

CombatVLA: An Efficient Vision-Language-Action Model for Combat Tasks in 3D Action Role-Playing Games

Peng Chen* Pi Bu* Yingyao Wang Xinyi Wang Ziming Wang Jie Guo
 Yingxiu Zhao Qi Zhu Jun Song† Siran Yang Jiamang Wang Bo Zheng
 Alibaba Group

chenpeng.cp0225@gmail.com {bupi.wj, jsong.sj}@alibaba-inc.com

Abstract

Recent advances in Vision-Language-Action models (VLAs) have expanded the capabilities of embodied intelligence. However, significant challenges remain in real-time decision-making in complex 3D environments, which demand second-level responses, high-resolution perception, and tactical reasoning under dynamic conditions. To advance the field, we introduce *CombatVLA*, an efficient VLA model optimized for combat tasks in 3D action role-playing games (ARPGs). Specifically, our *CombatVLA* is a 3B model trained on video-action pairs collected by an action tracker, where the data is formatted as action-of-thought (AoT) sequences. Thereafter, *CombatVLA* seamlessly integrates into an action execution framework, allowing efficient inference through our truncated AoT strategy. Experimental results demonstrate that *CombatVLA* not only outperforms all existing models on the combat understanding benchmark but also achieves a 50-fold acceleration in game combat. Moreover, it has a higher task success rate than human players. We will release all resources, including the action tracker, dataset, model weights, training code, and action execution framework implementation at <https://combatvla.github.io/>.

1. Introduction

Vision-Language-Action Models (VLAs) have achieved groundbreaking progress in embodied intelligence through unified frameworks integrating visual perception, semantic reasoning, and physical action control [1, 18, 19, 46]. In agent applications, they perform well in UI operations and navigation tasks [22, 23, 30, 36, 40] but **struggle with efficient decision-making in complex 3D environments**. A representative challenge lies in combat tasks in 3D action role-playing games (ARPGs) like “Black Myth: Wukong”,

*Equal Contribution.

†Corresponding author.

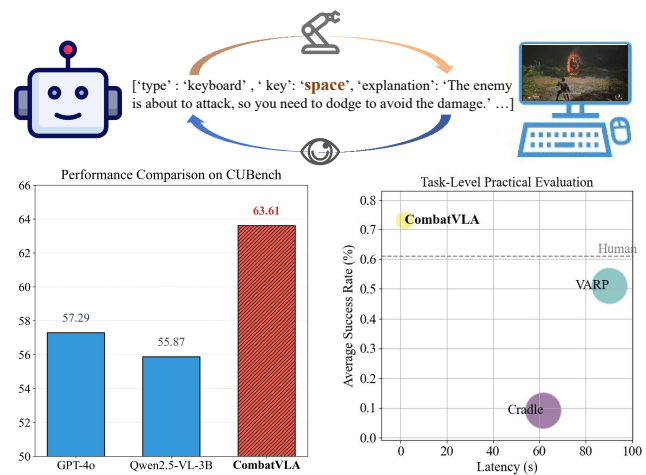


Figure 1. **CombatVLA surpasses GPT-4o and Qwen2.5-VL in combat understanding, is 50 times faster than Cradle and VARP framework, and has a higher success rate than humans.**

which present critical but underexplored demands: real-time processing of high-resolution visual streams, tactical adaptation to dynamically evolving enemy behaviors, and second-level action execution—requirements mirroring latency-sensitive real-world scenarios [3, 5]. In fact, the dynamic complexity of combat games rigorously challenges VLAs’ capabilities in: 1) Visual perception (e.g., enemy and self positioning, movement, and environmental awareness). 2) Combat reasoning (e.g., recognizing enemy attack patterns). 3) Efficient inference (i.e., real-time reaction). Currently, no framework excels in these tasks, nor is there a benchmark for assessing combat understanding.

The pioneers working on 3D combat games primarily access video game APIs to read memory and thereby acquire information on the game environment. For example, Wang et al. [35] used this strategy to employ LLM-driven agents to play Minecraft, enabling automatic mining, exploration, and combat with enemies. However, this method of interacting with the environment is significantly different from that of humans, who rely on vision rather than

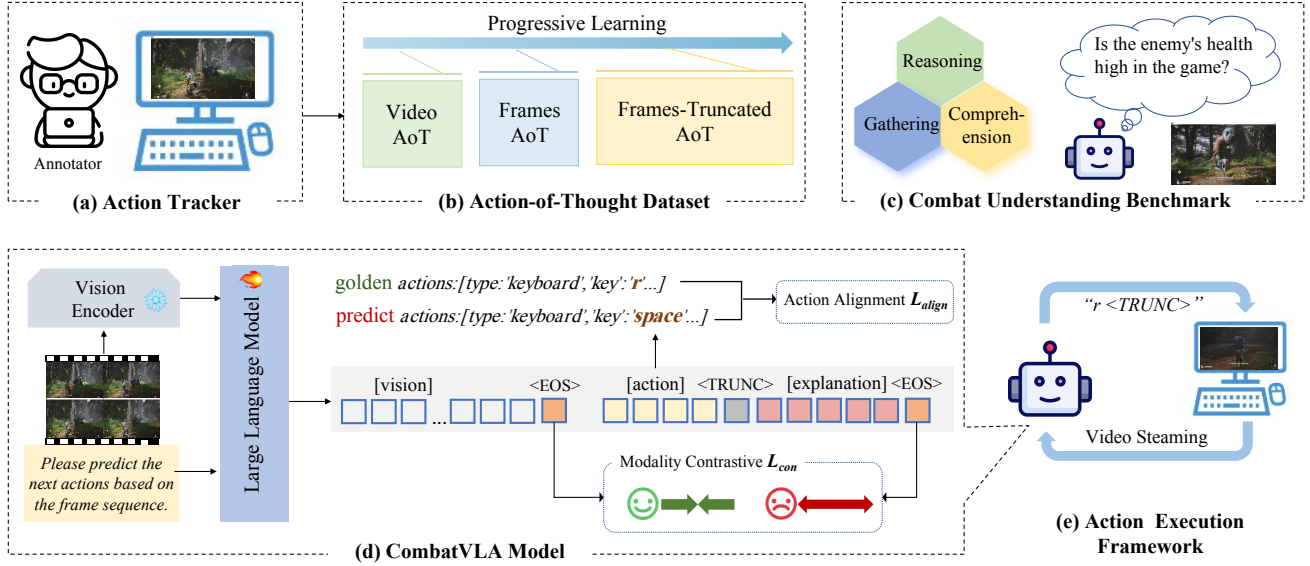


Figure 2. (a) **An action tracker** collects human data on keyboard and mouse use. (b) **Three types of AoT training data** collected by the action tracker are used for progressive learning. (c) **Combat understanding benchmark (namely CUBench)** assesses the model’s combat IQ in three areas: gathering, comprehension, and reasoning. (d) **CombatVLA model** is trained on AoT data with the constraint of action alignment loss and modality contrastive loss. (e) **Deployment of CombatVLA** to operate real PCs.

