Odysseus: Scaling VLMs to 100+ Turn Decision-Making in Games via Reinforcement Learning

Chengshuai Shi[1,*], **Wenzhe Li**[1,*], **Xinran Liang**[1,*], **Yizhou Lu**[2], **Wenjia Yang**[3], **Ruirong Feng**[1], **Seth Karten**[1], **Ziran Yang**[1], **Zihan Ding**[1], **Gabriel Sarch**[1], **Danqi Chen**[1], **Karthik Narasimhan**[1], **Chi Jin**[1]
[1] Princeton Language and Intelligence, Princeton University
[2] Fudan University [3] Tsinghua University
[*] Equal contribution in random order

Project Page: odysseus-project.github.io/

Abstract

Given the rapidly growing capabilities of vision-language models (VLMs), extending them to interactive decision-making tasks such as video games has emerged as a promising frontier. However, existing approaches either rely on large-scale supervised fine-tuning (SFT) on human trajectories or apply reinforcement learning (RL) only in relatively short-horizon settings (typically around 20–30 environment turns). In this work, we study RL-based training of VLMs for long-horizon decision-making in *Super Mario Land*, a visually grounded environment requiring 100+ turns of interaction with coordinated perception, reasoning, and action. We begin with a systematic investigation of key algorithmic components and propose an adapted variant of PPO with a lightweight turn-level critic, which substantially improves training stability and sample efficiency over critic-free methods such as GRPO and Reinforce++. We further show that pretrained VLMs provide strong action priors, significantly improving sample efficiency during RL training and reducing the need for manual design choices such as action engineering, compared to classical deep RL trained from scratch. Building on these insights, we introduce Odysseus, an open training framework for VLM agents, achieving substantial gains across multiple levels of the game and at least 3× average game progresses than frontier models. Moreover, the trained models exhibit consistent improvements under both in-game and cross-game generalization settings, while maintaining general-domain capabilities. Overall, our results identify key ingredients for making RL stable and effective in long-horizon, multi-modal settings, and provide practical guidance for developing VLMs as embodied agents.

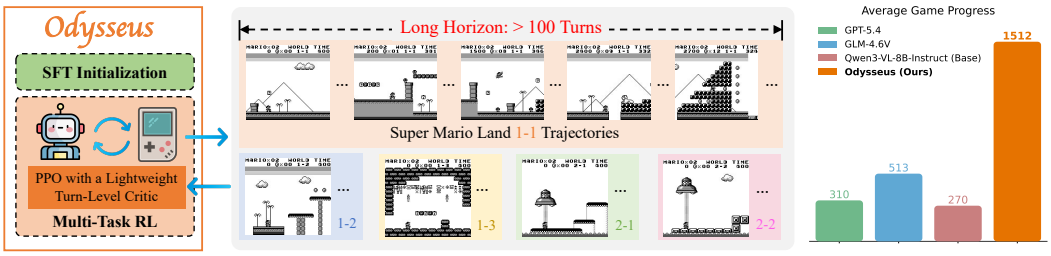


Figure 1: An overview of Odysseus for scaling VLMs to 100+ turn decision-making in the video game *Super Mario Land*, along with a comparison of performance averaged over the first five game levels across frontier models (GPT-5.4 and GLM-4.6V), the base model (Qwen3-VL-Instruct-8B) prior to training, and the Odysseus model after training. We observe that Odysseus achieves approximately 5× higher game progress than GPT-5.4, 3× higher than GLM-4.6V, and 6× higher than the base model.

1 Introduction

Multi-modal foundation models, particularly vision-language models (VLMs), have demonstrated remarkable capabilities across a wide range of domains, such as image captioning, object detection, and visual reasoning. Building on these advances, there is growing interest in extending them toward *agentic tasks*, where models are endowed with the ability to interact with external environments (Yao et al., 2022). Representative applications span web agent (Deng et al., 2023), GUI agents (Nguyen et al., 2025), and software engineering agent (Chan et al., 2024; Yang et al., 2024), with increasing attention on **embodied agents** performing interactive decision-making tasks in physically grounded or simulated environments (Driess et al., 2023; Zitkovich et al., 2023; Mu et al., 2023; Raad et al., 2024; Szot et al., 2025).

As a long-standing testbed of simulated embodied task (Mnih et al., 2015; Vinyals et al., 2019), video games have attracted growing interest for evaluating, scaffolding, and training VLM agents (Zhang et al., 2025; Hu et al., 2025; Wang et al., 2023; Karten et al., 2025a; Bolton et al., 2025). From the perspective of training, while reinforcement learning (RL) has achieved great success in training classical deep neural networks in video games (Badia et al., 2020), and has recently been applied to improve foundation models in domains such as preference alignment (Ouyang et al., 2022) and reasoning (Guo et al., 2025), existing approaches to fine-tuning VLMs for embodied tasks—particularly in video games—remain limited. Current methods either rely on large-scale supervised fine-tuning (SFT) with human trajectories (i.e., imitation learning) (Tan et al., 2025; Magne et al., 2026), which is difficult to scale, or apply RL only to relatively short-horizon tasks (typically around 20–30 turns) (Zhai et al., 2024; Wang et al., 2025a). It remains unclear whether RL can be effectively applied for training VLMs in more challenging, long-horizon ($>$ 100 turns) decision-making tasks.

In this work, we use the video game *Super Mario Land* to study this regime. Despite its simplicity, this environment remains challenging even for frontier models (Zhang et al., 2025). Successful performance requires coordinated perception, reasoning, and action over extended trajectories, often exceeding **100 turns**, and the ability to generalize across diverse levels with varying layouts and dynamics. Our contributions are summarized as follows.

• **Algorithmic ingredients.** We investigate key algorithmic choices required to effectively fine-tune VLMs via RL in long-horizon game environments. While popular critic-free methods perform poorly in this setting, we demonstrate that an adapted version of PPO achieves strong stability and sample efficiency. Crucially, we introduce a *lightweight turn-level critic* (instead of a large model based one as in Wang et al. (2025a); Zhai et al. (2024)) and *positive-advantage filtering*, which together decouple temporal credit assignment from token generation, mitigate optimization instability, and bypass the massive computational overhead usually associated with large-model-based actor-critic training.

• **VLM-based RL training v.s. classical deep RL.** Beyond enabling stable RL training of VLMs in this setting, we identify the advantages of VLM-based RL compared to classical deep RL which trains policies from scratch. We show that pretrained VLMs offer strong action priors that improve sample efficiency and reduce the need for manual designs such as action space engineering. This highlights the importance of general-purpose knowledge encoded in foundation models for scaling toward capable embodied agents.

• **An open training framework for practical agentic tasks.** Built upon these insights, we introduce the training framework **Odysseus** (inspired by the protagonist of Homer's Odyssey) that integrates lightweight supervised initialization with multi-task RL (Fig. 1). We show that Odysseus enables stable training while achieving substantial performance gains across the game over the base model, and outperforming both open-source and proprietary frontier models by a large margin (at least $3\times$ improvement in game progress). Furthermore, the resulting agents exhibit generalization both within the game and to related game environments, while retaining their general multi-modal capabilities.

Taken together, our results demonstrate that RL can be made stable and effective for training VLMs in **100+ turn decision-making environments**. Moreover, once RL is properly stabilized, foundation models provide strong priors that further facilitate learning. We hope this work provides a practical foundation and opens the door to future advances in RL training of multi-modal foundation models as embodied agents.

2 Related Works

Games and Simulated Environments. While video games and simulated environments have long served as testbeds for machine learning, the recent exploration of RL for VLMs is mostly focused on short-horizon scenarios, such as AlfWorld (Shridhar et al., 2021), Sokoban, and FrozenLake (Wang et al., 2025a). In contrast, we focus on the video game *Super Mario Land* as a compact but appealing testbed for VLMs in long-horizon embodied control. It imposes substantially richer spatial grounding and closed-loop control than short-horizon gridworld-style tasks, while remaining lightweight and easy to scale for controlled studies compared with large open-world simulators (Fan et al., 2022; Tan et al., 2025).

Foundation Models for Decision-Making. Recent advancements have shifted toward fine-tuning pretrained foundation models directly for embodied control, yielding capable agents in robotic manipulation (Black et al., 2024; Liu et al., 2024) and cross-game generalization (Tan et al., 2025; Bolton et al., 2025). However, these approaches heavily depend on Supervised Fine-Tuning (SFT) with large amounts of action-labeled demonstration data. Our work differentiates itself by focusing on the RL perspective.

RL for Foundation-Model Agents. A growing body of work studies RL for multi-turn language and vision-language agents (Zhai et al., 2024; Wang et al., 2025a;b; Li et al., 2026a; He et al., 2026). These methods often introduce specialized machinery for trajectory decomposition, token-level advantage estimation, or hierarchical credit assignment, and are typically evaluated on environments with relatively short horizons (20–30 turns). In contrast, we focus specifically on long-horizon, visually grounded embodied environments that require 100+ turns of interaction with chain-of-thought (CoT) reasoning. Through rigorous ablations, we show that a comparatively simple PPO-based approach with the right critic design is sufficient to make RL stable and effective.

3 VLMs for Decision-Making in *Super Mario Land*

3.1 Formulation

A commonly adopted abstraction for decision-making tasks follow the formulation of Partially Observable Markov Decision Process (POMDP) from classical RL literature (Sutton et al., 1998). Specifically, a POMDP is defined by the tuple $\langle \mathcal{S}, \mathcal{A}, \Omega, \mathcal{P}, \mathcal{O}, \mathcal{R} \rangle$, where $\mathcal{S}$ denotes the underlying state space, $\mathcal{A}$ is the action space, and Ω represents the observation space. At each environment turn t, the environment is in an unobserved state $s_t \in \mathcal{S}$, while the agent receives an observation $o_t \in \Omega$. The agent takes an action $a_t \in \mathcal{A}$, after which the environment transitions to a new state $s_{t+1} \sim \mathcal{P}(\cdot \mid s_t, a_t)$, and produces a new observation $o_{t+1} \sim \mathcal{O}(\cdot \mid s_{t+1})$ along with a scalar reward $r_t = \mathcal{R}(s_t, a_t)$.

The agent aims to achieve high performance in the environment, which in RL is formulated as maximizing the expected cumulative discounted reward under a parameterized policy π_θ (i.e., the adopted VLM), i.e., $\mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T-1} \gamma^t r_t\right]$, with T denoting the horizon and $\gamma \in [0, 1)$ denoting the discount factor. In the partially observable setting, the policy is defined as a mapping from the interaction history $h_t = (o_1, a_1, \ldots, o_t)$ to a distribution over actions, i.e., $a_t \sim \pi_\theta(\cdot \mid h_t)$, where the exact dependence on h_t depends on the design of the agent.

