Rank-Then-Act: Reward-Free Control from Frame-Order Progress

Yuriy Maksyuta **George Bredis** **Ruslan Rakhimov*** **Daniil Gavrilov**
T-Tech

Abstract

We introduce Rank-Then-Act (RTA), a framework for learning control policies from expert video demonstrations without environment rewards. RTA trains a Vision–Language Model (VLM) offline as a progress-based ordinal scorer, using a Group Relative Policy Optimization (GRPO) objective over shuffled frame sequences, which forces the model to recover temporal ordering from visual semantics rather than trivial time cues. Importantly, instead of using the scorer directly as a scalar reward model, we propose a correlation-based reward function for reinforcement learning: at each interaction window, we compute the Spearman rank correlation between predicted progress rankings and true temporal indices, yielding a bounded, scale-invariant learning signal. This design decouples reward learning from absolute calibration and enables stable transfer across tasks and environments. We evaluate RTA on discrete control benchmarks (PyBoy: Catrap, Kirby) and continuous control tasks (PointMaze, MetaWorld). RTA consistently matches or outperforms prior video-based reward learning methods and rank-based baselines, while demonstrating strong cross-task reuse of a single pretrained progress scorer. Our results suggest that correlation-structured supervision over video-derived ordinal signals is sufficient for policy learning, offering a scalable alternative to explicit reward design.

1 Introduction

Learning control from pixels *without* extrinsic rewards is a foundational challenge for generalist agents, particularly in domains where reward design is brittle, unavailable, or easily exploited-such as retro games, simulators with weak task APIs, and real-world robotics, where proxy rewards often lead to specification gaming. Training agents based on vision–language models (VLMs) as reward models offers a promising path: rather than relying on hand-crafted rewards, we can derive a dense notion of *progress* directly from expert video snippets and optimize policies with respect to this signal. However, three persistent obstacles remain: (i) chronological inputs invite trivial shortcuts (*later is better*), yielding vacuous, monotonically increasing progress scores; (ii) absolute progress scales are ambiguous across tasks and episodes; and (iii) online learning requires reward signals that are inexpensive to compute, sufficiently informative to guide exploration, and robust to distribution shift. We propose Rank-Then-Act (RTA), a framework for reinforcement learning from video in which supervision is defined over ordinal structure rather than scalar reward prediction.

In Stage 1, we train a Vision–Language Model (VLM) [11, 1] as a progressive ordinal estimator using Group Relative Policy Optimization (GRPO) [21] on shuffled video segments with anchor conditioning. This objective enforces recovery of temporal ordering from visual semantics while explicitly removing access to absolute frame positions, preventing trivial "later-is-better" solutions.

In Stage 2, we do not use the VLM as a scalar reward model. Instead, we define the reinforcement learning signal as a correlation functional over trajectories, computed as the Spearman rank correlation

*Correspondence: `ru.rakhimov@t-tech.dev`. Project page: `https://corl-team.github.io/rank-then-act`

Accepted at ICML 2026 Workshop on RLxF: Reinforcement Learning from World Feedback.

between predicted ordinal progress and true temporal indices within a sliding window of interaction. This produces a bounded, scale-invariant learning signal that depends only on agreement of ordering structure, rather than calibrated reward magnitude. This design enables policy learning driven entirely by progress inferred from expert demonstrations and environment interactions.

Existing approaches only partially address this setting. Video-based reward models and VLM reward heads typically require engineered goal text, per-task tuning, or produce uncalibrated scalar values that are hard to reuse. Reward-free RL and intrinsic motivation methods assume reward queries at deployment and do not capture progress from demonstrations. Most imitation-from-observation and video IRL methods require action labels or environment rewards, which are not assumed in our setting. RTA fills this gap by providing a simple, scale-free progress signal from expert videos, using only rank correlation with time as a reward for policy learning. Unlike prior video reward models that directly regress scalar progress or binary success signals, RTA uses a correlation objective over ordinal structure, making the reward invariant to scale, calibration drift, and cross-task shifts.

Our main contributions are as follows:

1. We introduce a simple yet effective Vision–Language Model (VLM) **progress scorer**, trained offline on shuffled expert video clips with a GRPO objective, taught to assign **progress ranks** on shuffled clips, and kept frozen during policy learning.
2. We propose a correlation-only reward signal for online control, computed as Spearman correlation between predicted **progress ranks** and time indices over sliding windows-enabling dense, shaped, reward-free feedback.
3. We demonstrate that this reward enables effective control across discrete and continuous benchmarks without environment rewards, outperforming strong video-based baselines.
4. We show that a single pretrained progress scorer transfers across tasks and environments, suggesting strong generality of ordinal video supervision.

RTA contributes a simple, correlation-only reward design that enables policies to learn in fully reward-free environments from video alone, and we provide direct comparisons to prior VLM reward models and ranking-based video rewards.

2 Related Work

Ordinal progress and ordering-based supervision from video. A growing line of work uses *ordering* as supervision to obtain shaped rewards from passive videos. Rank2Reward [26] similarly learns shaped rewards from passive video via temporal ranking. In our experiments, we adapt Rank2Reward as a non-VLM ranking baseline using the same game video data and compare its learned rewards and downstream control performance to RTA's correlation-only signal. VLM-RM [19] uses a clip-based progress scorer by comparing the goal completion frame with the current frame. In the original paper, it was shown that this approach often fails out-of-distribution and requires per-task environment tuning, such as changing textures or adjusting the view angle. Related efforts learn progress-sensitive embeddings or sequence rewards via temporal constraints and alignment [28, 25]. *RTA* is closest in spirit to [26] but differs in that (i) we fine-tune a compact LoRA *Vision–Language Model (VLM)* once with a *listwise* objective on *shuffled* clips (with *anchor conditioning*) to output per-frame *progress ranks*, and then *freeze* it; and (ii) we reward via *Spearman progress–time correlation* (correlation between predicted progress ranks and timestamps) over a *sliding window*, rather than learning a scalar progress head or using adversarial training.

Vision–language models as reward/value estimators. Pretrained VLMs have been used as zero-shot reward models by scoring observations against natural-language goals or ordering frames to estimate value [19], and as universal progress/value estimators by reframing value prediction as frame ordering over shuffled clips [16]. VLAC [29] train specific critic by using the same logic provided by GVL [16]. These approaches leverage world knowledge in VLMs to avoid task-specific reward engineering. *RTA* similarly capitalizes on a VLM prior but (i) fine-tunes it *offline* as an ordinal *progress scorer* on expert clips (outputting progress ranks) and (ii) converts its outputs into a correlation-based, sliding-window reward, enabling stable control from video alone without extrinsic environment rewards. We train the scorer with a *GRPO-style listwise preference objective* on *shuffled* clips; shuffling blocks "later-is-better" shortcuts and forces reliance on task-relevant visual cues.

Reward-free RL and intrinsic motivation. Classical reward-free RL formalizes a two-phase protocol: first explore without rewards, then solve downstream reward queries using the collected data [14, 15, 5]. In parallel, intrinsic-motivation methods (e.g., curiosity or prediction-error bonuses) provide task-agnostic signals that enable progress in sparse- or no-reward regimes [18]. These approaches emphasize state-space coverage or generic exploration rather than learning a task-specific progress signal from demonstrations. Our setting differs: we avoid extrinsic rewards and do not answer reward queries; instead, we optimize order consistency from demonstrations. *RTA* derives a demonstration-driven *ordinal* progress signal from expert video and optimizes *only* a progress–time correlation objective, without coverage objectives or reward queries.

Imitation from observation and IRL from video. When actions or rewards are unavailable, imitation-from-observation and IRL learn behaviors from state/video-only data. Adversarial formulations such as *Generative Adversarial Imitation Learning (GAIL)* and *Generative Adversarial Imitation from Observation (GAIfO)* learn policies by distribution matching without explicit reward engineering [10, 24]. Beyond frame-wise matching, video-based methods handle temporal misalignment and embodiment gaps via sequence-level objectives and cross-video temporal constraints, including XIRL [28], ordered-coverage alignment (ORCA) [12], alignments based on *Soft Dynamic Time Warping (Soft-DTW)* [25], and time alignment via video matching [9]. *RTA* departs from adversarial or regression-style reward learning by (i) training a listwise VLM progress scorer offline, then freezing it; and (ii) rewarding online control by the *Spearman* correlation between the scorer's progress ranks over a recent window and true timestamps-avoiding *both* adversarial training *and* scalar reward regression. Methods such as T-REX/D-REX [4, 3] and XIRL/ORCA [28, 12] operate in a different data regime: they require multiple demonstration trajectories of the same task or decompose the task into subgoals, whereas our setting assumes only raw videos with no actions or rewards. Adapting them would require either recovering actions from human play or changing the problem formulation. We therefore focus our empirical comparison on video-only reward models and ranking baselines (Rank2Reward, VLM-RM) that can be applied without action labels, and leave adapting action-dependent IRL methods to video-only VLMs for future work. VICtoR [13] is also not directly comparable to our setting, as it is a vision–instruction correlation method that relies on natural-language task descriptions and additional supervision (e.g., motion-level annotations and object/state signals).