memory-based reading. Recently, [3] implemented reinforcement learning (RL) to play “Black Myth: Wukong,” using DQN[26] and PPO[31] algorithms with pure visual input, allowing AI to autonomously learn combat scenarios. However, this RL-based approach requires a large number of pre-defined reward designs and extensive trial-and-error training. With the advancement of visual language models (VLMs) [32, 42], works like Cradle [33] and VARP [5] demonstrate significant potential in playing video games. Nonetheless, these efforts heavily depend on ultra-large-scale VLMs like GPT-4o, leading to delays that can exceed 60 or even 90 seconds, as shown in Fig.1. This latency severely hinders the performance in real-time combat games and limits the practical applicability.

In this paper, we propose **CombatVLA**, the first efficient vision-language-action model designed for 3D combat gameplay. For efficient decision making, our CombatVLA is a 3B model that processes visual inputs and outputs a sequence of actions to control the game (including keyboard and mouse operations). Specifically, we first develop an action tracker to collect a substantial amount of training data. The data gathered by this tracker is then structured into an action-of-thought (AoT) format to facilitate action reasoning by the model. Thereafter, CombatVLA is trained using a progressive learning paradigm, enabling the model to learn combat techniques, from video-level AoT tuning to frame-level AoT tuning. Ultimately, CombatVLA can be seamlessly integrated into an action execution agent, enabling efficient inference through our custom truncated output strategy. As shown in Fig.1, the experimental results demonstrate that CombatVLA not only outperformed

all existing models (e.g., GPT-4o and Qwen2.5-VL) in the combat understanding but also achieved a 50-fold increase in execution speed compared to existing VLM-based game agents. The contributions are summarized as follows:

- **AoT Dataset.** We introduce a three-stage AoT dataset consisting of coarse-grained video AoT, fine-grained frames AoT and frames-truncated AoT to enable the model to progressively learn combat skills. To support this dataset collection, we develop an action tracker that operates in the background of the game to record the player’s movements, which will expedite data collection and potentially advance research in the field.
- **Combat Understanding Benchmark.** Based on the action tracker, we establish a benchmark, namely CUBench, for combat understanding that evaluates the models’ performance in identifying enemy positions and action reasoning tasks through a VQA format.
- **CombatVLA Model.** CombatVLA is trained using a progressive learning paradigm, with constraints imposed by adaptive action-weighted losses, and it achieves optimal performance on combat understanding benchmark.
- **Action Execution Framework.** We integrate CombatVLA into an agent framework that operates on PCs, achieving a 50-fold acceleration via truncated strategies.

2. Related Work

2.1. Vision-Language-Action Models

With the development of Vision-Language Models (VLMs), several robust models, such as the Qwen series [38], have demonstrated strong visual capabilities [11]. Subsequently, VLMs were extended to Vision-Language Agents (VLAs)

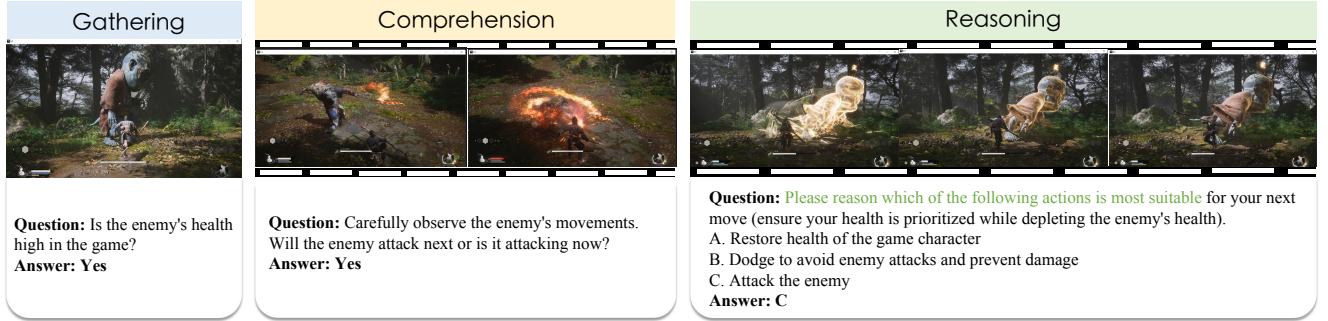


Figure 3. Combat understanding benchmark (i.e. CUBench) has three categories: gathering (single-image judgment), comprehension (multi-image judgment), and reasoning (multiple-image multiple-choice).

to further advance the development of embodied intelligence [12]. VLAs that utilize LLM-based control strategies exhibit strong generalization capabilities [4, 27]. For instance, RT-2 [1] integrates VIT-based vision with PaLM’s linguistic reasoning for robot control, while OpenVLA [18] enhances generalization through large-scale visual-motor pre-training. DeeR-VLA [43] reduces LLM computation through dynamic capacity adjustment, and RoboFlamingo [21] separates VLM strategies into distinct vision-language and action modules, making it suitable for resource-limited platforms. However, current VLA approaches still face challenges in achieving real-time response for latency-sensitive applications, particularly when executing long-horizon planning in complex 3D environments with dynamic visual effects and second-level action windows.