3.2 An Overview of *Super Mario Land*

To study the problem of effectively training VLMs for long-horizon decision-making tasks via RL, we consider the video game *Super Mario Land* as a compelling testbed. The game requires agents to perform accurate spatial perception and reasoning over extended trajectories (often 100+ turns), together with precise motor control to navigate diverse levels populated with obstacles and adversaries. Table 1 highlights the significantly longer horizon of *Super Mario Land* compared with previous game environments used in VLM-RL literature (Shridhar

Game	Effective Horizon
AlfWorld	$\sim 10 - 20$
Sokoban (6×6)	$\sim 5 - 30$
Frozen Lake (4×4)	$\sim 5 - 30$
Super Mario Land	> 100

Table 1: Effective horizon comparisons.

et al., 2021; Wang et al., 2025a). This game consists of 12 levels in total (in particular, 4 worlds, each with 3 levels), and we mainly consider 10 of these levels in this work, excluding 2 levels (World 2 Level 3 and World 4 Level 3) due to their distinct control mechanisms.

Recent works (Park et al., 2025; Zhang et al., 2025; Hu et al., 2025) have also used this game or other versions from the Super Mario series to benchmark foundation models. As demonstrated in these works—and as we will show later—even state-of-the-art models struggle in the zero-shot setting. For example, tasks such as jumping over a gap or avoiding a moving threat at the correct timing remain challenging, resulting in brittle policies that rarely progress beyond the initial stages of the game. In contrast, a human player with no prior experience can readily achieve non-trivial progress.

With the interaction protocols specified in Section 3.3, we briefly note that for the game, the **state space** $\mathcal{S}$ corresponds to the full internal state of game RAM. As in human gameplay, the agent does not have access to this state, and instead observes the **observation space** Ω, consisting of rendered pixel frames and textual prompts. The **action space** $\mathcal{A}$ is a discrete set of combinations derived from standard controller inputs. For language-based foundation models, actions are produced by generating text tokens that specify the buttons to be pressed. Finally, the **reward function** $\mathcal{R}$ reflects task progress, and is primarily defined in this work as forward movement at each turn, i.e., $r_t = x_{t+1} - x_t$, a minimal yet dense learning signal for task progress, where x_t is the x-coordinate of Mario at turn t read from game RAM.

3.3 Interaction Protocol

To facilitate the interaction between an VLM-based agent and the dynamic environment of *Super Mario Land*, we establish a structured, turn-based protocol (Fig. 2):

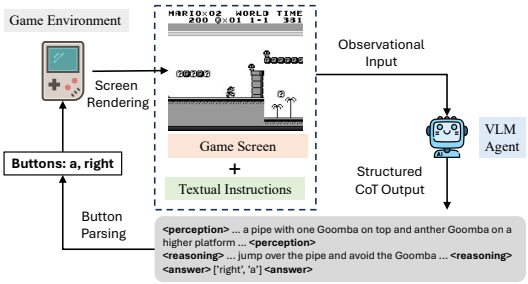


Figure 2: The interaction protocol between the VLM agent and the game environment.

Observational Inputs. The agent receives a comprehensive prompt specifying the overall rules of the environment. This includes the basic mechanics and objectives of the game, a precise definition of the available discrete action space, and output format instructions. The full text of this prompt is provided in Section C. In addition to these general instructions, the agent is given the current game frame rendered on the screen. While recent agent designs often provide richer inputs to the VLM (e.g., a longer observation history, or information parsed from underlying game states) (Park et al., 2025), we intentionally adopt a minimal-scaffolding design by providing only the current game frame and the prompt, which isolates what can be learned by RL fine-tuning the VLM itself.

Structured Chain-of-Thought (CoT). To elicit robust spatial-temporal grounding, we instruct the VLM to structure its decision process using three XML-style tags:

- **`<perception>`:** The agent first explicitly describes the visual state of the screen. This grounding step encourages the model to identify Mario's location, nearby obstacles, enemies, and interactive elements such as coins or pipes.
- **`<reasoning>`:** The agent then lays out its strategy step by step. It explains the actions required to respond to the current state, such as timing a jump to collect floating coins or moving right to approach a pipe stack.
- **`<answer>`:** Finally, the agent outputs the selected action combination as a list of button strings (e.g., `['a', 'right']`). The action space permits pressing up to two buttons simultaneously out from a total list of seven (i.e., `a`, `b`, `up`, `down`, `left`, `right`, `noop`), enabling more complex behaviors such as running jumps.

This response format largely follows the style of ReAct (Yao et al., 2022) and is also similarly adopted in Chen et al. (2025b); Wang et al. (2025a).

Action Execution. Once the structured CoT is generated by the VLM, the final action is parsed and executed in the game environment. Because a single emulator frame (1/60

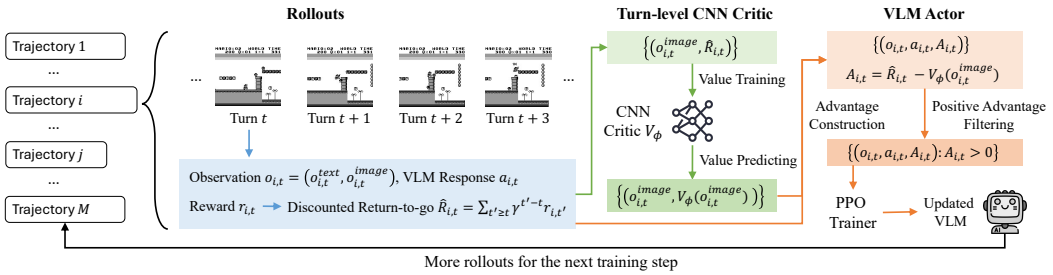


Figure 3: The adapted PPO algorithm used in Odysseus with a lightweight turn-level CNN critic and positive advantage filtering.

s) produces negligible movement, we implement a frame-skip mechanism: the chosen discrete action is repeatedly applied for a fixed number of consecutive frames to ensure an observable effect. Details are provided in Section C.

4 Algorithmic Ingredients of Stable RL for VLMs

In this section, we focus on examining whether RL can be effectively applied to train VLMs in the considered long-horizon game and if so, what algorithmic designs are important.

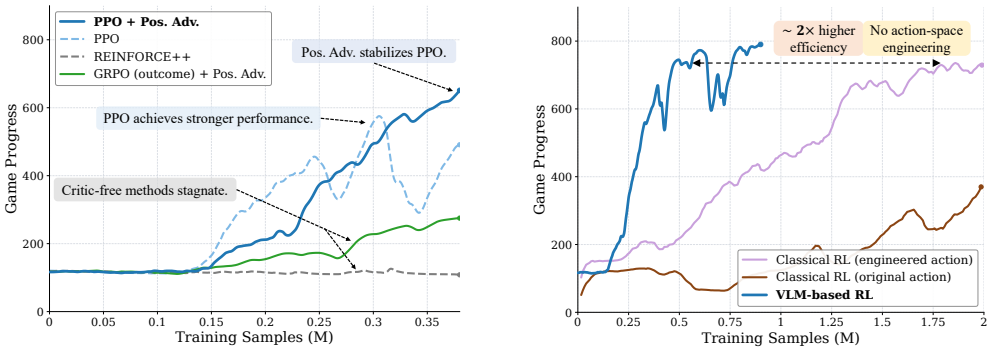
4.1 PPO with a Lightweight Turn-Level Critic

We first consider extending commonly-used RL fine-tuning algorithms to the considered long-horizon dense-reward settings, including GRPO (Shao et al., 2024) and Reinforce++ (Hu, 2025), which leverage critic-free strategies for advantage estimation. However, both outcome-reward and process-reward variants of these methods fail to learn effective policies that can make consistent multi-step progress (see results in Section 4.2). This particular failure mode motivates us to revisit the classical Proximal Policy Optimization (PPO) algorithm (Schulman et al., 2017), which employs a learned critic that enables better long-term credit assignment and low-variance advantage estimation (Schulman et al., 2015). We defer full algorithm details to Section D while highlighting several key design choices as follows, which are also illustrated in Fig. 3.

One major challenge in using PPO to train foundation models is the computation overhead of learning the critic — prior work usually learns a large-model-based token-level critic, which nearly doubles the memory and computation costs compared to critic-free methods. To address this issue, we propose two key changes to the original PPO algorithm used in RLHF (Ouyang et al., 2022). First, a *turn-level critic* is adopted. Compared to token-level critic learning, turn-level critic can directly leverage the dense reward signals from game environments and address the long-horizon temporal credit assignment more effectively.

Furthermore, we demonstrate that the critic network can be designed as a remarkably lightweight module, particularly for environments characterized by rich visual state information. Rather than employing a computationally expensive, secondary VLM as the value network, we show that Convolutional Neural Network (CNN) critics, which are sufficient for classical deep RL (Schulman et al., 2017; Raffin et al., 2021a; Huang et al., 2022), can already effectively stabilize training and lead to improved performance. These two designs together yield an important insight with broad implications for scaling VLM RL to long-horizon tasks: by delegating turn-level value estimation to a small module, we can drastically reduce the memory and computational bottlenecks associated with large-scale actor-critic training, making RL fine-tuning significantly more efficient and accessible.

Moreover, we consider an additional algorithmic modification that filters out samples with negative advantages (i.e., $\hat{A}_t < 0$) during training, effectively clipping the advantage at zero, which is referred to as "positive-advantage filtering" in subsequent discussions. This design is motivated by empirical observations that negative-advantage samples can destabilize optimization, a phenomenon reported in both foundation model fine-tuning (Xiong et al.,



(a) Different VLM-based RL methods.

(b) VLM-based RL v.s. classical deep RL.

Figure 4: (a) Comparison of VLM-based RL training methods with training samples limited to 0.38M. PPO with a turn-level CNN critic substantially outperforms critic-free methods, and positive-advantage filtering further stabilizes training. (b) Comparison between VLM-based RL (PPO with a turn-level CNN critic and positive-advantage filtering) and classical RL (PPO training a CNN policy from scratch). VLM-based RL achieves roughly 2× higher sample efficiency, even without action-space engineering. Curves are averaged over at least two independent runs and plotted via EMA smoothing with a factor 0.85; individual runs and additional methods are provided in Fig. 7 and Fig. 9.

2025; Deng et al., 2025) and classical deep RL (Hämäläinen et al., 2020; Jesson et al., 2023). We note that the role of positive versus negative advantage samples in RL training remains an actively studied question (Deng et al., 2025; Carrino et al., 2026; Zhu et al., 2025). Our results provide additional empirical evidence that may help shed light on this direction.

4.2 Comparisons with GRPO and Reinforce++

To better understand different RL algorithmic components, we conduct a controlled experiment on a challenging scenario (i.e., the scenario shown in Fig. 2) in World 1 Level 1 of *Super Mario Land*. The task is to progress as far as possible without dying, beginning with the immediate challenge of jumping over a tall pipe while avoiding two approaching enemies, followed by additional obstacles including platform gaps and additional enemies.

We compare the following candidate methods: (i) GRPO with outcome rewards; (ii) GRPO with outcome rewards and positive-advantage filtering; (iii) GRPO with process rewards; (iv) GRPO with process rewards and positive-advantage filtering; (v) Turn-level Reinforce++; (vi) PPO with a learned CNN critic; and (vii) PPO with a learned turn-level CNN critic and positive-advantage filtering. Their detailed implementations are provided in Section D. We report performance as a function of the number of training samples in Fig. 4a (with the full results in Fig. 7), with Qwen3-VL-8B-Instruct (Bai et al., 2025) as the base model.