3 Method

We present Rank-Then-Act (RTA), a two-stage framework that converts a model's grasp of frame order in expert videos into a bounded learning signal in $[-1, 1]$ for reward-free control. Stage 1 (Rank): fine-tune a VLM as a listwise progress scorer that assigns each frame a progress rank (larger = later), trained with GRPO [21] using Spearman's rank correlation between predicted scores and ground-truth timestamps as the reward. Stage 2 (Act): freeze the scorer and, during interaction, at query steps compute Spearman's ρ over a sliding window between the scorer's outputs and the window's timestamps; this correlation is the sole reward for policy-gradient training of the agent (no extrinsic environment rewards). We next formalize the control setting and define this correlation-based reward primitive.

3.1 Problem Setting and Correlation Primitive

We consider goal-conditioned, partially observed Markov decision processes with observation space $\mathcal{S}$, action space $\mathcal{A}$, transition function $\mathcal{P}$, episode horizon $\mathcal{T}$, and a textual goal g. The goal g is supplied as part of the agent's textual prompt to initialize the task. We do not, however, condition on g in the conventional RL sense; g is simply included in the prompt for goal-directed tasks. The agent $\pi_\theta : \mathcal{S} \to \Delta(\mathcal{A})$ maximizes expected return using an implicit, model-derived reward. When extrinsic rewards are absent, learning relies solely on an estimated measure of *progress consistency* produced by a model trained on expert video demonstrations.

Our sole scalar signal in both stages is the Spearman rank correlation,

$$\mathrm{spr}(x, y) \;:=\; \mathrm{Pearson}\big(\mathrm{rank}(x),\, \mathrm{rank}(y)\big) \in [-1, 1],$$

We use $\mathrm{spr}(\cdot, \cdot)$ both as the bounded, task-agnostic training objective in Stage 1 and as the reward signal in Stage 2. The idea of using rank correlation follows [16] as an estimator of trajectory correctness.

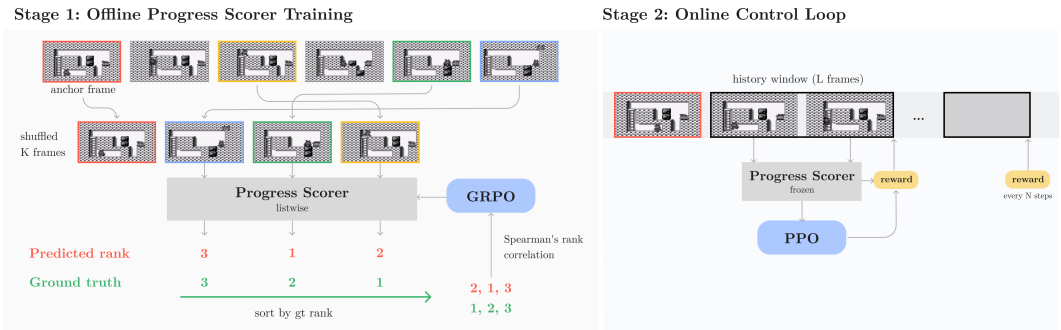

Figure 1: **RTA: reward-free control in two stages.** *Stage 1* trains a VLM *progress scorer* on shuffled clips with a GRPO objective that maximizes *progress–time* Spearman ρ. *Stage 2* uses the frozen scorer online: every N steps, we score a window (with L shuffles) and use the average $\rho \in [-1, 1]$ as the *only* reward for PPO-style learning.

3.2 Stage 1: Listwise Progress Scorer via GRPO

Expert clip batching. Given an expert video demonstration $\tau = (s_1, \ldots, s_T)$, we segment it into frames $(s_1, \ldots, s_{\tilde{K}})$, which we then use to construct training instances for progress prediction. Following [16], to avoid the shortcut "later is always better," we *anchor* the first frame and randomly shuffle the remaining frames:

$$
\begin{aligned}
&(s_{\text{anc}},\, s_{\tilde{2}}, \ldots, s_{\tilde{K}}),\\
&(\tilde{2}, \ldots, \tilde{K}) = \texttt{permute}(2{:}K),\\
&s_{\text{anc}} := s_1.
\end{aligned}
$$

Only the shuffled, non-anchor frames are scored; the anchor provides a fixed starting point for each training sequence. Fully shuffling frames can make the direction of progress ambiguous or overemphasize local appearance changes. Anchoring breaks this symmetry and encourages the model to evaluate true *progress* with respect to the task, rather than merely tracking local appearance drift. The same anchoring and shuffling procedure is applied at inference time.

VLM input and output. A vision–language model f_ϕ receives a sequence of K frames as input and is prompted to generate, for each (shuffled) frame excluding the anchor, a reasoning trace that specifies its predicted *progress rank* within the original sequence (higher = later). The canonical output format is:

$$
\begin{aligned}
&\texttt{Frame } i\texttt{:}\\
&\texttt{Frame Description: } \ldots\\
&\texttt{Rank: } p_i.
\end{aligned}
$$

Collecting these $K-1$ scalars yields the vector $\mathbf{p} = (p_2, \ldots, p_K)$, where each p_i denotes the predicted progress rank for frame i (anchor fixed at position 1 and never scored). All outputs are parsed using a strict regular expression. If parsing fails-for example, when the number of parsed scores does not equal $K-1$-we assign the minimum possible reward, $R_{\min} = -1$. For non-numeric or unparsable outputs, we treat parsing as failed. In case of ties equal ranks are permuted randomly.

Listwise reward and target. Let $\mathbf{p}$ denote the predicted *progress ranks* for the shown (shuffled) frames, and let $\mathbf{q}$ be their ground-truth temporal indices from original timestamps. We define the reward as

$$
R \;=\; \mathrm{spr}\big(\mathbf{p},\, \mathbf{q}\big), \tag{1}
$$

where $\mathrm{spr}(\cdot, \cdot)$ is Spearman's rank correlation. Using Spearman on continuous progress ranks makes the objective insensitive to the specific scale of p_i while rewarding correct monotone order.

GRPO objective and optimization. We treat the VLM's text generation process as a sequential policy over output tokens, and optimize a Group Relative Policy Optimization (GRPO) objective [21].

The GRPO loss is:

$$\frac{1}{|a|}\sum_{i=1}^{|a|}\min\Big(r_i\hat{A},\ \mathrm{clip}(r_i,1-\epsilon,1+\epsilon)\hat{A}\Big)$$

where a denotes the generated action (token) sequence, $r_i = \pi_\theta(a^i \mid s, a^{<i})/\pi_{\theta_{\text{old}}}(a^i \mid s, a^{<i})$ is the importance weight, ϵ is a clipping parameter, and $\hat{A}$ is the relative group advantage. This advantage $\hat{A}$ is defined as in Shao et al. [21], based on the maximized expected reward:

$$\max_{\phi}\ \mathbb{E}_{(\text{segment},\,\text{shuffle})}\ \mathbb{E}_{\text{output}\sim f_\phi}\,[\,R\,]$$

where R is the progress–time Spearman correlation reward defined above. After convergence, the progress scorer f_ϕ is frozen for use in Stage 2.

3.3 Stage 2: Online control from progress–time consistency

Inference window, anchor, and reward. At each environment step t, construct the window $\mathcal{W}_t = (s_{t-m+1}, \ldots, s_t)$ with $m = \min(N, t)$, and designate the oldest state s_{t-m+1} as the *reference anchor* s_{anc}. We query the frozen scorer only on *query steps* ($t \bmod N = 0$ or $t = \mathcal{T}$); otherwise we set $r_t = 0$. N is a hyperparameter, which introduces a tradeoff between reward frequency and computational efficiency, since VLM inference makes per-step reward computation prohibitive. On each query step, to reduce VLM cost and variance, draw L independent permutations of the $m-1$ non-anchor frames (keeping s_{anc} fixed), obtain progress ranks from the scorer for each permuted window, compute their progress–time correlations, and average these correlations to yield the scalar reward r_t. Unless stated otherwise, we use $L = 2$ and $N = 15$.

Policy optimization and normalization. We train the policy π_θ using policy gradient methods, using r_t as the sole reward signal (with no additional signals from the environment). The reward is bounded in $[-1, 1]$ by construction. Since rewards are sparse (computed every N steps) and bounded, we follow VL-DAC[2] use Generalized Advantage Estimation (GAE) based on the observed r_t sequence for VLM-based agents.