2.2. AI-Driven Game Agents

The development of game agents involves two main approaches: RL- and LLM-based architectures [2, 7, 8, 13, 14, 28, 29, 37, 44, 45]. RL-based methods excel in specific tasks through reward engineering. The project of Black Myth: Wukong AI [3] uses vision-based DQN/PPO algorithms for action RPGs. JARVIS-1 [39] and VPT [17] mimic human interactions using screenshots and keyboard/mouse inputs, but struggle with generalization due to predefined action spaces. LLM-driven agents use language models for reasoning in board games and text adventures. Minecraft agents like Voyager [35] show GPT-4’s ability to generate code. Agents such as PokeLLMon [16] enhance understanding and strategy in Pokémon battles, demonstrating LLMs’ decision-making potential. Cradle [33] offers universal computer control without dedicated APIs, but needs extensive feedback and is less adaptable to new tasks.

3. Tracker and Benchmark

We develop an action tracker to collect human action sequences in games. It provides extensive training data for a combat understanding model. Moreover, we establish a comprehensive combat understanding benchmark, with three tasks using the action tracker.

Action Tracker. Due to the scarcity of training data labeled with actions, we have developed a lightweight python tool for efficiently collecting video-action data pairs, called the action tracker. It can run in the background, monitoring keyboard and mouse activities to record the user’s actions, while simultaneously capturing game screenshots. Since it uses two separate threads, it is necessary to record timestamps to align frames with actions. The frame set $F = \{f_1, f_2, \dots, f_N\}$ consists of frames, where each frame f_i is associated with a timestamp t_{f_i} , and they are ordered such that $t_{f_1} \leq t_{f_2} \leq \dots \leq t_{f_N}$. Similarly, the action set $A = \{a_1, a_2, \dots, a_M\}$ comprises actions, where each action a_j is linked with a timestamp t_{a_j} , and they are arranged in a sequence such that $t_{a_1} \leq t_{a_2} \leq \dots \leq t_{a_M}$. Then the alignment formula is as follows:

$$\forall a_j \in A, a_j \mapsto f_{i_j} \quad (1)$$

where $i_j = \arg \min_i (t_{f_i} \geq t_{a_j})$. It ensures that each action is aligned with the nearest future frame.

Combat Understanding Benchmark. If a VLM-based or VLA-based model is expected to perform well in 3D ARPGs, it needs to have high-dimensional visual perception, and semantic combat logic understanding. Therefore, we establish a combat understanding benchmark (namely, CUBench) centered around three capabilities (i.e. gathering, comprehension, and reasoning), as shown in Fig.3, to assess the model’s combat IQ.

The evaluation data are sourced from the action tracker, which collected recordings and screenshots of human actions in “Black Myth: Wukong,” a 3D action game. The human annotation team comprised six individuals, each of whom had completed all levels of the game. Over the course of two weeks, they labeled the game data, resulting in a collection of 200 hours of recorded actions*.

We then used GPT-4o-0513 to create QA pairs for three tasks: single-image judgment, multi-image judgment, and multiple-image multiple-choice. Notably, all QA pairs were

*The cleaned video data will also be used for the AoT training set without overlap to ensure fair evaluation.

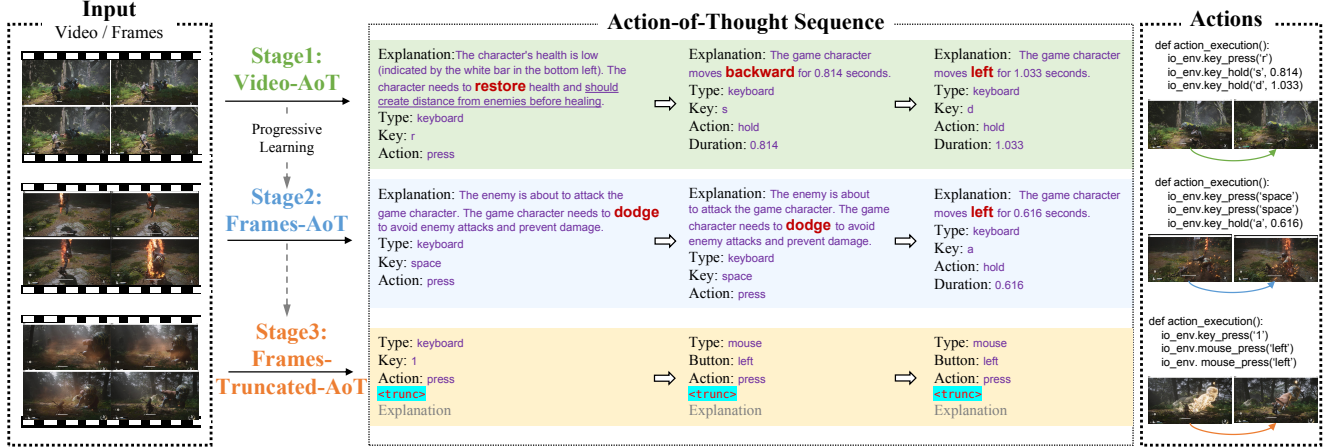


Figure 4. **Visualization of action-of-thought reasoning.** Given a video or frame input, the model can quickly infer the correct action under the semantic guidance of AoT. The special token $\langle \text{TRUNC} \rangle$ speeds up efficient reasoning by truncating output.

annotated by a team of ten people and cross-verified to ensure the quality. Ultimately, we compiled 914 data pieces (39.4% gathering, 22.3% comprehension, 38.3% reasoning) to test the model’s combat understanding. All prompts and data analysis are in the supplementary material.

4. CombatVLA

As illustrated in Fig.2, our CombatVLA is a 3B model designed for efficient inference, capable of processing visual inputs and producing a sequence of actions to control the game (including both keyboard and mouse operations). The training process of CombatVLA follows a three-step progressive learning paradigm, progressing from video-level training to frame-level training. Ultimately, CombatVLA can be seamlessly integrated into the action execution framework, allowing for efficient inference through our truncated AoT strategy.