Overall, critic-free methods (GRPO and Reinforce++) exhibit unstable learning dynamics and limited performance gains, regardless of reward design or the use of advantage filtering. Particularly, only GRPO with outcome rewards leads to observable improvements after training. In contrast, PPO-based methods achieve substantially stronger and more stable improvements, underscoring the importance of a learned critic for effective credit assignment in this long-horizon setting. Furthermore, positive-advantage filtering improves training stability when combined with PPO. This result highlights the importance of algorithmic components for making RL stable and effective for training VLMs on long-horizon tasks.

5 VLM-Based RL Training versus Classical Deep RL

With stable RL for training VLMs in long-horizon game environments established in Section 4, a natural follow-up question is whether this approach offers tangible advantages over classical deep RL methods, which are also capable of solving similar tasks. In this section, we study this question through the lens of sample efficiency.

Hypothesis: VLM-based RL is more sample-efficient than classical deep RL trained from scratch. Sample efficiency has long been a central challenge in classical deep RL, especially in complex visual environments (Mnih et al., 2015; Badia et al., 2020; Vinyals et al., 2019). A key reason is that such agents must learn both perception and control from scratch, without access to prior knowledge of visual semantics or action dynamics. In contrast, pretrained VLMs already encode rich visual representations and broad world knowledge, providing strong priors for both perception and action. From this perspective, VLM-based RL can be viewed as narrowing the gap between conventional RL agents and human gameplay, where prior knowledge plays a crucial role. We therefore hypothesize that VLM-based RL achieves higher sample efficiency than classical deep RL methods trained from scratch.

Experimental results: VLM-based RL achieves higher sample efficiency with less manual designs. To test this hypothesis, we evaluate classical deep RL on the same task used in Section 4.2. We adopt PPO with a CNN policy as the baseline, given its widespread use and strong empirical performance in prior work. We consider two action-space designs: (i) an *original* action space with 22 actions, covering all valid button combinations of up to two simultaneous presses, and (ii) an *engineered* action space with 8 button combinations designed to better reflect human gameplay. Implementation details are provided in Section E. Notably, this action-space engineering is applied only to classical deep RL; the VLM agent follows the interaction protocol described in Section 3.3.

As shown in Fig. 4b, PPO with the original action space makes only slow progress, likely because exploration is difficult in a large combinatorial action space. Using the engineered action space substantially improves performance by restricting the policy to a smaller and more semantically meaningful set of actions. However, even with this manual design, classical deep RL remains significantly less sample-efficient than VLM-based RL (i.e., PPO with positive-advantage filtering; see Section 4.2), requiring roughly $2\times$ more samples to reach a comparable converged maximum performance.

These results support our hypothesis that pretrained VLMs provide strong inductive biases that reduce the exploration burden in long-horizon tasks. More broadly, they suggest that pretrained VLMs can serve as knowledgeable priors for RL, improving sample efficiency while reducing the need for manual engineering in embodied decision-making problems.

6 Odysseus: An Open and Practical Training Framework

With the key algorithmic ingredients identified in Section 4 and the benefits of training a VLM-based agent established in Section 5, we further extend the scope to finetune a VLM on **multiple levels** of the game simultaneously. We present Odysseus, an open framework for training practical decision-making agents, which integrates supervised fine-tuning initialization and multi-task RL training into a pipeline. While our primary instantiation is in *Super Mario Land*, Odysseus is sufficiently general to inform a broader range of settings.

6.1 Supervised Initialization

Our initial experiments showed that the currently available small open-source VLMs (e.g., Qwen3-VL-8B-Instruct) sometimes lack sufficient domain knowledge and perceptual grounding in *Super Mario Land*, likely because such environments are underrepresented in their pre-training data (which typically contains limited coverage of games). For example, they may struggle to distinguish Mario from enemies or to accurately identify their spatial positions. We therefore begin with a light supervised fine-tuning (SFT) stage to inject domain-specific knowledge and improve environment-specific perception.

In particular, we first curate a dataset covering diverse scenarios of *Super Mario Land*. Specifically, we sample around 5,000 frames across the 10 considered levels from two walkthrough videos that complete the game. For each frame, a stronger model (in our case, GPT-o3) is used to generate teacher responses following the same format described in Section 3.3, including structured `<perception>`, `<reasoning>`, and `<answer>` fields. We qualitatively verify that these CoT annotations are of consistently high quality in terms of both game knowledge and visual perception. Using the sampled images and generated

responses as training data, we then perform standard SFT with cross-entropy loss, while preserving the same input-output format as in Section 3.3.

It is worth noting that the goal of this SFT stage is intentionally lightweight compared with previous works (Tan et al., 2025; Bolton et al., 2025): it focuses on improving domain knowledge and environment perception, rather than optimizing action control, which is deferred to RL. Accordingly, the curated dataset in this work is first significantly smaller in scale. Also, instead of relying on expert trajectories with annotated actions, the sampled frames from walkthrough videos do not inherently provide action labels. These actions are instead generated by the teacher model, which, while strong in perception and reasoning, does not necessarily produce optimal action decisions, as we will demonstrate later. Nevertheless, as shown in our experiments, this lightweight SFT stage improves the effectiveness of subsequent RL training. Furthermore, since game-play videos are far more readily available than collecting expert trajectories with actions, this approach is inherently more scalable.

6.2 Reinforcement Learning with Multi-Task Auto-Curriculum

Building on the environment knowledge and perception capabilities acquired during SFT, we further apply RL to optimize action selection in the environment, thereby improving final performance. Based on the algorithmic findings in Section 4, we adopt the adapted PPO with positive-advantage filtering.

To enable multi-task training, each training batch contains trajectories collected from multiple levels of the game. We further introduce an **auto-curriculum** mechanism to balance learning progress across tasks. Suppose there are K levels involved in training. A batch containing M trajectories in total is denoted by $\mathcal{D} = \bigcup_{k \in [K]} \{\tau_{k,m} : m \in [M_k]\}$, where $\tau_{k,m}$ denotes the m-th trajectory from level k, and M_k is the number of trajectories sampled from level k. A naive strategy is to sample levels uniformly in each batch. However, because different levels can vary substantially in difficulty, uniform level sampling can lead to an undesirable imbalance in the optimization objective. In particular, trajectories from easier levels tend to be longer, since the agent can naturally survive and progress further. As a result, optimization becomes biased toward easier levels, potentially at the expense of performance on harder ones.

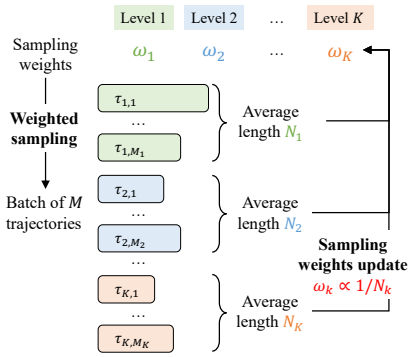


Figure 5: Auto-Curriculum.

To promote balanced learning across levels, we introduce an auto-curriculum mechanism based on inverse trajectory weighting (Li et al., 2024), as demonstrated in Fig. 5. For each level k present in the current batch, let N_k denote the average trajectory length from that level: $N_k = \frac{1}{M_k} \sum_{m \in [M_k]} \text{len}(\tau_{k,m})$. Then, for the next batch, levels are sampled according to $w_k \propto 1/N_k, \forall k \in [K]$. Intuitively, this inverse weighting up-weights levels with shorter trajectories and down-weights those with longer ones. As a result, it approximately balances the number of training samples contributed by each level. This auto-curriculum therefore shifts training toward less-explored levels in a dynamic manner.

7 The Effectiveness of Odysseus

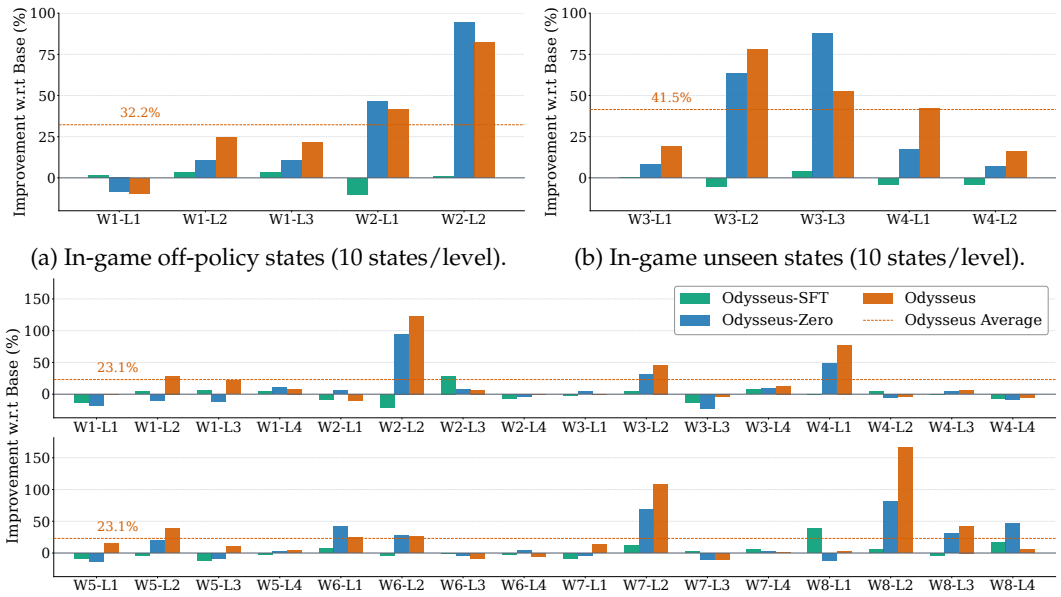
In the following, we demonstrate the effectiveness of Odysseus. All the training are conducted with Qwen3-VL-8B-Instruct (Bai et al., 2025) as the base model and the RL training is performed on the first five levels of the game. Detailed configurations are listed in Section F.

7.1 Superior Training Performances

In Table 2, we first report the performance of flagship frontier VLMs, including both proprietary and open-source models, on the five training levels used for RL training in Odysseus, which exhibit very limited capability in the considered game environment.