We use Qwen2.5-VL-7B as the backbone VLM for VL-DAC in all discrete environments. We choose a VLM backbone for game tasks because it can naturally handle combinations of actions described in text. This is important for the Kirby environment, which requires pressing multiple buttons at the same time, following the protocol of VideoGameBench [31].

To ensure that our results are driven by the reward design rather than biases of the VLM, we also evaluate a simple MLP backbone on Catrap games in combination with PPO [20]. These results are reported in Appendix A.5. For experiments with continuous control we use DrQv2 [27] as a policy. For experiments with LOOP [6], see Appendix A.1.

4 Experiments

We evaluate RTA as a reward-free approach to vision–language policy learning in game-like and continuous environments. Agents receive only expert video demonstrations; no extrinsic rewards, environment APIs, or task annotations are used.

The evaluation mirrors our two-stage method: (i) train and analyze a listwise VLM progress scorer that assigns each frame a progress rank; (ii) optimize a control policy using only the scorer's progress–time correlation as reward. This setup tests two hypotheses: VLMs can reliably infer ordinal progress from video, and these signals alone suffice to train agents to solve complex tasks.

Further details on the experimental setup, including data sources, task definitions, observation and action processing, and training and compute details, are provided in Appendix A.3.

4.1 Stage 1: Progress-Scorer Learning Dynamics and Generalization

Learning dynamics. We begin by verifying that a single VLM-based scorer can learn to score progress within shuffled expert clips across different levels, and we investigate its ability to generalize across levels, games, and domains. We run Stage 1 on both individual Catrap levels and for complete

games playthroughs. For almost all videos algorithm was able to converge to Spearman $\rho > 0.9$ in at most 300 steps. For more details refer to the appendix A.2.

Cross-level generalization. We train the progress scorer on a *source* set-either a single level i, a full-game playthrough, or a pool of 70 scraped GameBoy playthroughs-and evaluate on a disjoint *target* level j. Performance is the Spearman correlation between predicted progress and time (*progress–time* ρ). Figure 2 shows the mean validation ρ for all source–target pairs $(i \rightarrow j)$: diagonal cells (within-level) give upper bounds, while off-diagonals measure transfer.

Both per-level and pooled setups achieve high in-distribution ρ, however transfer is notably asymmetric: scorers trained on any level generalize especially well to level 1, likely due to its lower complexity. The strongest overall transfer comes from pooled training on all levels, indicating that a single, shared scorer can support domain-general inference-from full-game playthroughs to individual levels. Training on a diverse mix of 70 GameBoy playthroughs also yields solid zero-shot performance on held-out levels, despite limited direct exposure.

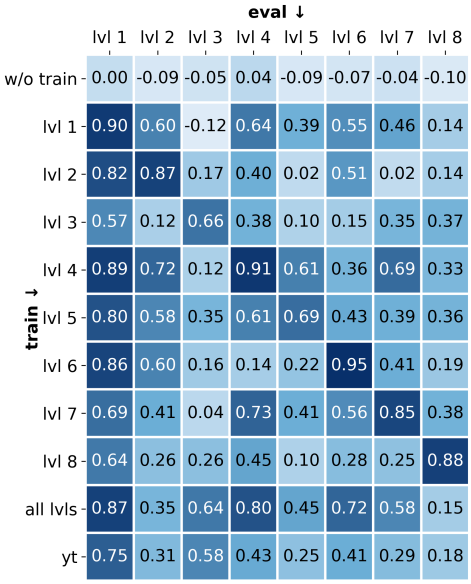

Figure 2: **Cross-level transfer of ordinal progress.** Heatmap of validation *progress–time* ρ for source (rows) $\rightarrow$ target (columns). Diagonals are in-level; off-diagonals show transfer. Pooled training (all levels or diverse games) transfers best; simpler Level 1 generalizes from most sources. VLM-based scorer trained on a single level (or YouTube dataset) recovers progress from shuffled sequences across different levels, showing higher correlation than an untrained model.

4.2 Stage 2: Control from progress–time consistency

We now deploy the frozen progress scorer in the environment and train a policy using policy-gradient methods, optimizing *only* the progress–time correlation reward r_t (see Sec. 3.3), with no access to any extrinsic task rewards.

GameBoy results. We conduct experiments on Catrap level 2, level 4, level 6, and the Kirby game. Levels 4 and 6 require advanced reasoning and backtracking from the model, as solving these levels involves overcoming dead ends. Kirby is a long-horizon control game. In the absence of explicit environment rewards, there are several approaches for current VLM and LLM-based control is to train Rank2Reward (which has not previously been applied to VLMs) and VLM-RM. Additionally, we compare our method against an untrained stage-1 model, which is equivalent to the GVL [16] formulation, and with Gemini-3.1-Thinking [8] as a backbone. A comparison of success rates can be found in Table 1, where RTA outperforms the other methods.

Additionally, to show that RTA provides a stable reward signal for agent learning, we report the Pearson correlation between mean cumulative reward and pass@5 after each training update for the best-performing run (in terms of success rate) for every method. Results are presented in Table 8. Appendix A.6 shows the evolution of the per-query *progress–time* Spearman reward during training on Catrap Level 2. We plot both the progress–time reward and the success rate for direct comparison; both metrics increase steadily and in pair, further validating the coherence and shaping power of our reward signal.

Continuous control. To evaluate RTA beyond discrete tasks, we test it on PointMaze-UMaze, and selected MetaWorld environments, which present more challenging long-horizon dynamics. For these tasks, we adopt DrQv2 [27] as the backbone and compare RTA with Rank2Reward (R2R). Both methods were trained for one million environment steps. Additionally, we ablate the effect of combining a GAIL-style reward with the ranking-based reward, following a similar approach to R2R, where the GAIL reward is added to the ranking-only reward to improve overall learning quality.

As shown in Figure 3, RTA consistently outperforms R2R on UMaze, both in the ranking-only and mixed reward settings, in terms of success rate. On MetaWorld tasks, RTA also surpasses R2R

Table 1: Comparison of success rates from different approaches. Means and std are computed over 5 seeds. Oracle reward is binary reward according to success of the end of level. RTA achieves the strongest performance across all levels except Kirby. Notably, it is the only method that attains a non-zero success rate on Kirby, despite not using any environment-provided rewards.

Task	Level 2	Level 4	Level 6	Kirby (level 0)
GVL	0.47 ± 0.25	0.00 ± 0.00	0.04 ± 0.08	0.00 ± 0.00
GVL-Gemini	0.27 ± 0.09	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
VLM-RM	0.40 ± 0.28	0.16 ± 0.20	0.08 ± 0.16	0.00 ± 0.00
VLM-RM$_{reg}(\alpha = 0.5)$	0.44 ± 0.32	0.00 ± 0.00	0.08 ± 0.16	0.00 ± 0.00
Rank2Reward	0.60 ± 0.28	0.20 ± 0.00	0.13 ± 0.09	-
Oracle reward	0.50 ± 0.22	0.07 ± 0.09	0.20 ± 0.00	**0.40 ± 0.28**
RTA (w stage1 training)	**1.00 ± 0.00**	**0.72 ± 0.35**	**0.32 ± 0.27**	0.07 ± 0.09

Table 2: Cross-domain evaluation of RTA using different video sources for Stage-1 training. Results in this table report mean ± std across 3 seeds. Scorers trained on different visual domains still provide meaningful rewards, enabling agents to complete levels as well as or better than the baselines reported in the Table 1.

Source	Level 2	Level 4	Level 6	Kirby (level 0)
Youtube	1.00 ± 0.00	0.47 ± 0.25	0.60 ± 0.28	0.20 ± 0.16
Full Catrap	1.00 ± 0.00	0.47 ± 0.38	0.60 ± 0.28	0.07 ± 0.09
Full Kirby	–	–	–	0.07 ± 0.09
MetaWorld	1.00 ± 0.00	0.87 ± 0.19	0.53 ± 0.19	0.13 ± 0.09
Coin (AssembleSofa)	1.00 ± 0.00	0.07 ± 0.09	0.20 ± 0.28	0.00 ± 0.00

under the ranking-only reward, though it still underperforms methods that incorporate an additional classifier-based (GAIL-style) reward. This behavior likely arises because MetaWorld environments require extensive exploration, and RTA occasionally assigns high VOC-scores to trajectories with plausible dynamics that are nevertheless unsuccessful.

However, R2R relies on pretraining a separate model for each task, whereas RTA allows a single pretrained model to be reused across multiple tasks, offering greater scalability. Success rates and comparisons to original dense rewards for MetaWorld tasks are provided in Figure 4.

We also analyze the efficiency of RTA compared to R2R under a matched compute budget (see Appendix A.7). The results show that RTA remains competitive across all tasks and outperforms R2R on the majority of them, indicating that its advantage is not due to increased computational resources.