4.1. Action-of-Thought Construction

Chain-of-Thought (CoT) prompting has been proven extremely effective in enhancing the complex reasoning capabilities of LLMs and MLLMs [20, 41]. Inspired by CoT, we transform the data collected from the action tracker—which includes the set of frames F , the set of actions A , and their alignment—into action-of-thought data (AoT), as illustrated in Fig. 4. Specifically, the model response is formatted in JSON, which includes [action] (such as “press space”) and [explanation] (used to describe the current state of the enemy, the physical meaning of the action, etc.). Additionally, the special token $\langle \text{TRUNC} \rangle$ in “Answer b” represents output truncation to increase inference speed.

Question: $\langle \text{IMG} \rangle \langle \text{IMG} \rangle \langle \text{IMG} \rangle$ Please predict the next actions based on the frame sequence.
Answer a: [explanation] [action] $\langle \text{EOS} \rangle$.
Answer b: [action] $\langle \text{TRUNC} \rangle$ [explanation] $\langle \text{EOS} \rangle$.

4.2. Three-Stage Progressive Learning

CombatVLA’s training follows a three-step progressive learning approach: coarse-grained video-level training (stage1), fine-grained frame-level training (stage2), and truncation strategy training (stage3). The vision encoder’s parameters remain frozen while the language model is fine-tuned. In the first two stages, AoT places [explanation] before [action] for reasoning. In the third stage, [explanation] follows [action] to enable fast truncation and acceleration.

Stage1: Coarse-Grained Video-AoT Tuning. The goal of this training stage is to help the model understand the combat environment, make learning easier, and stabilize training. Regarding the training data, Video-AoT, each video consists of n frames, with a frame rate set to m frames per second. We arrange the actions corresponding to each frame in chronological order, thereby generating video-AoT data pairs. Notably, actions are not precisely timed with frames, so the model must infer actions from the overall visual content rather than exact timing. This strategy enables our model to consider all possible actions and gain an initial understanding of the combat paradigm.

Stage2: Fine-Grained Frames-AoT Tuning. In 3D combat games, precise second-level reaction time is crucial, requiring the model to quickly understand environment and make fast decisions. In this stage, we create action-frame aligned data pairs, called Frames-AoT, tracing back k frames from the current action’s timestamp. For example, if k frames show the enemy preparing to attack, the model might decide to dodge. This strategy helps our model understand the sequence and logic of combat scenarios.

Stage3: Fine-Grained Frames-Truncated-AoT Tuning. The inference time of LLM/VLM-based models is proportional to token length due to the next token prediction requirement. Therefore, we designed a truncation strategy to mitigate the time increase associated with the introduction of AoT. As shown in Fig.4, we reorganized the AoT data by introducing a special token $\langle \text{TRUNC} \rangle$. During real-time

operations, any response following $\langle \text{TRUNC} \rangle$ will be truncated. This strategy allows our model to retain the benefits provided by AoT while accelerating the inference process.

Adaptive Action-Weighted Loss. Our CombatVLA is trained with three losses—language modeling $\mathcal{L}_{\text{lang}}$, action alignment $\mathcal{L}_{\text{align}}$, and modality contrastive $\mathcal{L}_{\text{con}}$ —to address action distribution imbalance.

Firstly, to better capture the correspondence between vision and action, following He et al. [15], we introduce a contrastive loss. Specifically, given an input visual image V and action-of-thought data A , the visual [EOS] and the final [EOS] of the LLM output serve as the local representations of V and A , respectively. Please refer to Fig. 2(d) for a quick understanding. Subsequently, based on whether the actions output by the model A_o **match** the actions in the label A_l , we adjust the distance between the embeddings $\hat{v}_{\text{EOS}}$ and $\hat{a}_{\text{EOS}}$, either bringing them closer or pushing them apart.

A subsequent question is: how do we determine whether A_o and A_l match? We introduce a priority-aware matching criterion based on a predefined action sequence $P = [c_0, \dots, c_{k-1}]$, where c_0 denotes the highest-priority action category (ranked by functional importance and occurrence frequency). The matching function $\mathcal{M}(A_l, A_o)$ evaluates whether the highest-priority action category c^* in A_l (determined by P) exists in A_o :

$$\mathcal{M}(A_l, A_o) = \begin{cases} 1 & \text{if } \arg \max_{c^* \in P} \mathbb{I}(c^* \in A_l) \in A_o \\ 0 & \text{if } \arg \max_{c^* \in P} \mathbb{I}(c^* \in A_l) \notin A_o \end{cases} \quad (2)$$

The loss function dynamically adapts to the matching result: for matched pairs ($\mathcal{M} = 1$), we minimize the cosine distance between $\hat{v}_{\text{EOS}}$ and $\hat{a}_{\text{EOS}}$ via:

$$\mathcal{L}_{\text{con}}^{\text{pull}} = 1 - \cos(\hat{v}_{\text{EOS}}, \hat{a}_{\text{EOS}}) \quad (3)$$

and for mismatched pairs ($\mathcal{M} = 0$), we maximize their separation using $\mathcal{L}_{\text{con}}^{\text{push}} = -\mathcal{L}_{\text{con}}^{\text{pull}}$ while enforcing action prediction accuracy through an alignment loss $\mathcal{L}_{\text{align}} = -\sum \log p(c^*)$. The composite action loss is defined as:

$$\mathcal{L}_{\text{act}} = \begin{cases} \mathcal{L}_{\text{con}}^{\text{pull}} & \text{if } \mathcal{M} = 1 \\ \mathcal{L}_{\text{con}}^{\text{push}} + \mathcal{L}_{\text{align}} & \text{if } \mathcal{M} = 0 \end{cases} \quad (4)$$

The final objective combines language modeling $\mathcal{L}_{\text{lang}}$:

$$\mathcal{L} = \mathcal{L}_{\text{lang}} + \alpha \cdot \mathcal{L}_{\text{act}}, \quad (5)$$

where α is derived from action priorities using exponential weights $\alpha_i = 2^{(k-i-1)}$ normalized to the range $[0.1, 1.0]$. Here, k represents the length of P or the number of action classes, and i is the index of action c^* within P . This formulation prioritizes rare critical actions, ensuring balanced learning and strong vision-action alignment, despite imbalanced action categories.

Table 1. Task definitions in *Black Myth: Wukong* (BMW) and *Sekiro: Shadows Die Twice* (SSDT).