Models	Average Progress by (World, Level)					
	(1,1)	(1,2)	(1,3)	(2,1)	(2,2)	Avg.
GPT-5.4	403.62 ± 78.70	67.62 ± 19.84	654.86 ± 96.57	252.75 ± 27.31	173.50 ± 13.15	310.47 ± 25.95
Gemini-3-Flash	529.12 ± 21.51	255.88 ± 97.11	493.43 ± 48.49	187.75 ± 24.90	239.50 ± 21.57	341.14 ± 23.09
Claude-Sonnet-4.6	608.12 ± 66.45	132.00 ± 58.28	502.62 ± 58.61	206.50 ± 18.64	291.75 ± 34.40	348.19 ± 22.61
Qwen3-VL-235B-A22B-Instruct	424.42 ± 13.26	51.69 ± 2.56	511.48 ± 15.58	186.88 ± 5.40	222.77 ± 4.50	279.45 ± 4.36
InternVL3.5-241B-A28B	442.41 ± 14.01	78.12 ± 6.57	390.11 ± 13.87	188.61 ± 5.04	196.40 ± 4.32	259.13 ± 4.36
GLM-4.6V (106B-A12B)	731.28 ± 15.15	364.85 ± 9.97	534.04 ± 9.79	478.46 ± 6.99	455.94 ± 4.81	512.91 ± 4.46
Qwen3-VL-8B-Instruct (base)	513.57 ± 21.08	129.14 ± 9.94	274.20 ± 9.49	238.92 ± 5.31	195.33 ± 4.50	270.23 ± 5.22
Odysseus-SFT (SFT on base)	479.47 ± 16.92	90.92 ± 7.55	300.76 ± 11.21	245.01 ± 5.69	192.69 ± 4.55	261.77 ± 4.57
Odysseus-Zero (RL on base)	1545.50 ± 35.34	1222.69 ± 21.52	1551.57 ± 30.68	1262.18 ± 39.95	1192.71 ± 20.74	1354.93 ± 13.68
Odysseus (RL on SFT)	**1644.43** ± 17.53	**1430.88** ± 22.00	**1603.36** ± 18.62	**1352.30** ± 14.14	**1528.51** ± 25.75	**1511.90** ± 8.95
Maximum	2351	2190	2336	2510	2191	2315.6

Table 2: Comparisons of Odysseus with frontier models across five levels used for RL training. We measure level progress, defined as the x-axis distance traveled by Mario from the start of each level. Results are aggregated over runs and reported as mean $\pm$ standard error. The last row indicates the maximum achievable progress on each level.



(a) In-game off-policy states (10 states/level).

(b) In-game unseen states (10 states/level).

(c) Unseen game *Super Mario Bros.* (32 levels)

Figure 6: Evaluation of Odysseus under three generalization settings: in-game off-policy states (a); in-game unseen states (b); and cross-game scenarios (c). Improvement is measured relative to the base pretrained VLM. For (a) and (b), the histograms report the average performance over the 10 states for each level, while for (c), performance is reported individually from the start of each level. The dotted horizontal line indicates the average improvement of Odysseus over the base model across the scenarios in the corresponding subplot. Consistent gains across all settings demonstrate the strong generalization capability of Odysseus.

We report the final performance of Odysseus in Table 2, together with two ablations: Odysseus-SFT, initialized via supervised fine-tuning, and Odysseus-Zero, trained with RL directly from the base model without SFT initialization. After training, Odysseus achieves substantial improvements, typically tripling—and in many cases increasing by an order of magnitude—the average level progress compared to the base model, while also significantly outperforming frontier models. In particular, in terms of average level progress, Odysseus improves over the base model by 5.59$\times$ and over the best-performing frontier model (GLM-4.6V) by 2.95$\times$. These results highlight the effectiveness of Odysseus.

Furthermore, although Odysseus-SFT alone does not provide observable gains over the base model, training on top of this initialization, i.e., Odysseus, consistently yields better performance than Odysseus-Zero across all levels. This observation supports the useful role of SFT initialization in enabling more effective RL training.

7.2 Generalizations in Games

We evaluate the generalization capabilities of Odysseus under three progressively challenging settings. First, we consider **in-game off-policy** evaluation, where the agent is tested on 50 manually curated states from the five training levels (i.e., 10 states each level). Although from the same training levels, they are sampled independently of the agent's trajectories and thus induce a state distribution shift. Second, we assess **in-game generalization** by evaluating the agent on 50 manually collected states from the remaining five levels of *Super Mario Land* (i.e., 10 states each level), which are entirely unseen during RL training but share the same game mechanics and visual structure. Finally, we examine **cross-game generalization** by testing the agent on all 32 levels of another game *Super Mario Bros.*, representing a more substantial domain shift in terms of level design and visual appearance. Together, these settings provide a comprehensive evaluation of robustness to distribution shifts, ranging from off-policy state variations to unseen levels and entirely new games. Details of the evaluation setup are provided in Section F.3.

The relative improvements of the Odysseus-series models on the first two in-game settings are reported in Fig. 6a and Fig. 6b, where Odysseus achieves average improvements of 32.2% and 41.5%, respectively. The full results for the cross-game evaluation are presented in Fig. 6c, where Odysseus yields a relative improvement of 23.1% over the base model on average. Overall, these results demonstrate that, despite being trained on only five levels of *Super Mario Land*, Odysseus exhibits clear signs of both in-game and cross-game generalization, highlighting the potential of this framework in building general agents.

7.3 Performances in General Domains

Finally, given the extensive training in the game environment (over tens of millions of interaction samples), a natural concern is that the model may overfit and lose its general capabilities. To assess this, we evaluate the Odysseus-series models on a set of general multimodal benchmarks, including STEM-oriented tasks (MMMU (Yue et al., 2024) and MathVision (Wang et al., 2024)) and real-world reasoning tasks (RealWorldQA (xAI, 2024)). The results, reported in Table 3, show that compared to the base model, the Odysseus-series models retain comparable performances. This suggests that Odysseus can effectively inject new decision-making capabilities without compromising the model's general-purpose strengths, highlighting its potential as a foundation for general-purpose agents.

Model	MMMUval	MathVision	RealWorldQA
Qwen3-VL-8B-Instruct	69.00	54.64	71.11
Odysseus-SFT	70.44	55.00	71.37
Odysseus-Zero	70.22	54.44	70.72
Odysseus	70.77	53.52	71.11

Table 3: General-purpose benchmarks.

8 Conclusions

In this work, we study the problem of training VLMs for long-horizon decision-making tasks, using the video game *Super Mario Land* as a testbed, which requires 100+ interaction turns per episode. We first introduce an adapted PPO algorithm with a lightweight turn-level critic, which substantially improves training stability and sample efficiency over critic-free methods. Building on this, we show that fine-tuning pretrained VLMs via RL is significantly more sample-efficient than training agents from scratch with classical deep RL, highlighting the value of the encoded knowledge priors. Leveraging these insights, we propose Odysseus, an open training framework for practical VLM agents that combines SFT initialization with multi-task RL training. Odysseus enables stable training across multiple levels of the game simultaneously, yielding substantial performance gains over the base model and outperforming frontier models by large margins. Moreover, the resulting agents exhibit emergent generalization to both in-domain and out-of-domain settings, while retaining strong general-purpose capabilities. Overall, our results demonstrate that with an appropriate recipe, RL can be effectively scaled to long-horizon decision-making tasks for VLMs, providing a promising path toward more capable embodied agents.

Acknowledgment

The authors thank Princeton Language and Intelligence (PLI) and Princeton AI Lab for their support of this work, including computational resources and API credits, as well as many members of these groups for helpful discussions and suggestions. CJ acknowledges the support from NSF-OAC-2411299, NSF-IIS-2239297, and Sloan Research Fellowship.

References

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce-style optimization for learning from human feedback in llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267, 2024.

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Mengyuan Yan, and Andy Zeng. Do as i can and not as i say: Grounding language in robotic affordances. In *arXiv preprint arXiv:2204.01691*, 2022.

Adrià Puigdomènech Badia, Bilal Piot, Steven Kapturowski, Pablo Sprechmann, Alex Vitvitskyi, Zhaohan Daniel Guo, and Charles Blundell. Agent57: Outperforming the atari human benchmark. In *International conference on machine learning*, pp. 507–517. PMLR, 2020.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Bowen Baker, Ilge Akkaya, Peter Zhokov, Joost Huizinga, Jie Tang, Adrien Ecoffet, Brandon Houghton, Raul Sampedro, and Jeff Clune. Video pretraining (vpt): Learning to act by watching unlabeled online videos. *Advances in Neural Information Processing Systems*, 35: 24639–24654, 2022.

Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of artificial intelligence research*, 47:253–279, 2013.

Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. pi_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Adrian Bolton, Alexander Lerchner, Alexandra Cordell, Alexandre Moufarek, Andrew Bolt, Andrew Lampinen, Anna Mitenkova, Arne Olav Hallingstad, Bojan Vujatovic, Bonnie Li, et al. Sima 2: A generalist embodied agent for virtual worlds. *arXiv preprint arXiv:2512.04797*, 2025.

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. *arXiv preprint arXiv:2508.04280*, 2025.

Gabriele Carrino, Andrea Sassella, Nicolò Brunello, Federico Toschi, and Mark James Carman. Are complicated loss functions necessary for teaching llms to reason? *arXiv preprint arXiv:2603.18756*, 2026.

Jun Shern Chan, Neil Chowdhury, Oliver Jaffe, James Aung, Dane Sherburn, Evan Mays, Giulio Starace, Kevin Liu, Leon Maksin, Tejal Patwardhan, et al. Mle-bench: Evaluating machine learning agents on machine learning engineering. *arXiv preprint arXiv:2410.07095*, 2024.

Hanyang Chen, Mark Zhao, Rui Yang, Qinwei Ma, Ke Yang, Jiarui Yao, Kangrui Wang, Hao Bai, Zhenhailong Wang, Rui Pan, et al. Era: Transforming vlms into embodied agents via embodied prior learning and online reinforcement learning. *arXiv preprint arXiv:2510.12693*, 2025a.

Liang Chen, Hongcheng Gao, Tianyu Liu, Zhiqi Huang, Flood Sung, Xinyu Zhou, Yuxin Wu, and Baobao Chang. G1: Bootstrapping perception and reasoning abilities of vision-language model via reinforcement learning. *arXiv preprint arXiv:2505.13426*, 2025b.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34: 15084–15097, 2021.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Marc Deisenroth and Carl E Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pp. 465–472, 2011.

Wenlong Deng, Yi Ren, Muchen Li, Danica J Sutherland, Xiaoxiao Li, and Christos Thrampoulidis. On the effect of negative gradient in group relative deep reinforcement optimization. *arXiv preprint arXiv:2505.18830*, 2025.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36:28091–28114, 2023.

Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International conference on machine learning*, pp. 1329–1338. PMLR, 2016.

Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar, Yuncong Yang, Haoyi Zhu, Andrew Tang, De-An Huang, Yuke Zhu, and Anima Anandkumar. Minedojo: Building open-ended embodied agents with internet-scale knowledge. *Advances in Neural Information Processing Systems*, 35:18343–18362, 2022.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. *arXiv preprint arXiv:2505.10978*, 2025.

ARC Foundation. Arc-agi-3: A new challenge for frontier agentic intelligence. *arXiv preprint arXiv:2603.24621*, 2026.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

Perttu Hämäläinen, Amin Babadi, Xiaoxiao Ma, and Jaakko Lehtinen. Ppo-cma: Proximal policy optimization with covariance matrix adaptation. In *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6. IEEE, 2020.