Cross-domain generalization. To examine the generalization of the Stage 1 scorer across video sources, we carried out additional experiments in which RTA is trained on (a) a single full-game Catrap playthrough, (b) a mixture of YouTube PyBoy playthroughs from multiple games and players (excluding Catrap and Kirby), and (c) a Kirby playthrough. These sources differ in visual appearance (compression artifacts, overlays, distinct play styles, etc.) but share the same underlying game dynamics. Results are shown in Table 2.

To further show the generalization ability of RTA, we run additional experiments where the Stage 1 ranker is trained on data with very different visual dynamics. In particular, we train the ranker on expert trajectories from MetaWorld and on AssembleSofa videos from the COIN dataset. [23].

We also show that the generalization ability is preserved in MetaWorld environments when the ranker is trained on different source videos, including a YouTube GameBoy playthrough collection and AssembleSofa videos from the COIN dataset. Fig. 4).

Ablations. We conduct several ablations on window size, reward frequency, and reward averaging. In addition, we experiment with using an MLP backbone for the agent in Stage 2 instead of a VLM, demonstrating that the choice of policy model does not affect training outcomes and that performance gains are primarily due to reward design. Finally, we compare giving rewards after every N steps versus providing a single reward at the end of the trajectory (with samples taken uniformly over the trajectory). Further details are provided in Appendix A.5.

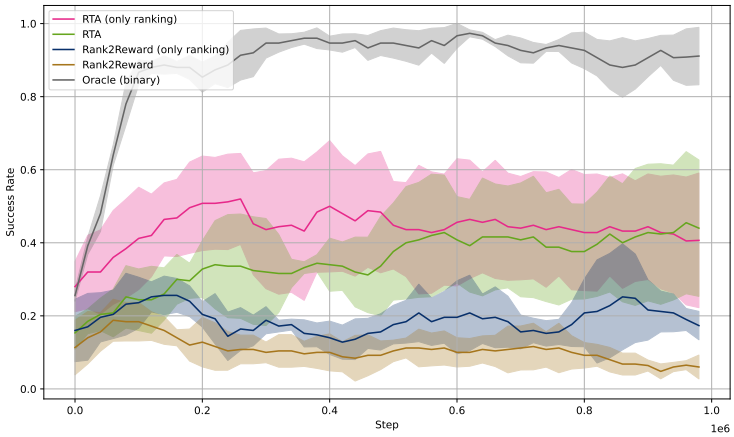

Figure 3: **PointMaze-UMaze results**. Results in this table report mean ± std across 5 seeds, smoothed with a moving average (window size 5) for visual clarity. RTA outperforms R2R both with and without GAIL. The oracle reward is included for comparison, although it is not directly comparable to our method.

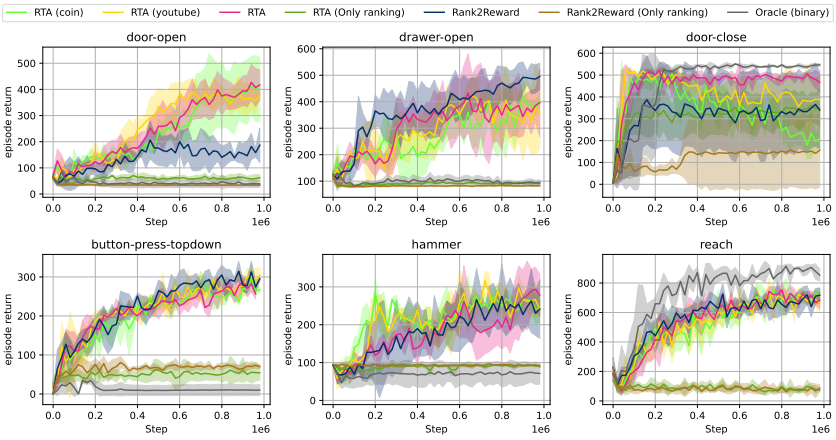

Figure 4: **MetaWorld different tasks results**. Stage-1 RTA is trained once on a mixed dataset of demonstrations, while R2R is trained separately per task. We additionally evaluate Stage-1 scorers trained on different data sources, including COIN and YouTube datasets. Results report mean ± std over 4 seeds. RTA + GAIL achieves comparable returns to R2R across all tasks, and outperforms it on `door-open` and `door-close`. Unlike R2R, RTA requires no per-task fine-tuning, and the Stage-1 scorer generalizes well across data sources and unseen environments, enabling successful task completion.

Robustness to temporal structure and window size. We study how the windowed rank-correlation reward behaves under different temporal scales using cyclic trajectories composed of forward and reversed segments (Appendix A.8). This construction explicitly removes global monotonic progress, allowing us to isolate the inductive bias of the reward signal.

In Fig. 5, we report cumulative normalized reward over time using non-overlapping windows of varying length. We find a clear scale-dependent behavior: (i) very small windows are overly sensitive to local temporal consistency and can produce monotonically increasing reward even in cyclic trajectories; (ii) intermediate windows suppress these local artifacts and better reflect the absence of global progress; and (iii) large windows lead to signal dilution due to temporal averaging.

Despite these differences, all window sizes consistently assign high reward to the final segment where the goal is achieved, indicating that the signal preserves sensitivity to true task completion. Overall, this reveals a stable intermediate regime in which windowed rank-correlation acts as a reliable progress estimator, while also exposing its limitations under extreme temporal scales.

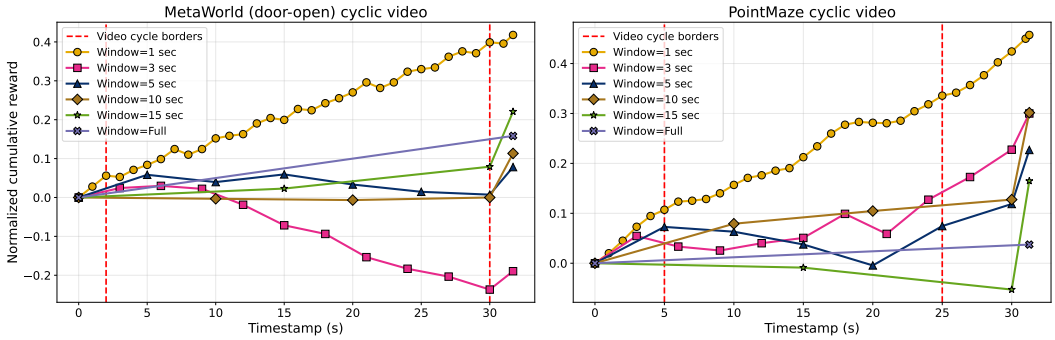

Figure 5: **Effect of window size on the reward signal on cyclic trajectory.** Small windows produce increasing reward despite no true progress, reflecting local consistency. Medium windows better capture the absence of global progress, while large windows yield near-zero signal due to averaging. All window sizes assign high reward to the final segment where the goal is achieved.

5 Discussion

By learning policies directly from expert videos using a progress–time correlation signal, our work suggests a scalable approach to reward-free learning. This reduces the need for manually designed reward functions and may help avoid reward hacking and unintended behaviors.

We leave detailed parsing issues to prior work. Stage 1 ranking is similar to reasoning-style VLM supervision, where parsing and ranking stability have been studied [22]. Similar issues in Stage 2 VLM-based learning have also been discussed in multi-turn settings [30].

We also show examples of correct and incorrect scoring, including cases where an undertrained Stage 1 model leads to wrong reward assignment (see Appendix A.6).

Our method supports scoring long trajectories by sampling frames and leveraging VLM reasoning to infer temporal order, which allows application beyond strictly monotonic tasks.

Limitations. The approach relies on sufficiently diverse expert video data; limited coverage or dataset bias may lead to level-specific overfitting and weaker cross-level transfer, as reflected in asymmetric Stage 1 generalization.

While RTA does not assume globally monotonic progress, the windowed rank-correlation signal exhibits scale-dependent behavior: small windows can overemphasize local temporal consistency, while large windows may dilute meaningful structure. Our cyclic analysis (Appendix A.8) and ablations (Appendix A.5) show that an intermediate window regime is most stable across environments. Future work could further improve robustness in highly non-monotonic settings via multi-scale or hierarchical temporal aggregation, or explicit subgoal-aware reward modeling.

Combining RTA with complementary signals (e.g., a GAIL-style reward) may further mitigate cases where the scorer assigns high rewards to out-of-domain or non-progressing trajectories.

Current VLM backbones also introduce latency despite caching and striding, motivating more efficient or distilled models.

Overall, ordinal video-based rewards remain a promising direction for scalable generalist control.

6 Conclusion

We present Rank-Then-Act (RTA), a simple two-stage method that teaches vision-language models (VLMs) to act using only expert video and progress ranks (via Spearman correlation), with no extrinsic rewards. In both challenging discrete-action games and long-horizon continuous environments, this purely reward-free signal trains agents that achieve high success rates and outperform strong baselines.