Game	Task ID	Description	Difficulty	Zero-Shot
BMW	1	Defeat WolfScout	Easy	✓
	2	Defeat WolfStalwart	Easy	✓
	3	Defeat WolfSwornsword	Easy	✓
	4	Defeat WolfSoldier	Easy	✓
	5	Defeat Croaky	Easy	✓
	6	Defeat Crow Diviner	Middle	✓
	7	Defeat Bandit Chief	Middle	✓
	8	Defeat Bullguard	Hard	✓
	9	Defeat Wandering Wight	Very Hard	✗
	10	Defeat Guangzhi	Very Hard	✗
SSDT	11	Defeat Katana	Easy	✓
	12	Defeat Hassou Stance	Middle	✓
	13	Defeat Shigenori Yamauchi	Hard	✓

4.3. Action Execution Framework

VLA-based Agent Framework. To support VLMs in playing computer games like humans, we developed a lightweight and fast action execution agent. For instance, our fine-tuned VLM is similar to the human brain, responsible for reasoning and decision-making, while the framework is akin to human eyes and hands, responsible for observation and execution. In real-time PC gameplay, the input of the action execution agent is the real-time game video footage captured, and the output is actions based on mouse and keyboard operations. Specifically, we perform frame sampling on the captured real-time game footage at over 60 FPS, removing redundant visual information to reduce the computational pressure on VLMs during inference. Finally, the model’s output adopts a truncated inference strategy to extract useful action information for execution.

Truncated Inference and Execution. During inference, we monitor each new output token and stop when we see the $\langle \text{TRUNC} \rangle$ token, converting prior tokens into actions. This strategy accelerates the inference speed. Subsequently, we translate the actions into Python code using the “pyautogui” library to automate mouse and keyboard controls, enabling the game character to execute combat tasks.

5. Experiments

5.1. Implementation Details

Dataset. Following VARP [5], we employ the two games, “Black Myth: Wukong(BMW)” and “Sekiro: Shadows Die Twice(SSDT)”, as experimental platforms. The annotators defined 13 combat tasks based on their difficulty levels, categorizing them into four distinct levels: easy, medium, hard, and very hard, as shown in Tab. 1.

We collected training data from task 9 and 10 of “Black Myth: Wukong” using our action tracker. After manual selection, we organized approximately 25k game screenshots and 5k high-quality AoTs, split into 95% for training and 5% for validation. The AoTs includes 10 actions: “wsad” for movement, “shift” for quick moves, “space” for dodging, “r” for healing, “1” for immobilization, “left mouse

Table 2. Performance comparison of closed source and open source LVLMs on the combat understanding benchmark and general benchmark. The highest scores among models in each metric are highlighted in **FirstBest**.

Model	Combat Understanding				General Benchmark		
	Gathering	Comprehension	Reasoning	Avg.	MME	VideoMME	OCRBench
<i>Closed-Source Large Vision Language Models</i>							
GPT-4o-0513	58.06	66.67	47.14	57.29	2328	71.9	736
GPT-4o-mini-0718	59.44	66.18	42.57	56.06	2003	64.8	785
GPT-4-vision-preview	52.78	53.92	43.71	50.14	1926	59.9	645
Gemini-2.0-flash	58.61	64.22	50.86	57.90	—	—	—
Gemini-1.5-pro	64.44	62.75	41.71	56.30	2110	75.0	754
Claude3.5-sonnet	53.89	57.35	55.43	55.56	1920	60.0	788
<i>Open-Source Large Vision Language Models</i>							
LLaVA-1.5-7B	50.56	60.29	42.86	51.24	1510	—	—
InternVL2.5-4B	53.89	48.04	43.71	48.55	2337	62.3	828
Qwen2-VL-7B	55.28	59.80	43.14	52.74	2326	63.3	866
Qwen2-VL-2B	53.33	46.57	42.86	47.59	1872	55.6	809
Qwen2.5-VL-7B	45.56	52.94	50.57	49.69	2347	65.1	864
Qwen2.5-VL-3B	53.61	56.86	57.14	55.87	2157	61.5	797
CombatVLA-3B (Ours)	60.83	60.29	69.71	63.61	2141	58.7	741

button” for light attacks, and hold “right mouse button” for heavy attacks. Notably, these actions can be combined.

Benchmarks. We evaluated baselines using the combat understanding benchmark (CUBench), general benchmarks (i.e., MME [9], VideoMME [10], OCRBench [25]), and practical tests. In task-level practical tests, the action execution framework controls the PC to engage in real combat. All baselines are tested 10 times per task, with success marked by defeating the enemy and failure by being defeated. We calculate success rates from these attempts and record average inference time. Notably, our CombatVLA, fine-tuned only on very hard tasks (tasks 9 and 10), uses tasks 1-8 (same game, different tasks) and 11-13 (different game, different tasks) as zero-shot tests for assessing generalization. For fair comparison, we made adaptation improvements to Cradle framework [33] for the two games.

Baselines. We selected closed-source models such as GPT-4o, GPT-4o-mini, GPT-4-vision-preview[†], Gemini-2.0-flash[‡], Gemini-1.5-pro [34], and Claude3.5-Sonnet[§], as well as open-source models of similar sizes like LLaVA-1.5-7B [24], InternVL2.5-4B [6], Qwen2-VL-7B/2B [38] and Qwen2.5-VL-7B/3B[¶], as baselines for the CUBench and general benchmarks. Cradle [33], VARP [5] and ten human players were chosen as baselines for task-level prac-

Table 3. Comparison of latency and model calls per inference.

Method	Latency(s)↓	Model Calls↓
Cradle [33]	61.68	5
VARP [5]	90.23	10
CombatVLA (Ours)	1.85	1

tical tests.

Training Settings. Our backbone leverages Qwen2.5-VL-3B with a full-parameter supervised fine-tuning(SFT) on our AoT dataset. We have configured a learning rate of 1e-5, with a batch size of 1, and set the temperature to 0.7. During the coarse-grained video-AoT tuning phase, we use $n = 20$ and $m = 10$, training for 3 epochs. In the subsequent fine-grained frames-AoT tuning phase, we set $k = 4$ and train for 1 epoch. The finally stage3 phase is carried out over 3 epochs. All training and benchmark evaluations were conducted on 4 NVIDIA H20 GPUs. Task-level practical tests were conducted on a NVIDIA RTX 4090 GPU.