Shuo He, Lang Feng, Qi Wei, Xin Cheng, Lei Feng, and Bo An. Hierarchy-of-groups policy optimization for long-horizon agentic tasks. *arXiv preprint arXiv:2602.22817*, 2026.

Jian Hu. Reinforce++: A simple and efficient approach for aligning large language models. *arXiv e-prints*, pp. arXiv–2501, 2025.

Lanxiang Hu, Mingjia Huo, Yuxuan Zhang, Haoyang Yu, Eric P Xing, Ion Stoica, Tajana Rosing, Haojian Jin, and Hao Zhang. lmgame-bench: How good are llms at playing games? *arXiv preprint arXiv:2505.15146*, 2025.

Shengyi Huang, Rousslan Fernand Julien Dossa, Antonin Raffin, Anssi Kanervisto, and Weixun Wang. The 37 implementation details of proximal policy optimization. *The ICLR Blog Track 2023*, 2022.

Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems*, 34: 1273–1286, 2021.

Andrew Jesson, Chris Lu, Gunshi Gupta, Nicolas Beltran-Velez, Angelos Filos, Jakob Nicolaus Foerster, and Yarin Gal. Relu to the rescue: Improve your on-policy actor-critic with positive advantages. *arXiv preprint arXiv:2306.01460*, 2023.

Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, et al. Model-based reinforcement learning for atari. *arXiv preprint arXiv:1903.00374*, 2019.

Seth Karten, Jake Grigsby, Tersoo Upaa Jr, Junik Bae, Seonghun Hong, Hyunyoung Jeong, Jaeyoon Jung, Kun Kerdthaisong, Gyungbo Kim, Hyeokgi Kim, Yujin Kim, Eunju Kwon, Dongyu Liu, Patrick Mariglia, Sangyeon Park, Benedikt Schink, Xianwei Shi, Anthony Sistilli, Joseph Twin, Arian Urdu, Matin Urdu, Qiao Wang, Ling Wu, Wenli Zhang, Kunsheng Zhou, Stephanie Milani, Kiran Vodrahalli, Amy Zhang, Fei Fang, Yuke Zhu, and Chi Jin. The pokeagent challenge: Competitive and long-context learning at scale. In *NeurIPS Competition Track*, April 2025a.

Seth Karten, Andy Luu Nguyen, and Chi Jin. Pokéchamp: an expert-level minimax language agent. In *International Conference on Machine Learning*, pp. 29205–29222. PMLR, 2025b.

Seth Karten, Joel Zhang, Tersoo Upaa Jr, Ruirong Feng, Wenzhe Li, Chengshuai Shi, Chi Jin, and Kiran Vodrahalli. Continual harness: Online adaptation for self-improving foundation agents. *arXiv preprint arXiv:2605.09998*, 2026.

Kuang-Huei Lee, Ofir Nachum, Mengjiao Sherry Yang, Lisa Lee, Daniel Freeman, Sergio Guadarrama, Ian Fischer, Winnie Xu, Eric Jang, Henryk Michalewski, et al. Multi-game decision transformers. *Advances in neural information processing systems*, 35:27921–27936, 2022.

Jiazheng Li, Yawei Wang, Qiaojing Yan, Yijun Tian, Zhichao Xu, Huan Song, Panpan Xu, and Lin Lee Cheong. Salt: Step-level advantage assignment for long-horizon agents via trajectory graph. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 4709–4725, 2026a.

Junbo Li, Peng Zhou, Rui Meng, Meet P Vadera, Lihong Li, and Yang Li. Turn-ppo: Turn-level advantage estimation with ppo for improved multi-turn rl in agentic llms. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 6227–6243, 2026b.

Wenzhe Li, Hao Luo, Zichuan Lin, Chongjie Zhang, Zongqing Lu, and Deheng Ye. A survey on transformers in reinforcement learning. *arXiv preprint arXiv:2301.03044*, 2023.

Wenzhe Li, Zihan Ding, Seth Karten, and Chi Jin. Fightladder: A benchmark for competitive multi-agent reinforcement learning. *arXiv preprint arXiv:2406.02081*, 2024.

Zihan Lin, Xiaohan Wang, Jie Cao, Jiajun Chai, Li Wang, Xiaodong Lu, Wei Lin, Ran He, and Guojun Yin. Resrl: Boosting llm reasoning via negative sample projection residual reinforcement learning. *arXiv preprint arXiv:2605.00380*, 2026.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. *arXiv preprint arXiv:2410.07864*, 2024.

Loïc Magne, Anas Awadalla, Guanzhi Wang, Yinzhen Xu, Joshua Belofsky, Fengyuan Hu, Joohwan Kim, Ludwig Schmidt, Georgia Gkioxari, Jan Kautz, et al. Nitrogen: An open foundation model for generalist gaming agents. *arXiv preprint arXiv:2601.02427*, 2026.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Igor Mordatch and Pieter Abbeel. Emergence of grounded compositional language in multi-agent populations. *arXiv preprint arXiv:1703.04908*, 2017.

Yao Mu, Qinglong Zhang, Mengkang Hu, Wenhai Wang, Mingyu Ding, Jun Jin, Bin Wang, Jifeng Dai, Yu Qiao, and Ping Luo. Embodiedgpt: Vision-language pre-training via embodied chain of thought. *Advances in Neural Information Processing Systems*, 36:25081–25094, 2023.

Dang Nguyen, Jian Chen, Yu Wang, Gang Wu, Namyong Park, Zhengmian Hu, Hanjia Lyu, Junda Wu, Ryan Aponte, Yu Xia, et al. Gui agents: A survey. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 22522–22538, 2025.

Alex Nichol, Vicki Pfau, Christopher Hesse, Oleg Klimov, and John Schulman. Gotta learn fast: A new benchmark for generalization in rl. *arXiv preprint arXiv:1804.03720*, 2018.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Dongmin Park, Minkyu Kim, Beongjun Choi, Junhyuck Kim, Keon Lee, Jonghyun Lee, Inkyu Park, Byeong-Uk Lee, Jaeyoung Hwang, Jaewoo Ahn, et al. Orak: A foundational benchmark for training and evaluating llm agents on diverse video games. *arXiv preprint arXiv:2506.03610*, 2025.

Maria Abi Raad, Arun Ahuja, Catarina Barros, Frederic Besse, Andrew Bolt, Adrian Bolton, Bethanie Brownfield, Gavin Buttimore, Max Cant, Sarah Chakera, et al. Scaling instructable agents across many simulated worlds. *arXiv preprint arXiv:2404.10179*, 2024.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of machine learning research*, 22(268):1–8, 2021a.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021b. URL http://jmlr.org/papers/v22/20-1364.html.

Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Max Schwarzer, Ankesh Anand, Rishab Goel, R Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. *arXiv preprint arXiv:2007.05929*, 2020.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. URL https://arxiv.org/abs/1912.01734.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning Text and Embodied Environments for Interactive Learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021. URL https://arxiv.org/abs/2010.03768.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, pp. 387–395. Pmlr, 2014.

Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Andrew Szot, Bogdan Mazoure, Omar Attia, Aleksei Timofeev, Harsh Agrawal, Devon Hjelm, Zhe Gan, Zsolt Kira, and Alexander Toshev. From multimodal llms to generalist embodied agents: Methods and lessons. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 10644–10655, 2025.

Weihao Tan, Xiangyang Li, Yunhao Fang, Heyuan Yao, Shi Yan, Hao Luo, Tenglong Ao, Huihui Li, Hongbin Ren, Bairen Yi, et al. Lumine: An open recipe for building generalist agents in 3d open worlds. *arXiv preprint arXiv:2511.08892*, 2025.

Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, et al. Gemini robotics: Bringing ai into the physical world. *arXiv preprint arXiv:2503.20020*, 2025.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *nature*, 575 (7782):350–354, 2019.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

Kangrui Wang, Pingyue Zhang, Zihan Wang, Yaning Gao, Linjie Li, Qineng Wang, Hanyang Chen, Chi Wan, Yiping Lu, Zhengyuan Yang, Lijuan Wang, Ranjay Krishna, Jiajun Wu, Li Fei-Fei, Yejin Choi, and Manling Li. Vagen:reinforcing world model reasoning for multi-turn vlm agents, 2025a. URL https://arxiv.org/abs/2510.16907.

Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. Measuring multimodal mathematical reasoning with math-vision dataset. *Advances in Neural Information Processing Systems*, 37:95095–95169, 2024.

Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, et al. Ragen: Understanding self-evolution in llm agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*, 2025b.

Zihao Wang, Xujing Li, Yining Ye, Junjie Fang, Haoming Wang, Longxiang Liu, Shihao Liang, Junting Lu, Zhiyong Wu, Jiazhan Feng, et al. Game-tars: Pretrained foundation models for scalable generalist multimodal game agents. *arXiv preprint arXiv:2510.23691*, 2025c.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

xAI. Realworldqa: A benchmark for real-world spatial understanding. 2024. URL https://huggingface.co/datasets/xai-org/RealworldQA.

Wei Xiong, Jiarui Yao, Yuhui Xu, Bo Pang, Lei Wang, Doyen Sahoo, Junnan Li, Nan Jiang, Tong Zhang, Caiming Xiong, et al. A minimalist approach to llm reasoning: from rejection sampling to reinforce. *arXiv preprint arXiv:2504.11343*, 2025.

John Yang, Carlos E Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering. *Advances in Neural Information Processing Systems*, 37:50528–50652, 2024.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*, 2022.

Weirui Ye, Shaohuai Liu, Thanard Kurutach, Pieter Abbeel, and Yang Gao. Mastering atari games with limited data. *Advances in neural information processing systems*, 34:25476–25488, 2021.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pp. 1094–1100. PMLR, 2020.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, et al. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9556–9567, 2024.

Yuexiang Zhai, Hao Bai, Zipeng Lin, Jiayi Pan, Shengbang Tong, Yifei Zhou, Alane Suhr, Saining Xie, Yann LeCun, Yi Ma, et al. Fine-tuning large vision-language models as decision-making agents via reinforcement learning. *Advances in neural information processing systems*, 37:110935–110971, 2024.

Alex L Zhang, Thomas L Griffiths, Karthik R Narasimhan, and Ofir Press. Videogamebench: Can vision-language models complete popular video games? *arXiv preprint arXiv:2505.18134*, 2025.

Xinyu Zhu, Mengzhou Xia, Zhepei Wei, Wei-Lin Chen, Danqi Chen, and Yu Meng. The surprising effectiveness of negative reinforcement in llm reasoning. *arXiv preprint arXiv:2506.01347*, 2025.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pp. 2165–2183. PMLR, 2023.