Because real-world rewards are often sparse or hard to design, learning from expert video alone offers a scalable path to multimodal agency while reducing risks like reward hacking.

We see this as a step toward robust, generalist agents that can understand and act in open, dynamic, and underspecified environments.

References

[1] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang 0028, Shijie Wang 0007, Jun Tang 0008, Humen Zhong, Yuanzhi Zhu, Ming-Hsuan Yang 0001, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang 0003, Haiyang Xu 0001, and Junyang Lin. Qwen2.5-vl technical report. *CoRR*, abs/2502.13923, 2025. doi: 10.48550/arxiv.2502.13923. URL https://doi.org/10.48550/arxiv.2502.13923.

[2] George Bredis, Stanislav Dereka, Viacheslav Sinii, Ruslan Rakhimov, and Daniil Gavrilov. Enhancing vision-language model training with reinforcement learning in synthetic worlds for real-world success. *CoRR*, abs/2508.04280, 2025. doi: 10.48550/arxiv.2508.04280. URL https://doi.org/10.48550/arxiv.2508.04280.

[3] Daniel S. Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. *ICML*, pages 783–792, 2019. URL http://proceedings.mlr.press/v97/brown19a.html.

[4] Daniel S. Brown, Wonjoon Goo, and Scott Niekum. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. *CoRL*, pages 330–359, 2019. URL http://proceedings.mlr.press/v100/brown20a.html.

[5] Jinglin Chen, Aditya Modi, Akshay Krishnamurthy, Nan Jiang, and Alekh Agarwal. On the statistical efficiency of reward-free exploration in non-linear rl. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 20960–20973. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/8433bb4f7477bf8202614ce1ae8b1169-Paper-Conference.pdf.

[6] Kevin Chen, Marco F. Cusumano-Towner, Brody Huval, Aleksei Petrenko, Jackson Hamburger, Vladlen Koltun, and Philipp Krähenbühl. Reinforcement learning for long-horizon interactive llm agents. *CoRR*, abs/2502.01600, 2025. doi: 10.48550/arxiv.2502.01600. URL https://doi.org/10.48550/arxiv.2502.01600.

[7] Rodrigo de Lazcano, Kallinteris Andreas, Jun Jet Tai, Seungjae Ryan Lee, and Jordan Terry. Gymnasium robotics, 2024. URL http://github.com/Farama-Foundation/Gymnasium-Robotics.

[8] Google. A new era of intelligence with gemini 3., 2025. URL https://deepmind.google/models/gemini/pro/.

[9] Sanjay Haresh, Sateesh Kumar, Huseyin Coskun, Shahram Najam Syed, Andrey Konin, M. Zeeshan Zia, and Quoc-Huy Tran. Learning by aligning videos in time. *CVPR*, pages 5548–5558, June 2021. doi: 10.1109/cvpr46437.2021.00550. URL https://doi.org/10.1109/cvpr46437.2021.00550.

[10] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 4572–4580, Red Hook, NY, USA, 2016. Curran Associates Inc. ISBN 9781510838819.

[11] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

[12] William Huey, Huaxiaoyue Wang, Anne Wu, Yoav Artzi, and Sanjiban Choudhury. Imitation learning from a single temporally misaligned video. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=YV05KZt7v2.

[13] Kuo-Han Hung, Pang-Chi Lo, Jia-Fong Yeh, Han-Yuan Hsu, Yi-Ting Chen, and Winston H. Hsu. Victor: Learning hierarchical vision-instruction correlation rewards for long-horizon manipulation. *CoRR*, abs/2405.16545, 2024. doi: 10.48550/arxiv.2405.16545. URL https://doi.org/10.48550/arxiv.2405.16545.

[14] Chi Jin, Akshay Krishnamurthy, Max Simchowitz, and Tiancheng Yu. Reward-free exploration for reinforcement learning. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4870–4879. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/jin20d.html.

[15] Emilie Kaufmann, Pierre Ménard, Omar Darwiche Domingues, Anders Jonsson, Edouard Leurent, and Michal Valko. Adaptive reward-free exploration. In Vitaly Feldman, Katrina Ligett, and Sivan Sabato, editors, *Proceedings of the 32nd International Conference on Algorithmic Learning Theory*, volume 132 of *Proceedings of Machine Learning Research*, pages 865–891. PMLR, 16–19 Mar 2021. URL https://proceedings.mlr.press/v132/kaufmann21a.html.

[16] Yecheng Jason Ma, Joey Hejna, Chuyuan Fu, Dhruv Shah, Jacky Liang, Zhuo Xu, Sean Kirmani, Peng Xu, Danny Driess, Ted Xiao, Osbert Bastani, Dinesh Jayaraman, Wenhao Yu, Tingnan Zhang, Dorsa Sadigh, and Fei Xia. Vision language models are in-context value learners. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=friHAl5ofG.

[17] Reginald McLean, Evangelos Chatzaroulas, Luc McCutcheon, Frank Röder, Tianhe Yu, Zhanpeng He, K.R. Zentner, Ryan Julian, J K Terry, Isaac Woungang, Nariman Farsad, and Pablo Samuel Castro. Meta-world+: An improved, standardized, RL benchmark. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2025. URL https://openreview.net/forum?id=1de3azE606.

[18] Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 2778–2787. JMLR.org, 2017.

[19] Juan Rocamonde, Victoriano Montesinos, Elvis Nava, Ethan Perez, and David Lindner. Vision-language models are zero-shot reward models for reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=N0I2RtD8je.

[20] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://dblp.uni-trier.de/db/journals/corr/corr1707.html#SchulmanWDRK17.

[21] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024. doi: 10.48550/arxiv.2402.03300. URL https://doi.org/10.48550/arxiv.2402.03300.

[22] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024. doi: 10.48550/arxiv.2402.03300. URL https://doi.org/10.48550/arxiv.2402.03300.

[23] Yansong Tang, Dajun Ding, Yongming Rao, Yu Zheng, Danyang Zhang, Lili Zhao, Jiwen Lu, and Jie Zhou. Coin: A large-scale dataset for comprehensive instructional video analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[24] Faraz Torabi, Garrett Warnell, and Peter Stone. Generative adversarial imitation from observation. In *Imitation, Intent, and Interaction (I3) Workshop at ICML 2019*, June 2019.

[25] Huaxiaoyue Wang, William Huey, Anne Wu, Yoav Artzi, and Sanjiban Choudhury. Time your rewards: Learning temporally consistent rewards from a single video demonstration. In *CoRL 2024 Workshop on Whole-body Control and Bimanual Manipulation: Applications in Humanoids and Beyond*, 2024. URL https://openreview.net/forum?id=gsgkiuv9BS.

[26] Daniel Yang, Davin Tjia, Jacob Berg, Dima Damen, Pulkit Agrawal, and Abhishek Gupta. Rank2reward: Learning shaped reward functions from passive video. *ICRA*, 2024.

[27] Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning, 2021. URL https://arxiv.org/abs/2107.09645.

[28] Kevin Zakka, Andy Zeng, Pete Florence, Jonathan Tompson, Jeannette Bohg, and Debidatta Dwibedi. Xirl: Cross-embodiment inverse reinforcement learning. *Conference on Robot Learning (CoRL)*, 2021.

[29] Shaopeng Zhai, Qi Zhang, Tianyi Zhang, Fuxian Huang, Haoran Zhang, Ming Zhou, Shengzhe Zhang, Litao Liu, Sixu Lin, and Jiangmiao Pang. A vision-language-action-critic model for robotic real-world reinforcement learning. *CoRR*, abs/2509.15937, 2025. doi: 10.48550/arxiv.2509.15937. URL https://doi.org/10.48550/arxiv.2509.15937.

[30] Yuexiang Zhai, Hao Bai, Zipeng Lin, Jiayi Pan 0002, Shengbang Tong, Yifei Zhou, Alane Suhr, Saining Xie, Yann LeCun, Yi Ma 0001, and Sergey Levine. Fine-tuning large vision-language models as decision-making agents via reinforcement learning. *CoRR*, abs/2405.10292, 2024. doi: 10.48550/arxiv.2405.10292. URL https://doi.org/10.48550/arxiv.2405.10292.

[31] Alex L. Zhang, Thomas L. Griffiths, Karthik R. Narasimhan, and Ofir Press. Videogamebench: Can vision-language models complete popular video games?, 2025. URL https://arxiv.org/abs/2505.18134.