5.2. Main Results

Combat Understanding Evaluation. We evaluated CombatVLA and the baselines on CUBench and general benchmarks, with experimental results shown in Tab.2. On CUBench, our CombatVLA achieved the highest average score of 63.61, surpassing the second-highest Gemini-2.0-flash by 5.71 points. Compared to the original backbone, Qwen2.5-VL-3B, it improved by 7.74 points. This indicates that our method significantly enhances the model’s capability in combat understanding. Specifically, while

[†]<https://openai.com/index/>

[‡]<https://deepmind.google/technologies/gemini/flash/>

[§]<https://www.anthropic.com/news/claude-3-5-sonnet>

[¶]<https://help.aliyun.com/zh/model-studio/developer-reference/use-qwen-by-calling-api>

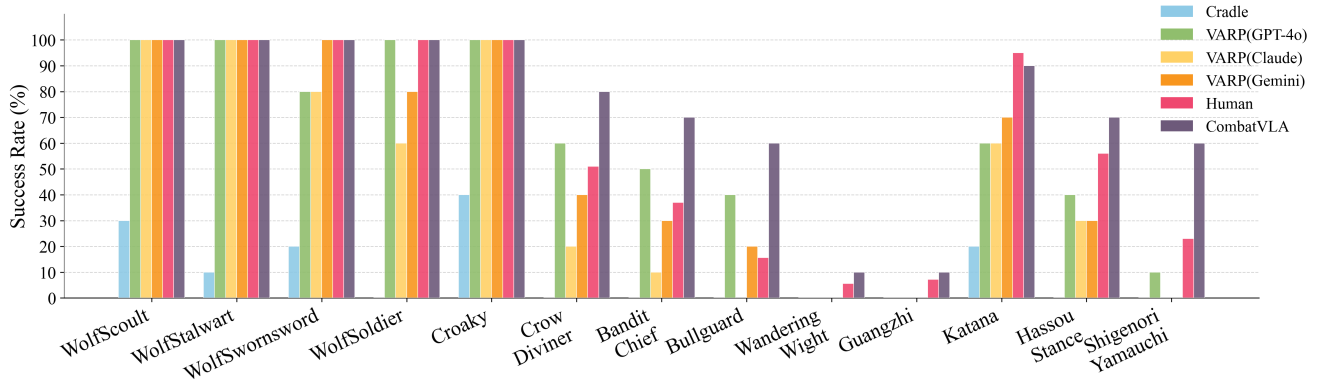


Figure 5. Comparison of task-level practical tests. Our CombatVLA not only outperforms all VLM-based agents (i.e., Cradle and VARP) but also has a higher task success rate than human players.

ours did not perform best in low-level abilities such as information gathering and combat comprehension, it still demonstrated considerable competitiveness. However, in high-level action reasoning, ours outperformed the second-highest Claude35-sonnet by 14.28 points, thanks to the action-of-thought data enhancing the model’s reasoning capability.

General Benchmark Evaluation. MME, VideoMME, and OCRBench are representative benchmarks for image, video, and rich text, respectively. They are used to evaluate the impact of AoT training on general capabilities. As shown in Tab.2, despite being trained on specific tasks, our CombatVLA maintains comparable performance to the backbone model Qwen2.5-VL-3B. This further confirms the robustness of our approach.

Task-Level Practical Evaluation. We integrated CombatVLA into the action execution agent to play the game like a human, automatically carrying out combat tasks. Due to unavoidable inference delays with the model, the agent pauses the game during model inference and resumes the game when executing actions. To ensure the fairness and feasibility of the experiment, we set all methods to have the game character’s attack attribute at 100 and defense attribute at 600 when testing task 9 and task 10. The key settings for SSDT and BMW are consistent, such as the block action in SSDT and the dodge action in BMW, both of which are assigned to the “space” key.

The experimental results are shown in Fig. 5. The following observations can be made: 1) Although we made adaptation improvements for Cradle, its reasoning heavily relies on explicit text prompts within the game screenshot. Since combat tasks require the ability to extract implicit visual information, Cradle performed the worst across all tasks. 2) VARP, despite extensive engineering adaptations for the BMW game, performed poorly on BMW’s hard and very hard difficulty tasks. Moreover, VARP’s success rate in the SSDT game significantly decreased, indicating low generalization capability. 3) Our CombatVLA, apart from being comparable to humans on some easy tasks, surpassed baselines on the other tasks, especially the hard and very

Table 4. Ablation study of progressive learning.

Training	Gathering	Comprehension	Reasoning	Avg.	Time(s)↓
Stage1	53.89	57.35	60.57	57.27	3.73
Stage2	59.17	62.25	62.86	61.43	3.73
Stage3	60.83	60.29	69.71	63.61	1.85

Table 5. Ablation study of adaptive action-weighted loss.

Loss Setting	Gathering	Comprehension	Reasoning	Avg.
Stage3	60.83	60.29	69.71	63.61
w/o $\mathcal{L}_{con}$	62.78	58.82	63.14	61.58
w/o $\mathcal{L}_{align}$	61.39	59.80	63.71	61.64

hard tasks. Zero-shot tests on tasks 1 to 8 of the same game, and tasks 11 to 13 of a different game, also demonstrated CombatVLA’s strong generalization ability.

Inference Latency. We also compared the average latency per inference (i.e., the process of inputting visual information and outputting actions) and the number of model calls with the baselines, as shown in Tab. 3. Our model requires only 1.8 seconds of delay and one model call, compared to VARP, representing a speed increase of 50 times and reducing model call costs to $\frac{1}{10}$ of VARP’s. For more detailed information and visualizations, please refer to our supplementary materials and demo videos.

5.3. Ablation Studies

Ablation of Progressive Learning. We evaluated the different stages of progressive learning, as well as the experimental results of $\mathcal{L}_{con}$ and $\mathcal{L}_{align}$ on CUBench. As shown in Tab. 4, *Stage3* represents our full CombatVLA model. The following observations can be made: 1) *Stage1* can only learn coarse-grained actions from the video, but the actions are not aligned with specific frames in the video, so its performance is the worst. 2) Additionally, Tab. 4 reports the average inference time for invoking the model once, and due to the truncation mechanism, the speed of *Stage3* is about 2 times that of *Stage2*.