A Extended Related Works

Games and Simulated Environments. Games and simulated environments have long served as controlled testbeds during the development of modern machine learning. In classical deep RL, benchmarks such as ALE (Bellemare et al., 2013) and MuJoCo (Duan et al., 2016) played a central role in studying algorithms learning from interactions with the environments. Later benchmarks extend this paradigm to more complex and advanced settings, including multi-agent RL (Mordatch & Abbeel, 2017; Berner et al., 2019; Vinyals et al., 2019; Li et al., 2024), multi-task generalization (Nichol et al., 2018; Yu et al., 2020), embodied tasks (Shridhar et al., 2020; 2021; Liu et al., 2023), and open-world environments (Fan et al., 2022; Tan et al., 2025). More recently, several works have begun to use video games as direct testbeds for foundation models (Hu et al., 2025; Karten et al., 2025a; Zhang et al., 2025; Park et al., 2025; Foundation, 2026). However, the recent exploration of RL for VLMs in simulated environments is mostly focused on short-horizon scenarios, such as AlfWorld (Shridhar et al., 2021), Sokoban, and FrozenLake (Wang et al., 2025a). In contrast to these settings, we focus on *Super Mario Land* as a compact but appealing testbed for VLMs in long-horizon embodied control: it imposes substantially richer spatial grounding and closed-loop control than short-horizon gridworld-style tasks, while remaining lightweight and easy to scale for controlled studies compared with large open-world simulators.

Foundation Models for Decision-Making. The integration of foundation models into sequential decision-making has evolved through several distinct paradigms. Initial efforts treated control primarily as a sequence modeling problem, training Transformers from scratch on offline trajectory data (Janner et al., 2021; Chen et al., 2021; Li et al., 2023). This approach demonstrated strong potential for task generalization (Lee et al., 2022; Reed et al., 2022) and skill composition in open-ended environments (Baker et al., 2022; Fan et al., 2022). As large language and vision-language models advanced, a second line of work emerged that leverages frozen large models with context engineered states and agentic scaffolding or harnesses in long context sequential decision-making tasks, such as navigating complex video games like Pokémon (Karten et al., 2025a;b; 2026; Comanici et al., 2025) and Minecraft (Wang et al., 2023), or facilitating robotics manipulation tasks (Huang et al., 2023; Ahn et al., 2022). More recently, the field has shifted toward fine-tuning pretrained foundation models directly for embodied control. This has yielded highly capable agents in both robotic manipulation (Black et al., 2024; Liu et al., 2024; Team et al., 2025) and cross-game generalization (Tan et al., 2025; Bolton et al., 2025; Wang et al., 2025c). While highly promising, these fine-tuning approaches heavily depend on SFT with large amounts of high-quality, action-labeled demonstration data. In contrast, our work focuses on the RL perspective, investigating how to effectively adapt foundation models for long-horizon decision-making tasks without relying on extensive SFT data.

RL from Classical Control to Foundation-Model Agents. RL has a long history in training neural networks for sequential decision-making, spanning policy-gradient (Williams, 1992; Silver et al., 2014), value-based (Mnih et al., 2015; Van Hasselt et al., 2016) and actor-critic (Schulman et al., 2017; Haarnoja et al., 2018) methods. Particularly, sample efficiency is a key challenge for RL algorithms as they require far more environmental interactions than humans during training time, motivating a line of research on sample-efficient RL (Deisenroth & Rasmussen, 2011; Kaiser et al., 2019; Hafner et al., 2019; Schwarzer et al., 2020; Ye et al., 2021). More recently, RL has also become a central ingredient for improving foundation models, especially in reasoning-oriented settings with verifiable rewards, through REINFORCE-style and critic-free methods (Ahmadian et al., 2024; Hu, 2025; Shao et al., 2024). A growing body of work further studies RL for multi-turn language and vision-language agents (Zhai et al., 2024; Chen et al., 2025a; Wang et al., 2025b;a; Feng et al., 2025; Li et al., 2026a; Feng et al., 2025; He et al., 2026), often introducing specialized machinery for trajectory decomposition, turn-level and token-level advantage estimation, or hierarchical credit assignment, and evaluating on environments with relatively short horizons (20–30 turns). In contrast to these works, we focus specifically on long-horizon, visually grounded embodied environments that require 100+ turns of interaction with chain-of-thought rea-

soning, and through rigorous ablations, we show that a comparatively simple PPO-based approach with the right critic design is sufficient to make RL stable and effective.

B Future Work and Discussion

In the following, we discuss several directions that we believe are interesting for further investigation but are beyond the scope of the current work.

Incorporating More Advanced Harnesses. As discussed in Section 3.3, we deliberately adopt minimal scaffolding in this work—i.e., providing the model only with task instructions and the current visual observation—to focus on RL training and, in particular, to isolate the capabilities that can be acquired by directly fine-tuning the VLM through RL. Our results show that, even under this constrained interaction interface, an appropriate training recipe can substantially improve the game-playing capabilities of VLMs.

More advanced harnesses, such as those incorporating explicit interaction history, reusable skills, or external memory, could further improve performance. At the same time, they may introduce additional training challenges, including higher computational costs due to longer contexts and increased complexity in credit assignment. Exploring how such scaffolding can be effectively integrated with long-horizon VLM-RL remains an interesting direction for future work.

Further Investigations on the Design of Critics. In Odysseus, we use a lightweight CNN critic, whereas prior works such as Wang et al. (2025a); Zhai et al. (2024); Bredis et al. (2025) adopt VLM-based critics. This choice reflects a tradeoff between computational efficiency and representational capacity: VLM critics can leverage richer information, while CNN critics are considerably cheaper and easier to scale to long-horizon online RL.

The relative advantages of different critic designs may depend on the task and training setting. In visual control tasks such as Mario, the current frame can already provide strong signals about local progress, and our results suggest that a lightweight CNN critic can provide sufficiently informative value estimates even when paired with a much richer VLM actor. A more systematic study of critic architectures, inputs, and capacity in VLM-RL would therefore be an interesting direction for future work.

More Careful Handling of Negative-Advantage Samples. The positive-advantage filtering used in Odysseus provides a simple and effective mechanism for stabilizing long-horizon VLM-RL in our experiments. Intuitively, it motivates learning toward actions that perform better than the critic's baseline, bearing some resemblance to expert-iteration-style methods that preferentially reinforce improved behaviors. Unlike standard expert iteration, however, the filtering in Odysseus operates at the turn level, and the retained samples remain weighted by their estimated advantages.

While this simple strategy works well in our setting, discarding negative-advantage samples may also leave potentially useful learning signals underutilized. Recent work such as Lin et al. (2026) has begun to investigate related issues on how to better leverage conflicting signals in RL for foundation models. Developing more principled approaches for handling negative-advantage samples and gradient conflicts could further improve the stability and sample efficiency of long-horizon VLM-RL.

C Details of the Interaction Protocol

First, the full prompt used for instructing the VLM agent to interact with the game environment is provided in the following.

Prompt for the Agent

You are playing Super Mario Land.

The goal is to progress through levels, collect coins and power-ups when safe, and ultimately finish the game by rescuing Princess Daisy.

You can control the game by pressing buttons on the Game Boy.

Available buttons:
- 'a': Jump (used to make Mario jump)
- 'b': Run/Shoot (hold to run faster or shoot fireballs if available)
- 'up': Climb ladders or vines (if present)
- 'down': Crouch or enter pipes (when standing on a pipe)
- 'left': Move Mario left
- 'right': Move Mario right
- 'noop': Do nothing (used to wait for a brief period without performing any action)

Please analyze the game screen and decide which buttons to press to progress.

Return your answer as follows:
1. Button sequence: a list of buttons to press simultaneously
2. Each button should be one of: 'a', 'b', 'up', 'down', 'left', 'right', 'noop'

First describe what you see on the screen in <perception></perception>. Then, in <reasoning></reasoning>, break down your reasoning step by step, justifying each action you consider. Output your final action in <answer>['button1', 'button2', ...]</answer>.

The maximum number of buttons you can press simultaneously in one turn is 2.

At each turn, the agent observes the current game frame and the instruction prompt, and produces an action in the prescribed format. Since the original game resolution of 160×144 is relatively low compared with the visual resolutions used during modern VLM pre-training, we up-sample the image by a factor of 8 to 1280×1152. The final action is extracted from the `<answer></answer>` field and executed in the game environment as button presses. As described in Section 3.3, we adopt a frame-skipping mechanism: when the action includes a jump (i.e., `a`), the environment advances for 15 frames; otherwise, it advances for 5 frames, including the case of `noop`, where the game proceeds without any button input.

D Details of RL Algorithms for VLM Training

In this section, we provide the implementation details of the RL algorithms discussed in Section 4, including the proposed adapted version of PPO.

D.1 Advantage Constructions

Provided with a dataset batch $\mathcal{D}$ of trajectories collected via the policy from the previous training step, denoted as π_{old}, all methods are instantiated with the same surrogate loss (Schulman et al., 2017):

$$\mathcal{L}(\theta) = \mathbb{E}_{o_t, a_t \sim \mathcal{D}} \left[\min \left(\frac{\pi_\theta(a_t|o_t)}{\pi_{\theta_{\text{old}}}(a_t|o_t)} \hat{A}_t, \text{clip} \left(\frac{\pi_\theta(a_t|o_t)}{\pi_{\theta_{\text{old}}}(a_t|o_t)}, 1 - \epsilon_{\text{low}}, 1 + \epsilon_{\text{high}} \right) \hat{A}_t \right) \right],$$

where ϵ_{low} and ϵ_{high} are the clipping factors while $\hat{A}_t$ is the advantage estimator at turn t. The key differences of the algorithms are in how advantages are constructed, which are detailed in the following.

PPO with a Turn-level Critic. We first discuss the adapted version of PPO proposed in this work. A *turn-level* critic model $V_\phi(o_t)$ is learned to approximate the value of policy starting from s_t, and use the discounted return-to-go $\hat{R}_t = \sum_{i \geq t} \gamma^{i-t} r_i$ as the target value:

$$\mathcal{L}_V(\phi) = \mathbb{E}_{o_t \sim \mathcal{D}} \left[\text{SmoothL}_1 \left(V_\phi(o_t) - \hat{R}_t \right) \right].$$

The raw per-turn advantage is first computed as

$$\tilde{A}_t = \hat{R}_t - V_\phi(o_t).$$

We perform one more batch-level variance normalization over it to obtain the advantage as

$$\hat{A}_t = \frac{\tilde{A}_t}{\sigma\left(\{\tilde{A}_{t'} : o_{t'} \in \mathcal{D}\}\right)},$$

where $\sigma(\cdot)$ denotes standard deviation.

