filtering	13	9	-5562	525

↑ **Outperforms behavioral cloning**, both naively (BC Naive) and filtering by whole trajectories (VLM BC)
↑ **Outperforms offline RL with ground truth** (GT) reward

↑ **Outperforms** offline RL **with VLM as reward** (VLM+TD3+BC)
↑ **Outperforms** method **with VLM as preferences** (S-DPO)

Sub-Trajectory Filtered Optimization (**SFO**)

↓ Removing filtering or retrospective filtering decreases performance
↓ Removing non-Markov prompt decreases performance
↓ Removing weighting of trajectories decreases performance

Ablations

Github.com/
Jacooba/OfflineRLAIF

Reinforcement Learning Conference