Ablation of Adaptive Loss. As progressive learning proceeds, the model’s performance in gathering and reason-



Figure 6. Visualizations of some representative cases involving BMW and SSDT are provided, along with explanations of the model outputs, the corresponding action execution codes, and the frame-by-frame sequences.

ing gradually improves. Due to the introduction of the $\langle \text{TRUNC} \rangle$ token in *Stage3*, which places the action’s explanation after the action and truncates it, the model cannot access the semantic information when generating actions, thus somewhat impairing the model’s understanding performance. However, as shown in Tab. 5, the introduction of $\mathcal{L}_{\text{con}}$ and $\mathcal{L}_{\text{align}}$ enhanced the model’s reasoning performance, reaching 69.71, which is 6.85 points higher than *Stage2*, with an average score increase of 2.18 points.

5.4. Qualitative Visualization

We demonstrated some representative cases from the task-level practical tests. We reported the AoT explanations inferred by CombatVLA, the actions parsed into Python code, and the sequence of frames after executing the actions, as shown in Fig. 6. The first three rows are belong to BMW, and the fourth is SSDT. We have the following observations:

- In the first row, CombatVLA detected its own low health and decided to immediately use the restore health action. Therefore, it first moved the game character backward to a safe position and then pressed the “r” key to increase its health.
- In the second row, CombatVLA detected that its immobilizing skill was in an available state, so it pressed the “l” key to immobilize the enemy, then immediately launched a series of continuous attacks, depleting the enemy’s health significantly.

- The third row shows how our model effectively dodged the enemy’s attack and then used a heavy attack with a wind-up at an opportune moment.
- In the fourth row, CombatVLA first used the block action to withstand an enemy attack, then executed a light attack to perform a shinobi deathblow, killing the enemy in one hit.

Overall, CombatVLA demonstrates a strong ability to understand combat tasks and can effectively reason out for various complex situations with the help of advanced semantic information from AoT explanations.

6. Conclusion

In this paper, we aim to address the issue that VLMs or VLAs lack second-level response times, high-resolution perception, and tactical reasoning in 3D action role-playing games. Specifically, we introduce CombatVLA, a 3B model trained on AoT sequences with the constraint of action alignment loss and modality contrastive loss. Thereafter, CombatVLA seamlessly integrates into an action execution framework, allowing efficient inference through our truncated AoT strategy. Experimental results demonstrate that our CombatVLA not only surpasses all existing models in combat understanding benchmarks, while maintaining generalization capability, but also achieves a 50-fold speedup in real-time combat scenarios. In the future, we will further enhance the model’s understanding of gaming scenarios, thereby expanding its application to more games.

Acknowledgments. This work was supported by Alibaba Group through Alibaba Research Intern Program.

References

- [1] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishk Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspier Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-2: Vision-language-action models transfer web knowledge to robotic control, 2023. 1, 3
- [2] Leng Cai, Junxuan He, Yikai Li, Junjie Liang, Yuanping Lin, Ziming Quan, Yawen Zeng, and Jin Xu. Rtbagent: A llm-based agent system for real-time bidding, 2025. 3
- [3] Turing’s Cat. Ai-wukong: RL-based arpg gamebot. <https://github.com/Turing-Project/RL-ARPG-Agent>, 2024. 1, 2, 3
- [4] Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, Hanbo Zhang, and Minzhao Zhu. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation, 2024. 3
- [5] Peng Chen, Pi Bu, Jun Song, Yuan Gao, and Bo Zheng. Can vlms play action role-playing games? take black myth wukong as a study case, 2024. 1, 2, 5, 6
- [6] Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, Lixin Gu, Xuehui Wang, Qingyun Li, Yimin Ren, Zixuan Chen, Jiapeng Luo, Jiahao Wang, Tan Jiang, Bo Wang, Conghui He, Botian Shi, Xingcheng Zhang, Han Lv, Yi Wang, Wenqi Shao, Pei Chu, Zhongying Tu, Tong He, Zhiyong Wu, Huipeng Deng, Jiaye Ge, Kai Chen, Kaipeng Zhang, Limin Wang, Min Dou, Lewei Lu, Xizhou Zhu, Tong Lu, Dahua Lin, Yu Qiao, Jifeng Dai, and Wenhao Wang. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling, 2025. 6
- [7] Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web, 2023. 3
- [8] Benjamin Ellis, Jonathan Cook, Skander Moalla, Mikayel Samvelyan, Mingfei Sun, Anuj Mahajan, Jakob Foerster, and Shimon Whiteson. Smacv2: An improved benchmark for cooperative multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024. 3
- [9] Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, Yunsheng Wu, and Rongrong Ji. Mme: A comprehensive evaluation benchmark for multimodal large language models, 2024. 6
- [10] Chaoyou Fu, Yuhao Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, Peixian Chen, Yanwei Li, Shaohui Lin, Sirui Zhao, Ke Li, Tong Xu, Xiawu Zheng, Enhong Chen, Rongrong Ji, and Xing Sun. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis, 2024. 6
- [11] Jihao Gu, Yingyao Wang, Pi Bu, Chen Wang, Ziming Wang, Tengtao Song, Donglai Wei, Jiale Yuan, Yingxiu Zhao, Yancheng He, Shilong Li, Jiaheng Liu, Meng Cao, Jun Song, Yingshui Tan, Xiang Li, Wenbo Su, Zhicheng Zheng, Xiaoyong Zhu, and Bo Zheng. “see the world, discover knowledge”: A chinese factuality evaluation for large vision language models, 2025. 2
- [12] Yanjiang Guo, Jianke Zhang, Xiaoyu Chen, Xiang Ji, Yen-Jen Wang, Yucheng Hu, and Jianyu Chen. Improving vision-language-action model with online reinforcement learning, 2025. 3
- [13] Izzeddin Gur, Hiroki Furuta, Austin Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. A real-world webagent with planning, long context understanding, and program synthesis, 2024. 3
- [14] Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. Webvoyager: Building an end-to-end web agent with large multimodal models, 2024. 3
- [15] Hulingxiao He, Geng Li, Zijun Geng, Jinglin Xu, and Yuxin Peng. Analyzing and boosting the power of fine-grained visual recognition for multi-modal large language models, 2025. 5
- [16] Sihao Hu, Tiansheng Huang, and Ling Liu. Pokellmon: A human-parity agent for pokemon battles with large language models, 2024. 3
- [17] Karolis Jucys, George Adamopoulos, Mehrab Hamidi, Stephanie Milani, Mohammad Reza Samsami, Artem Zhulov, Sonia Joseph, Blake Richards, Irina Rish, and Özgür Şimşek. Interpretability in action: Exploratory analysis of vpt, a minecraft agent, 2024. 3
- [18] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model, 2024. 1, 3
- [19] Chengmeng Li, Junjie Wen, Yan Peng, Yaxin Peng, Feifei Feng, and Yichen Zhu. Pointvla: Injecting the 3d world into vision-language-action models. *arXiv preprint arXiv:2503.07511*, 2025. 1
- [20] Chengzu Li, Wenshan Wu, Huanyu Zhang, Yan Xia, Shaoguang Mao, Li Dong, Ivan Vulić, and Furu Wei. Imagine while reasoning in space: Multimodal visualization-of-thought, 2025. 4
- [21] Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, Hang Li, and Tao Kong. Vision-language foundation models as effective robot imitators, 2024. 3