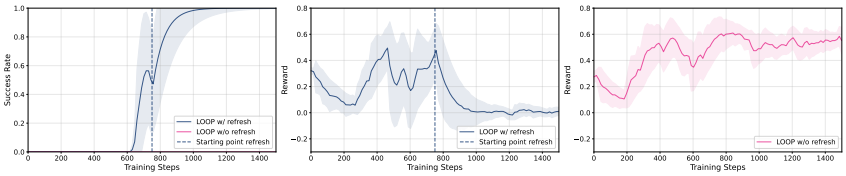

Figure 6: **LOOP benefits from starting-point refreshing.** Without refreshing, rewards improve but may not reach success when N is shorter than the task. Refreshing from the best terminal state after threshold τ enables completion; brief post-refresh dips arise if N exceeds remaining steps.

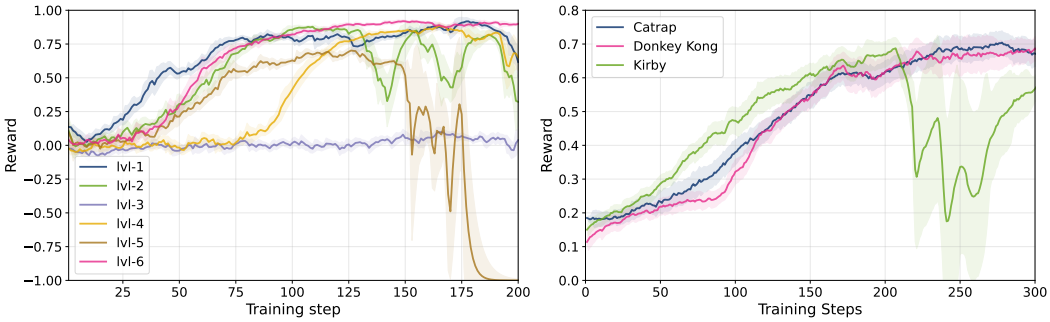

Figure 7: **Stage 1 learns reliable ordinal progress. Left:** per-level Catrap training shows fast rises in *progress–time* ρ with slower convergence on intricate levels. **Right:** training on full playthroughs across games also converges, with slightly lower plateaus due to non-informative segments (e.g., menus).

A Appendix

A.1 Experiments with LOOP

For experiments with LOOP-an extension of GRPO to multi-turn settings-we set the episode length to $\mathcal{T} = N$ and compute the reward only at the end of each episode, assigning the same advantage to all output tokens. Because this limits long-horizon environment interaction, we restart the trajectory from a new starting point whenever the average episode reward $r_{\mathcal{T}}$ across sampled traces exceeds a threshold τ. At each reset, we select the trace with the highest reward and repeat the process.

Figure 6 presents the per-query *progress–time* Spearman reward during training with LOOP [6]. We report two reward curves: one for training with **starting point refreshing** enabled, and another without refreshing.

When the model is trained without refreshing the starting point, rewards increase steadily, but there is no corresponding rise in success rate. This occurs because the default window length N is insufficient for the agent to complete the full task within a single episode. To address this, we introduce **starting point refreshing**: after achieving a threshold τ, we select the trajectory with the highest reward and begin the next rollout from the final state of that trajectory. This approach enables the algorithm to achieve high success rates, as reflected by the success curve.

However, if the new rollout's length N exceeds the number of steps required to finish the level, this can occasionally produce a temporary decline in reward after refreshing, as seen in the post-refresh segments of the plot. This effect is due to the agent finishing the level quickly, leaving extra rollout steps with little meaningful scoring signal.

A.2 Stage 1: Progress-Scorer Learning Dynamics

Per-level training curves. We first train the scorer on individual Catrap levels. The left panel of Fig. 7 shows the GRPO training curves for the first six levels. Most levels exhibit rapid convergence within 200 steps, achieving asymptotic *progress–time* Spearman ρ near 0.9. Some training runs diverge after reaching high performance-a phenomenon also observed in extended GRPO training,

and consistent with known GRPO instabilities. All training videos in this experiment were extracted under human supervision.

As shown in A.6, level 3 displays alternating periods of convergence and divergence, suggesting that even in levels with less clear task goals, progress can be learned, albeit with greater sensitivity to data preparation and training duration.

Full-playthrough training. Next, we pool all Catrap levels (1–8) and train a *single* scorer across this part of game. The right panel of Fig. 7 shows training ρ on a mixed-level and games held-out set. In these experiments, we train the model for 300 steps on three different games to achieve and sustain high asymptotic quality.

Pooling all levels increases data diversity and produces smoother training dynamics, though the final performance plateau is slightly lower than in most single-level runs. This drop is likely due to segments with static or non-informative frames (such as menus) present in full playthroughs. Importantly, this experiment demonstrates that a single progress scorer can be trained end-to-end from full-game videos, obviating the need for level-specific tuning and opening a path toward scalable, domain-general learning. Additional results confirm that per-game training also provides robust performance on level-specific validation.

A.3 Setup

Expert videos. For Stage 1, we use human *Catrap* playthroughs (levels 1–8; successful episodes), one *full-game* Catrap run, and 70 additional GameBoy playthroughs scraped from YouTube for discrete environments and 100 trajectories collected from expert agents for MetaWorld [17] and PointMaze [7] tasks. Videos are split into fixed-length clips of K consecutive frames and converted to the *anchor+shuffle* interface (Sec. 3.2). To standardize inputs and avoid cues such as completion time or lives, GameBoy frames are cropped from 640×360 to 640×330.

Tasks and levels. For control evaluation, we consider multiple Catrap levels (L2, L4, L6, with progressive difficulty) and a second game, Kirby (part until the first boss is met, corresponding to the first checkpoint in VideoGameBench [31], we'll call it level-0 for simplicity), in the PyBoy emulator. Catrap is a logically complex puzzle requiring spatial planning. Kirby is a standard VLM benchmark from VideoGameBench. Both use the simplest available levels that require at least 15 steps for successful completion. For MetaWolrd we choose tasks `door-ope`, `drawer-open`, `door-close`, `button-press-topdown`, `hammer`, `reach`, and for PointMaze we test our setup on UMaze task. Each task is defined by a goal description and termination condition, and we train separate policies per task while reusing a trained (from per-task to general) Stage-1 progress scorer. This setup stresses both cross-level and cross-game reuse of the scorer, and enables us to probe robustness of the progress–time correlation reward beyond a single level.

Observations and actions. For Stage 2 policy learning in PyBoy games, we render environment observations at 160×144 pixels; for policy inputs, frames are centrally cropped and resized to 160×120. The action space consists of the four D-pad directions (`up`, `down`, `left`, `right`) defined by PyBoy's discrete control interface. To ensure consistency with Stage 1, we resize frames back to the original training resolution before querying the frozen scorer f_ϕ to obtain progress ranks, which we then correlate (*progress–time* Spearman) with timestamps. For Kirby we allow pressing multiple buttons simultaneously, following the VideoGameBench Light setup. For continuous environments we follow the setup from Rank2Reward [26].

Technical details. For Stage1 and Stage 2 experiments with VL-DAC as a backbone policy, we train only the LoRA adapters (rank 16, alpha 32) on top of the provided models. For Stage 1 scorer training, each reported experiment requires 4 H100 GPUs, and 200 training steps take approximately 2 hours. For Stage 2 policy learning (with RTA as a reward model), 1 H100 GPU is required, and training takes up to 24 hours. Detailed hyperparameter settings are provided in Appendix A.4.

A.4 Experimental details

Hyperparameters (summary). Stage 1: GRPO on a tuned VLM; the first frame is anchored, and the remaining $K-1$ frames are shuffled; the listwise reward is $R = \mathrm{spr}(\mathbf{p}, \mathbf{q})$, where $\mathbf{p}$ denotes *progress ranks* and $\mathbf{q}$ denotes temporal indices (timestamps); a moving-average baseline is used. **Stage 2:** Reward frequency $N = 15$; $L = 2$ shuffled-window evaluations per query; window length m as in

Table 3: **Stage 2 (VL-DAC) hyperparameters.** Policy learns from the windowed *progress–time* $\rho \in [-1, 1]$ computed every N steps (default $N = 15$, $L = 2$). γ_g, λ_g are GAE parameters; LR follows a cosine decay.

Hyperparameter	Values
Env.steps	57600
Learning Rate (init $\rightarrow$ final)	1e-5 $\rightarrow$ 5e-7
Scheduler	cosine
GAE λ_g	0.95
γ_g	0.99
Value Loss Coeff.	0.15
KL β	0.05
Policy Freeze (steps)	2
Grad Accum. Steps	32
Mini-batch Size	1
PPO Epochs	2
Obs. Image Length	5
Rollout Size	256
Max Episode Steps	64
Temperature	0.2

Table 4: **Stage 2 (LOOP) hyperparameters.** Multi-step GRPO variant using the same correlation-only reward. Training optionally applies *starting-point refreshing* when reward exceeds threshold τ to enable long-horizon completion.