When performing positive-advantage filtering, we instead keep the signed advantages and normalize them over them:

$$\hat{A}_t = \frac{\max(\tilde{A}_t, 0)}{\sigma\left(\{\tilde{A}_{t'} : \tilde{A}_{t'} > 0, o_{t'} \in \mathcal{D}\}\right)}. \tag{1}$$

GRPO with Outcome Rewards. For GRPO with outcome rewards, all turns within the same trajectory share a common trajectory-level outcome return. Specifically, for a trajectory

$$\tau = (o_0, a_0, r_0, \cdots, o_{T-1}, a_{T-1}, r_{T-1}),$$

we define the outcome return as the cumulative reward over the full trajectory:

$$\hat{R}^{\text{out}}(\tau) = \sum_{i=0}^{T-1} r_i, \tag{2}$$

which is equivalent to setting the discount factor $\gamma = 1$ when calculating return-to-go. This same outcome return is assigned to every turn t in the trajectory. Following standard GRPO (Shao et al., 2024), we standardize the trajectory-level outcome return over the batch:

$$\hat{A}_t = \frac{\hat{R}^{\text{out}}(\tau) - \mathbb{E}_{\tau' \in \mathcal{D}}[\hat{R}^{\text{out}}(\tau')]}{\sigma\left(\{\hat{R}^{\text{out}}(\tau') : \tau' \in \mathcal{D}\}\right)}. \tag{3}$$

When performing positive advantage filtering, it is further set that $\hat{A}_t = \max(0, \hat{A}_t)$.

GRPO with Process Rewards. For GRPO with process rewards, following Shao et al. (2024), we first standardize turn rewards over the current batch before constructing the return-to-go as

$$\tilde{r}_t = \frac{r_t - \mathbb{E}_{r_{t'} \in \mathcal{D}}[r_{t'}]}{\sigma(\{r_{t'} : r_{t'} \in \mathcal{D}\})}.$$

The per-turn training signal as the un-discounted return-to-go of the standardized process rewards:

$$\hat{A}_t = \sum_{i=t}^{T-1} \tilde{r}_i.$$

When performing positive advantage filtering, it is further set that $\hat{A}_t = \max(0, \hat{A}_t)$.

Parameter	Value
max turns	80
top p	1
temperature	1
max response length	1024
number of trajectories per batch	128
optimization epochs per training step	1
mini-batch size	1024
discount factor γ	0.95
policy clip range $\epsilon_{\text{high}}, \epsilon_{\text{low}}$	$0.28, 0.2$
learning rate	5×10^{-5}
learning rate scheduler	constant

Table 4: Hyper-parameters for comparisons of VLM-based RL algorithms.

Reinforce++. For Reinforce++ (Hu, 2025), we use the same turn-level discounted return-to-go as PPO, i.e., $\hat{R}_t = \sum_{i \geq t} \gamma^{i-t} r_i$, and standardize it over the entire batch as

$$\hat{A}_t = \frac{\hat{R}_t - \mathbb{E}_{o_{t'} \in \mathcal{D}}[\hat{R}_{t'}]}{\sigma\left(\{\hat{R}_{t'} : o_{t'} \in \mathcal{D}\}\right)}.$$

D.2 Experimental Details and Additional Results

In Table 4, we provide the detailed configurations of experiments reported in Section 4.2, which are shared across the tested algorithms. During all experiments, the training is performed on all components of the base model (i.e., Qwen3-VL-8B-Instruct), including vision encoder, multi-modal projector, and language model backbone. For the adapted PPO, the network architecture for the CNN-based critic and its training hyper-parameters are further detailed in Section F, which are shared between experiments in Section 4.2 and Section 7. Additional experimental results, including all seven evaluated methods and their individual runs, are provided in Fig. 7.

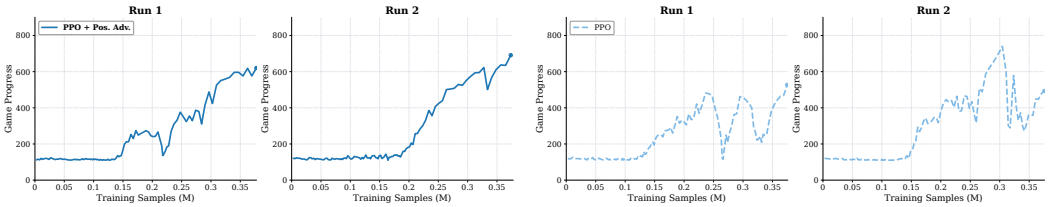
In addition, we incorporate three more baselines: VAGEN (Wang et al., 2025a), VL-DAC (Bredis et al., 2025), and Turn-PPO (Li et al., 2026b), all of which use the same VLM backbone as the actor for the critic. The default hyperparameters from their implementations are utilized. Due to the substantially higher computational cost of VLM-based critics and our available compute budget, we run each of these baselines once. Despite their more expressive critic architectures and higher computational requirement, VAGEN and Turn-PPO show limited progress in our setting, while VL-DAC achieves performance closer to that of Odysseus.

Moreover, to further evaluate the generality of the proposed method, we conduct experiments with a different reward signal. Instead of the fully dense progress reward described in Section 3.2, we use a coarser reward based on predefined milestones, such as passing pipes, gaps, and enemies, where rewards are provided only when these milestones are reached. As shown in Fig. 8, Odysseus still exhibits stable training under this coarser reward signal, with only slightly slower learning than under the dense reward, while continuing to substantially outperform GRPO.

E Details of Comparisons with Classical Deep RL

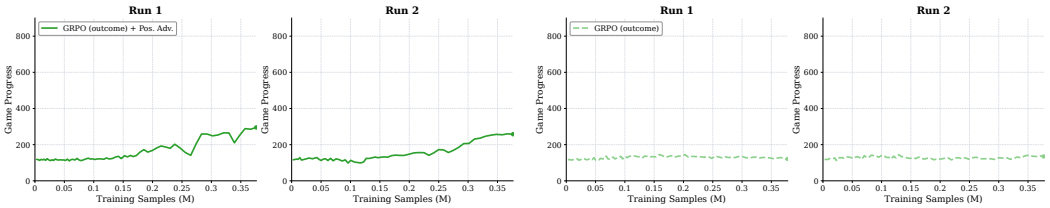
In this section, we provide the implementation details of the classical deep RL baselines used in Section 5, with the complete experimental results provided in Fig. 9.

Environment and Action Spaces. The environment used for classical deep RL training is set to be exactly the same as that used in VLM training to ensure comparable results, including but not limited to, rewards and frame-skipping mechanisms. The major differences



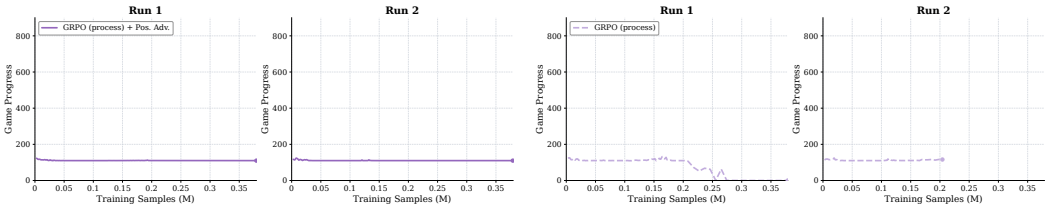
(a) PPO + Pos. Adv.

(b) PPO.



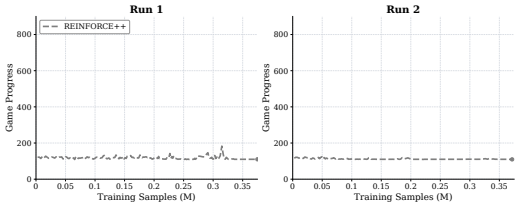
(c) GRPO (outcome) + Pos. Adv.

(d) GRPO (outcome).

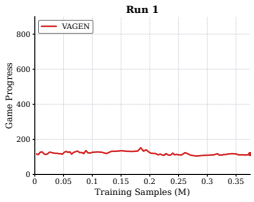


(e) GRPO (process) + Pos. Adv.

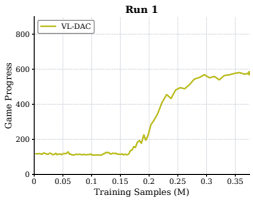
(f) GRPO (process).



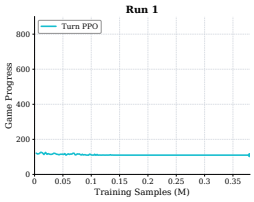
(g) Reinforce++.



(h) VAGEN.



(i) VL-DAC.



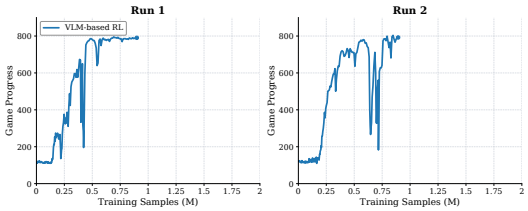
(j) Turn-PPO.

Figure 7: Comparison of VLM-based RL training methods. Individual runs are plotted separately without smoothing.

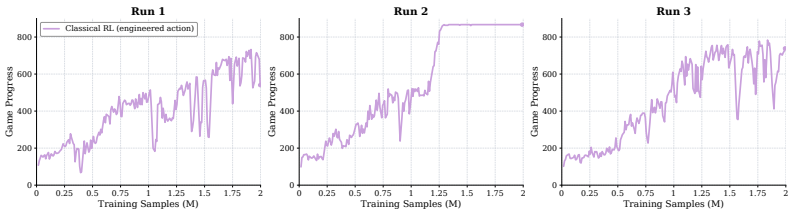
lie in the action spaces. As mentioned in Section 3.3, VLMs are allowed to output up to press two buttons at the same time out from a total list of seven (i.e., `a`, `b`, `up`, `down`, `left`, `right`, `noop`). For classical RL, the "original" action space (with a size of 22) contains `noop` and up to two buttons from the remaining list. The engineered action space is more tailored towards human game-playing behaviors, which is illustrated in Table 5.

Algorithm and hyper-parameters. We use the PPO with a CNN policy implemented in Stable-Baselines 3 (SB3) (Raffin et al., 2021b) as the classical RL baseline. The policy is instantiated as `CnnPolicy` with the default SB3 `NatureCNN` backbone, with hyper-parameters listed in Table 6. Note that the chosen learning rate is selected via grid-search over $[5 \times 10^{-5}, 1 \times 10^{-4}, 1.5 \times 10^{-4}, 2.5 \times 10^{-4}]$. Training is run for up to 2×10^6 samples, i.e., state-

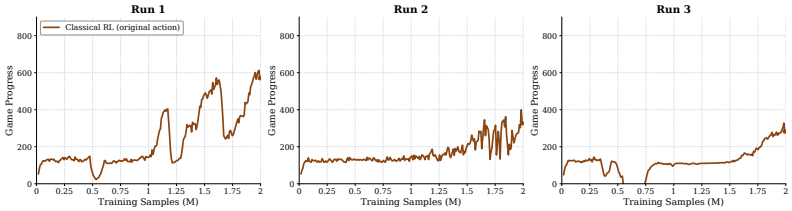
Figure 8: Comparison of VLM-based RL under coarser rewards.



(a) VLM-based RL (i.e., PPO with a turn-level CNN critic and positive advantage filtering).



(b) Classical RL (i.e., PPO training a CNN policy from scratch) using the engineered action space.



(c) Classical RL (i.e., PPO training a CNN policy from scratch) using the original action space.

Figure 9: Comparison between VLM-based RL and classical RL. Individual runs are plotted separately without smoothing.

action pairs, and evaluations are performed every 10^4 samples for 128 trajectories with different random seeds.