- [22] Xiangyu Li, Yawen Zeng, Xiaofen Xing, Jin Xu, and Xiangmin Xu. Hedgeagents: A balanced-aware multi-agent financial trading system, 2025. 1
- [23] Kevin Qinghong Lin, Linjie Li, Difei Gao, Zhengyuan Yang, Shiwei Wu, Zechen Bai, Weixian Lei, Lijuan Wang, and Mike Zheng Shou. Showvui: One vision-language-action model for gui visual agent, 2024. 1
- [24] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning, 2024. 6
- [25] Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xu-Cheng Yin, Cheng-Lin Liu, Lianwen Jin, and Xiang Bai. Ocrbench: on the hidden mystery of ocr in large multimodal models. *Science China Information Sciences*, 67(12), 2024. 6
- [26] 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. 2
- [27] Keyu Pan and Yawen Zeng. Do llms possess a personality? making the mbti test an amazing evaluation for large language models, 2023. 3
- [28] Joon Sung Park, Joseph O’Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22, 2023. 3
- [29] Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, et al. Chatdev: Communicative agents for software development. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15174–15186, 2024. 3
- [30] Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, et al. Ui-tars: Pioneering automated gui interaction with native agents. *arXiv preprint arXiv:2501.12326*, 2025. 1
- [31] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 2
- [32] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023. 2
- [33] Weihao Tan, Wentao Zhang, Xinrun Xu, Haochong Xia, Ziluo Ding, Boyu Li, Bohan Zhou, Junpeng Yue, Jiechuan Jiang, Yewen Li, Ruyi An, Molei Qin, Chuqiao Zong, Longtao Zheng, Yujie Wu, Xiaoqiang Chai, Yifei Bi, Tianbao Xie, Pengjie Gu, Xiyun Li, Ceyao Zhang, Long Tian, Chaojie Wang, Xinrun Wang, Börje F Karlsson, Bo An, Shuicheng Yan, and Zongqing Lu. Cradle: Empowering foundation agents towards general computer control, 2024. 2, 3, 6
- [34] Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024. 6
- [35] 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, 2023. 1, 3
- [36] Junyang Wang, Haiyang Xu, Haitao Jia, Xi Zhang, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. Mobile-agent-v2: Mobile device operation assistant with effective navigation via multi-agent collaboration. *Advances in Neural Information Processing Systems*, 37:2686–2710, 2024. 1
- [37] Junyang Wang, Haiyang Xu, Jiabo Ye, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. Mobile-agent: Autonomous multi-modal mobile device agent with visual perception, 2024. 3
- [38] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model’s perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024. 2, 6
- [39] Zihao Wang, Shaofei Cai, Anji Liu, Yonggang Jin, Jinbing Hou, Bowei Zhang, Haowei Lin, Zhaofeng He, Zilong Zheng, Yaodong Yang, Xiaojuan Ma, and Yitao Liang. Jarvis-1: Open-world multi-task agents with memory-augmented multimodal language models, 2023. 3
- [40] Ziyue Wang, Yurui Dong, Fuwen Luo, Minyuan Ruan, Zhili Cheng, Chi Chen, Peng Li, and Yang Liu. Escapecraft: A 3d room escape environment for benchmarking complex multimodal reasoning ability. *arXiv preprint arXiv:2503.10042*, 2025. 1
- [41] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023. 4
- [42] Hui Yang, Sifu Yue, and Yunzhong He. Auto-gpt for online decision making: Benchmarks and additional opinions. *arXiv preprint arXiv:2306.02224*, 2023. 2
- [43] Yang Yue, Yulin Wang, Bingyi Kang, Yizeng Han, Shenzhi Wang, Shiji Song, Jiashi Feng, and Gao Huang. Deer-vla: Dynamic inference of multimodal large language models for efficient robot execution, 2024. 3
- [44] Yuexiang Zhai, Hao Bai, Zipeng Lin, Jiayi Pan, Shengbang Tong, Yifei Zhou, Alane Suhr, Saining Xie, Yann LeCun, Yi Ma, and Sergey Levine. Fine-tuning large vision-language models as decision-making agents via reinforcement learning, 2024. 3
- [45] Chi Zhang, Zhao Yang, Jiaxuan Liu, Yucheng Han, Xin Chen, Zebiao Huang, Bin Fu, and Gang Yu. Appagent: Multimodal agents as smartphone users, 2023. 3
- [46] Wenqi Zhang, Mengna Wang, Gangao Liu, Xu Huixin, Yiwei Jiang, Yongliang Shen, Guiyang Hou, Zhe Zheng, Hang Zhang, Xin Li, et al. Embodied-reasoner: Synergizing visual search, reasoning, and action for embodied interactive tasks. *arXiv preprint arXiv:2503.21696*, 2025. 1