Hyperparameter	Values
Algorithm steps	225
Learning Rate (init $\rightarrow$ final)	1e-5 $\rightarrow$ 1e-6
Scheduler	linear with warmup
Num. warmup steps	10
KL β	0.05
Grad Accum. Steps	32
Mini-batch Size	1
PPO Epochs	1
Obs. Image Length	5
Rollout Size	120
τ threshold	0.5
τ threshold steps	3
K	4
Temperature	1

Sec. 3.3 (we ablate N and L); per-episode standardization is used only for advantage estimation. The policy and value networks in Stage 2 are trained with VL-DAC [2] and a multi-step variant of GRPO, LOOP [6]. Importantly, no environment rewards are used at any stage. Additionaly we employ kl control on stage1, we set target kl to 0.1. Also we list hyperparameters for stage1 and stage2 in tables 4 3 5

A.5 Ablations

In this section, we present an expanded set of ablations evaluating both algorithmic hyperparameters of RTA and architectural choices for the downstream policy. These studies are designed to isolate the contribution of each component and determine the robustness of RTA across implementation variations. All experiments are conducted on Catrap environments and results are averaged over multiple seeds.

Policy backbone choice. We further evaluate how RTA interacts with different policy backbones, comparing a VLM with a lightweight MLP trained with PPO. Table 6 summarizes the results for two

Table 5: **Stage 1 scorer hyperparameters.** GRPO on anchor+shuffle clips; the VLM predicts per-frame *progress ranks* and maximizes *progress–time* Spearman ρ. After convergence, the scorer is frozen for Stage 2.

Hyperparameter	Values
Algo steps	200-400
Learning Rate	1e-5
Scheduler	constant with warmup
Num. warmup steps	10
Grad Accum. Steps	16
Mini-batch Size	1
GRPO Epochs	1
Obs. Image Length	15
K	4
Temperature	1

downstream settings: providing RTA reward every $N = 15$ steps and providing reward only at the end of an episode.

We observe that the optimal schedule differs between the VLM and MLP policies, plausibly due to different exploration dynamics: an initialized MLP benefits from termination step RTA feedback (we reward full trajectory), whereas the VLM policy often benefits from shorter-window shaping. Overall, these results are consistent with the view that RTA's gains are driven by the reward signal's ordering stability rather than a particular policy architecture.

Table 6: Downstream PPO evaluation results. Mean ± std over 3 seeds for VLM-based agents and 5 seeds for MLP agents. RTA maintains a high success rate even with an MLP backbone, indicating that its performance stems from the reward design rather than biases in VLMs.

Task	Level 2	Level 4	Level 6
VLM + RTA (reward every 15 steps)	1.00 ± 0.00	0.60 ± 0.28	0.33 ± 0.34
VLM + RTA (only-end reward)	0.33 ± 0.19	0.00 ± 0.00	0.00 ± 0.00
MLP + RTA (reward every 15 steps)	0.79 ± 0.14	0.27 ± 0.21	-
MLP + RTA (only-end reward)	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00

Window mechanics. We vary the window length m around our default. Across Catrap levels, we find that performance is stable with the default value $m = 15$ and higher, while very short windows (too few frames) perform worse, as expected. For larger windows than default, convergence tends to be the same, and inference is slower due to the larger context processed by the reward model. We fix the reward frequency to 15 and L to 2.

Reward frequency. We vary the frequency at which reward is provided to the learning process. We find that this parameter does not qualitatively affect stability, and its effect on convergence speed appears to depend more on optimization dynamics. We fix the window length to 15 and L to 2.

Number of shuffles L. Reducing the number of shuffles to 1 degrades reward stability and can lead to slower convergence, indicating that multiple shuffles are important for learning a robust ordinal signal. We fix the window length to 15 and reward frequency to 15.

More detailed results of hyperparameters sweep can be found in 7.

A.6 Examples

Level 2 training curves. Figure 12 reports the *success rate* (rolling window over episodes) alongside the per-query *progress–time* reward when optimizing with VL-DAC [2]. In the plot, we show results from two random seeds, with quality smoothed with an EMA over the last 10 *queries*. At several points during training, the RTA-trained agent attains 100% success (if we turn off smoothing).

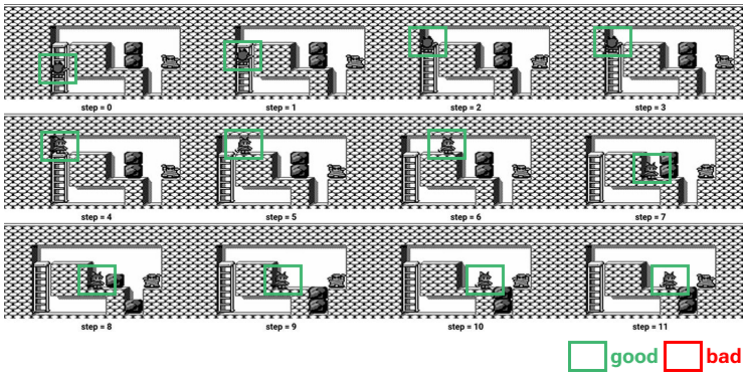

Figure 8: Example of good level completion with well-trained stage 1 model with reward 0.77. The character (highlighted by a box) moves toward completing the level and receives a high reward from the scorer.

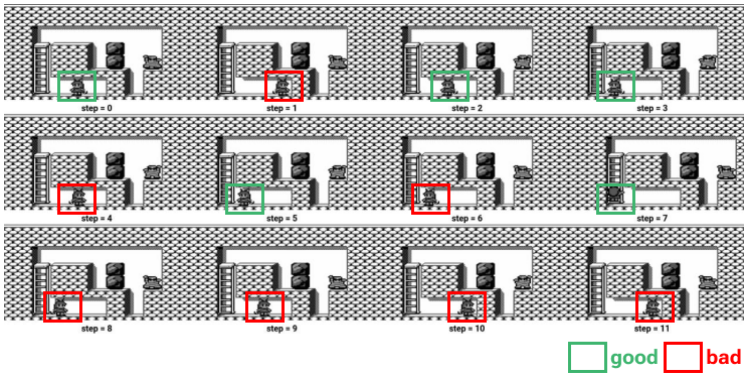

Figure 9: Example of bad level compeletion with well-trained stage 1 model with reward -0.04. The character (highlighted by a box) moves back and forth without making progress, and is therefore assigned a low score.

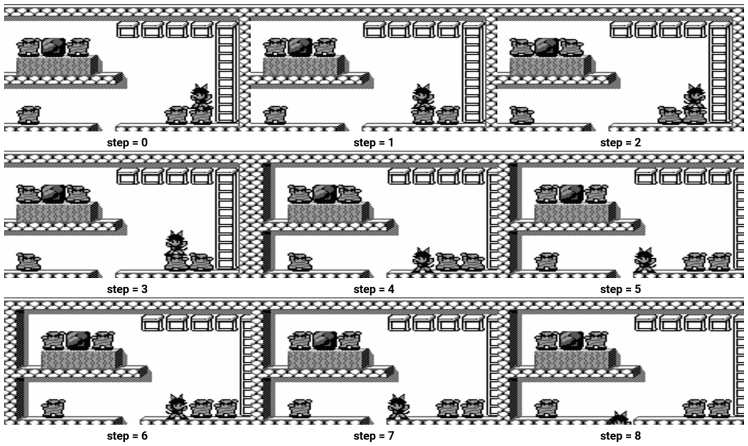

Figure 10: Example of bad level compeletion with undertrained stage 1 model with reward 0.66 (high). We apply a scorer trained on a different level to Level 3. It still assigns a high score to a trajectory in which the character moves left but ends in a dead end.

Table 7: **Ablation study on window length M, reward frequency, and number of shuffles L on Catrap L2.** RTA achieves strong performance across a range of settings, with best results at intermediate values. Performance degrades slightly and variance increases at extreme configurations, indicating sensitivity to the choice of hyperparameters.

Ablation: Window Length M	Success rate	Update steps
5	0.73 ± 0.09	65.33 ± 12.76
15	**1.00 ± 0.00**	**37.00 ± 10.98**
25	**1.00 ± 0.00**	43.33 ± 5.79
Ablation: Reward Frequency		
5	**1.00 ± 0.00**	175.60 ± 76.53
15	**1.00 ± 0.00**	**37.00 ± 10.98**
25	**1.00 ± 0.00**	128.67 ± 84.32
Ablation: Number of Shuffles L		
1	**1.00 ± 0.00**	143.66 ± 63.43
2	**1.00 ± 0.00**	**37.00 ± 10.98**
4	0.93 ± 0.09	123.33 ± 83.83

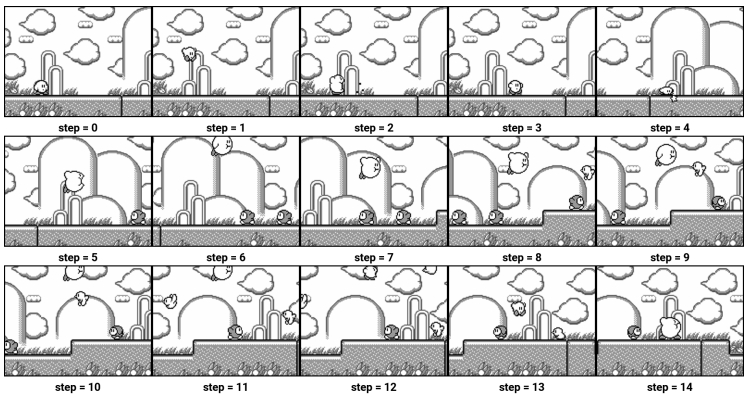

Figure 11: Example of good level compeletion with undertrained stage 1 model with reward 0.15 (low). We apply a scorer trained on a different level to Kirby lvl 0. As this environment is out-of-distribution and includes a moving background, the scorer fails to assign a high score even when the character makes clear progress.