F Details of Training and Evaluations of Odysseus

F.1 Details of the CNN Critic

We use the same CNN backbone as (Mnih et al., 2015) for the critic learning and provide the detailed hyper-parameters for critic learning in Table 7.

Action	Buttons
RIGHT	{right}
RIGHT_JUMP	{right, a}
RIGHT_SPRINT_JUMP	{right, a, b}
LEFT	{left}
LEFT_JUMP	{left, a}
RIGHT_SPRINT	{right, b}
LEFT_SPRINT	{left, b}
JUMP	{a}

Table 5: The engineered action space used in the classical RL baselines.

Parameter	Value
batch size	8192
optimization epochs	4
mini-batch size	1024
learning rate	5×10^{-5}
discount factor γ	0.95
GAE parameter λ	0.95
policy clip range ϵ	0.1
value clip range	0.1
maximum gradient norm	0.5
target KL	0.03

Table 6: Hyper-parameters for classical RL experiments with PPO via SB3.

F.2 Other Training Details

During all experiments, the training is performed on all components of the base model (i.e., Qwen3-VL-8B-Instruct), including vision encoder, multi-modal projector, and language model backbone. The SFT initialization is performed using the Qwen3-VL repo[1] with configurations listed in Table 8. RL training is conducted under a substantially modified version of the VeRL framework[2] (Sheng et al., 2024) with configurations listed in Table 9.

F.3 Evaluation Details

We compute mean and standard error over multiple runs of each model. For close-source proprietary models requiring API access, we compute summary statistics over 8 runs. For remaining open-source models, we compute summary statistics over 256 runs. We use the same set of hyperparameters in inference as in training.

For the in-game generalization evaluations, we manually collect game states across 10 levels in *Super Mario Land*. While for cross-game generalization evaluations, the game environment initializes from start in 32 levels of *Super Mario Bros.*

[1] https://github.com/QwenLM/Qwen3-VL
[2] https://github.com/verl-project/verl

Parameter	Value
hidden dimension	512
optimization epochs per training step	1
weight decay	0.0
gradient clip	1.0
learning rate	3×10^{-4}
learning rate scheduler	constant

Table 7: Hyper-parameters for CNN critic learning in Odysseus.

Parameter	Value
dataset size	5058
total epoch	1
batch size	128
gradient accumulation step	2
learning rate	1×10^{-7}
learning rate scheduler	cosine
warmup ratio	0.03

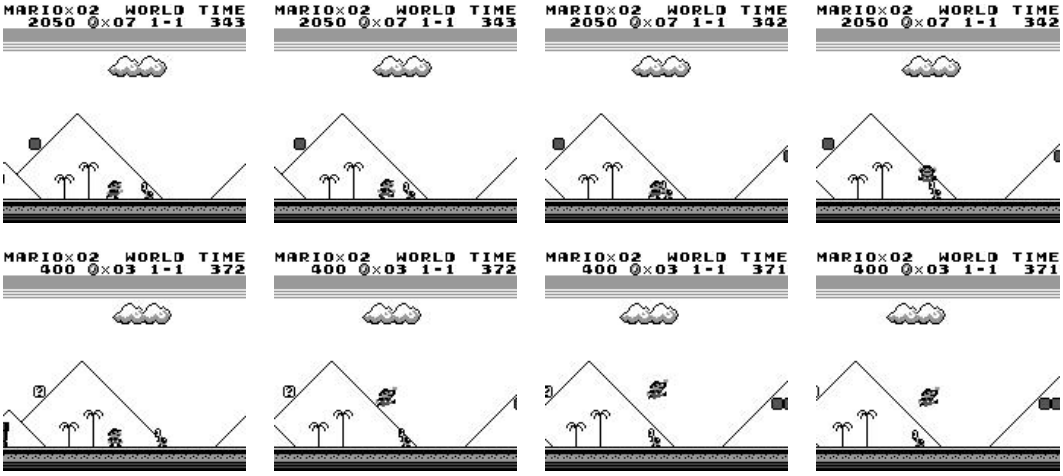
Table 8: Hyper-parameters for SFT training in Odysseus.

Parameter	Value
training steps of Odysseus	190
training steps of Odysseus-Zero	175
max turns	400
top p	1
temperature	1
max response length	4096
number of trajectories per batch	1024
optimization epochs per training step	1
mini-batch size	4096
discount factor γ	0.95
policy clip range $\epsilon_{\text{high}}, \epsilon_{\text{low}}$	$0.28, 0.2$
learning rate	1×10^{-6}
learning rate scheduler	constant

Table 9: Hyper-parameters for RL training in Odysseus.

G Additional Visualizations

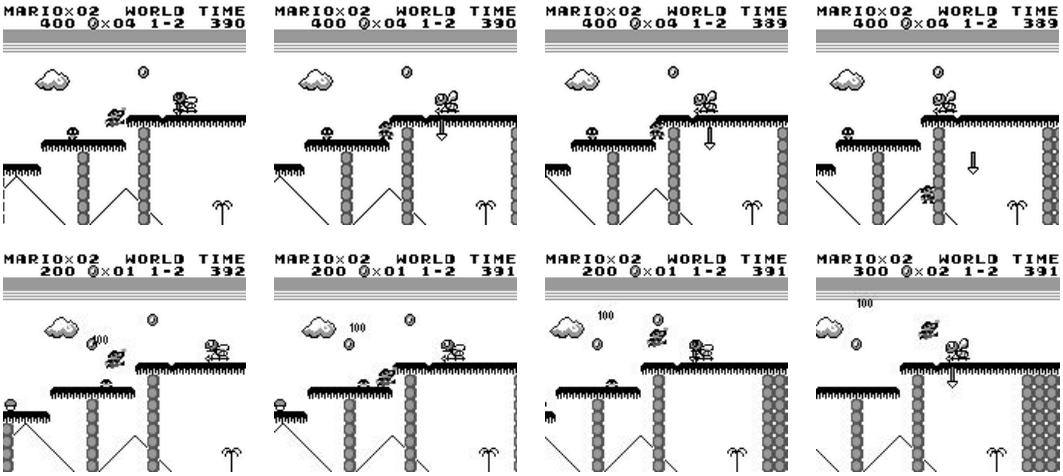
In Fig. 10 and Fig. 11, we provide additional visualizations illustrating representative game scenarios. In each case, the base model fails to make progress, while Odysseus successfully continues. We further highlight key differences in the generated chains of thought (CoTs), which shed light on the improved decision-making behavior of Odysseus.

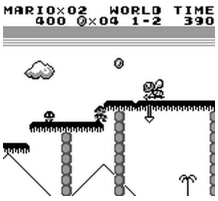


<perception> Mario is standing on a platform at the bottom of the screen in World 1-1. There are two palm trees and a gray block to the left. To the right of Mario, there is a small enemy (likely a Koopa Troopa or similar). There are no coins or power-ups visible. The ground slopes upward to the right, and there is a cloud in the sky. No pipes or ladders are visible. </perception>
<reasoning> 1. Mario is at the start of the level and needs to move forward to progress. 2. The enemy is to the right of Mario and appears to be stationary or moving slowly. Jumping over it or running past it is a safe option. 3. The ground slopes upward, which means running right will take Mario onto a higher platform, potentially avoiding the enemy and progressing. 4. There are no immediate obstacles in front of Mario that require jumping over, and no power-ups or coins to collect right now. 5. Pressing 'right' will move Mario forward while 'b' (hold for running) will allow him to move faster and safely pass the enemy. 6. Since no blocking obstacles are present, it is safe to run without needing to jump. 7. While waiting ('noop') might be an option, it would slow progress. Therefore, moving forward is the best choice. </reasoning>
<answer>['right', 'b']</answer>

<perception>Mario is on the ground in World 1-1, with 2 lives and 3 coins. There is a coin floating above a small hill to the left, and a Goomba enemy is standing on the ground to the right. The screen shows a clear path ahead with no immediate obstacles.</perception>
<reasoning> 1. Mario is currently on the ground and needs to progress forward. The immediate goal is to collect the coin and defeat the Goomba to continue. 2. The coin is floating above a small hill, which means Mario needs to jump to reach it. Pressing 'a' will allow Mario to jump and collect the coin. 3. After collecting the coin, Mario should move right to approach the Goomba. Pressing 'right' will move Mario toward the Goomba. 4. To defeat the Goomba, Mario needs to jump on it or use a fireball if available. Since Mario is on the ground and the Goomba is standing, jumping on it is the most straightforward approach. 5. Therefore, the best action is to jump to collect the coin and then move right to approach the Goomba.</reasoning>
<answer>['a', 'right']</answer>

Figure 10: Example trajectories of base model (top) and Odysseus (bottom). Base model fails in front of the enemy while Odysseus successfully jumps over it. CoT shows that the base model continues to output `right` action, since it fails to accurately recognize the distance between Mario and enemy; however Odysseus is able to jump at the right timing. red and blue highlight the key differences in their generated CoTs.





<perception> Mario is standing on a ledge in World 1-2, with two Goombas on the platforms below and to the left. There is a coin floating above Mario. To the right, there is a pipe with a downward arrow indicating it can be entered. The screen also shows Mario has 2 lives and 4 coins, with 390 time remaining. </perception>
<reasoning> Mario is currently safe on the ledge, with no enemies directly below him. The coin above is within reach and can be collected by jumping. However, the immediate objective is to progress through the level. The pipe on the right is the most direct way to advance, but it requires Mario to crouch or press down to enter it. Since the pipe is located to the right, Mario should move right to reach it. Pressing 'down' will allow him to enter the pipe if he steps on it, as this is standard behavior in Super Mario Land for pipe entry. Therefore, the sequence should be to move right toward the pipe, and then press down to enter it, which will lead to the next part of the level. The coin can be collected later if needed, as progression is the priority. </reasoning>
<answer>['right', 'down']</answer>

<perception> Mario is on a platform in World 1-2, with 2 lives and 1 coin. There are coins floating in the air and a Goomba enemy on the platform to the right. Mario is positioned on a lower platform, facing right, with a gap ahead that requires jumping to cross. There are vertical pillars between the platforms. </perception>
<reasoning> 1. Mario needs to cross the gap to progress. The gap is wide, so Mario must jump to reach the next platform. 2. There is a Goomba enemy on the platform to the right. If Mario jumps without avoiding it, he will take damage. 3. The best strategy is to jump while moving right to land on the next platform, avoiding the Goomba. 4. Since Mario is on a platform, he can jump and move right simultaneously to cross the gap and land on the next platform. 5. This will allow Mario to progress and collect the coins in the air once he lands. </reasoning>
<answer>['a', 'right']</answer>

Figure 11: Example trajectories of base model (top) and Odysseus (bottom). Base model falls into the gap while Odysseus successfully jumps to cross the gap onto the higher platform. CoT shows that base model output `right` and `down` action, since it treats background scene object as the pipe available for entry; however Odysseus is able to realize jumping onto the next platform is necessary to proceed in the game. red and blue highlight the key differences in their generated CoTs.