Although the reward is computed sparsely (every $N = 15$ steps) and is always bounded in $[-1, 1]$, it provides sufficiently shaped feedback for policy learning: the mean reward increases steadily throughout training, closely tracking the agent's success rate.

This experiment demonstrates that the progress–time correlation signal-derived solely from video-based ordinal progress-is both informative and stable enough to drive effective online RL, even in the complete absence of extrinsic rewards.

Overall Pearson correlation for GameBoy games is reported in Table 8.

Level 3 performance Notably, level 3 did not converge within 200 steps, prompting further experiments with longer training on this level (see Fig. 13).

A.7 Training efficiency

We have compared RTA and Rank2Reward under the same compute budget. Fig. 14 reports episode return versus total wall-clock time, rather than training iterations, so the comparison is budget-aligned rather than step-aligned. Under this matched budget, RTA is stronger on `door-open`, `door-close`, and `reach`; it reaches near-parity on `drawer-open`; and Rank2Reward is somewhat stronger on `button-press-topdown` and `hammer`. The main point is therefore not that RTA wins every task,

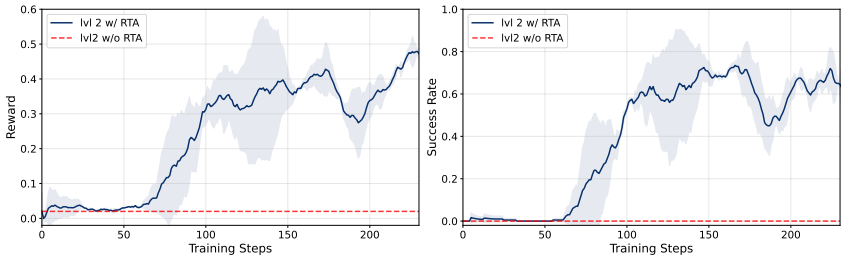

Figure 12: **Correlation-only reward drives control.** For Catrap L2 mean accumulated VOC-reward during training highly correlates with the success rate over episode.

Table 8: Pearson correlation of mean cumulative reward and pass@5 during training. We report correlation of 0 if method is unable to solve level and N/A if experiment was not conducted. Results in this table reported across 5 seeds. RTA-assigned rewards exhibit a strong correlation with successful level completion.

Task	Level 2	Level 4	Level 6	Kirby (level 0)
GVL	-0.01	0.00	-0.01	0.00
GVL-Gemini	0.33	0.00	0.00	0.00
Rank2Reward	-0.04	-0.02	-0.04	N/A
VLM-RM	0.53	-0.19	-0.33	0.00
VLM-RM$_{\text{reg}}$ ($\alpha = 0.5$)	0.25	0.00	-0.16	0.00
RTA (w stage1 training)	**0.76**	**0.87**	**0.42**	**0.13**

but that its advantage is not explained by extra compute. Under matched budget, it is competitive across all tasks and better on a majority of them. The same holds for PointMaze environment, RTA outperforms Rank2Reward given equal compute budget Fig. 15.

A.8 Cyclic trajectory construction.

We evaluate reward behavior using cyclic trajectories constructed by combining forward and reversed segments of expert demonstrations Fig.5. We mark the cyclic region explicitly and plot cumulative normalized Spearman correlation reward over time. We test this setup on expert trajectories from MetaWorld `door-open` task and one UMaze expert video from PointMaze. We use Stage 1 scorers that were trained per those tasks respectively.

The trajectory is divided into non-overlapping windows of varying sizes (1, 3, 5, 10, 15 seconds and full video). From each window, we uniformly sample a fixed number of frames and compute rank correlation with respect to temporal order.

We observe three regimes. For small windows (e.g., 1 second), the reward increases steadily even within cyclic regions, indicating sensitivity to local ordering but not global progress. For intermediate windows (e.g., 3–5 seconds), the cumulative reward decreases or stabilizes, better reflecting the absence of true progress. For large windows ($\geq$ 5 seconds), the signal approaches zero due to averaging across inconsistent temporal segments. Since the PointMaze trajectory is longer than the MetaWorld one, the segments outside the cyclic boundary are more pronounced. As a result, the 5-second window yields the most stable behavior when evaluated relative to the cycle boundary onset.

After the cyclic region ends and the trajectory resumes normal forward progress, all window sizes produce a sharp increase in reward, correctly identifying goal completion. These results highlight the importance of window size as a trade-off between local sensitivity and global consistency.

Window size selection. Based on the analysis above, we fix a single intermediate window size across all experiments and do not tune it per task. Additional ablations on window size (Catrap Level 2) are provided in Table 7. Empirically, this choice falls within a stable regime where the reward signal avoids both local artifacts from very small windows and signal dilution from very large ones, while remaining consistent across environments.

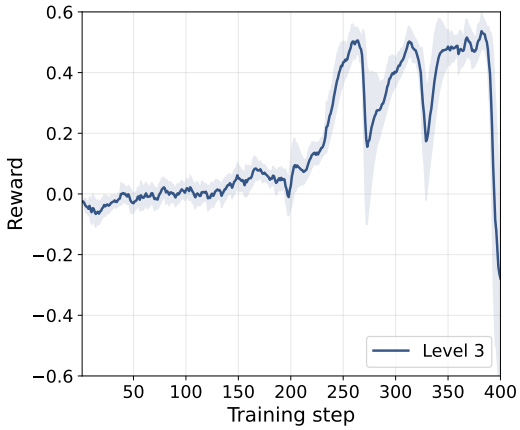

Figure 13: **Level 3 exposes training sensitivity.** Over 400 GRPO steps, *progress–time* ρ alternates between gains and dips, indicating that ambiguous content learns but benefits from careful data prep and sufficient duration.

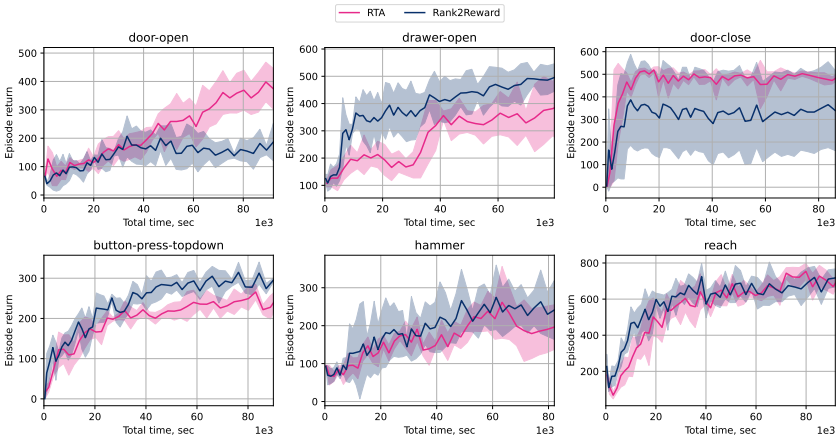

Figure 14: Training efficiency of RTA compared to Rank2Reward on MetaWorld tasks under aligned computational budget.

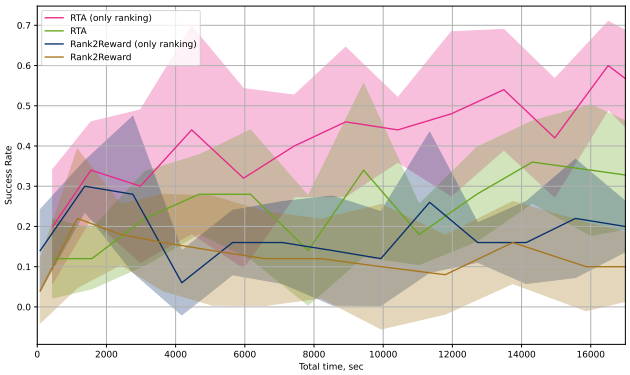

Figure 15: Training efficiency of RTA compared to Rank2Reward on UMaze under aligned computational budget.