

NaviMaster: Learning a Unified Policy for GUI and Embodied Navigation Tasks

Anonymous ACL submission

Abstract

Recent advances in Graphical User Interface (GUI) and embodied navigation have driven progress, yet these domains have largely evolved in isolation, with disparate datasets and training paradigms. In this paper, we observe that both tasks can be formulated as Markov Decision Processes (MDP), suggesting a foundational principle for their unification. Hence, we present NaviMaster, the first unified agent capable of unifying GUI navigation and embodied navigation within a single framework. Specifically, NaviMaster (i) proposes a visual-target trajectory collection pipeline that generates trajectories for both GUI and embodied tasks using a single formulation. (ii) employs a unified reinforcement learning framework on the mix data to improve generalization. (iii) designs a novel distance-aware reward to ensure efficient learning from the trajectories. Through extensive experiments on out-of-domain benchmarks, NaviMaster is shown to outperform state-of-the-art agents in GUI navigation, spatial affordance prediction, and embodied navigation. Ablation studies further demonstrate the efficacy of our unified training strategy, data mixing strategy, and reward design. Resources will be released to the community.

1 Introduction

Graphical user interface (GUI) navigation agents and embodied navigation agents are designed to traverse virtual and physical environments, respectively. Recent advances in multimodal large language models (MLLMs) (Bai et al., 2025) have enabled the integration of their strong perception and planning abilities for both agents (Wu et al., 2025; Lin et al., 2025). Leveraging these capabilities, such agents have shown substantial potential in instruction-guided multimodal navigation tasks.

Despite the progress of previous agents, as illustrated in Fig. 1, the long-term separation between GUI and embodied navigation, coupled with their

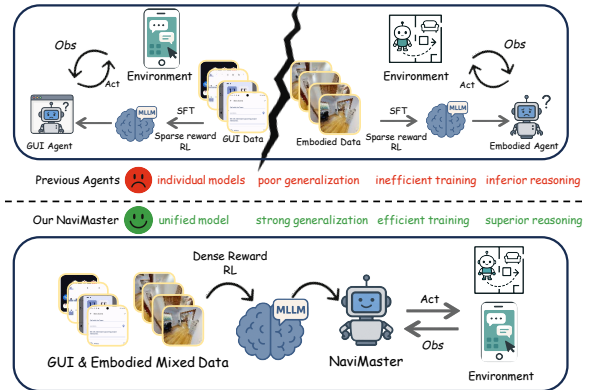


Figure 1: Previous methods involve individual models for GUI and embodied navigation. Our NaviMaster is a unified learning framework.

training strategies, has resulted in four persistent challenges. (1) These approaches employ two individual models for navigation, which increases training and deployment costs and precludes synergistic interaction between the two tasks (Hong et al., 2025). (2) Although prior works (Rawles et al., 2023b; Ramakrishnan et al., 2021) have improved performance in respective tasks by scaling data within specific task data, they exhibit limited cross-task performance due to poor generalization to out-of-domain (OOD) data. (3) They face a training-efficiency bottleneck: previous RFT-based models employ a sparse reward signal, rendering reinforcement learning optimization inefficient. (4) Current RFT reasoning models often generate correct thoughts but wrong actions, as their “understanding” is primarily distilled from texts rather than visual observations.

To address these challenges, we propose a unified policy that integrates GUI and embodied navigation with an efficient training strategy. Inspired by VIS-Bench (Yang et al., 2025a), humans convert egocentric perceptions into an allocentric mental map to support perspective-taking and spatial reasoning. Similarly, both GUI and embodied navigation operate purely on egocentric visual ob-

069 servations, without direct access to a global state. 119
 070 Despite differences in surface-level action semantics, 120
 071 the two tasks are therefore isomorphic at the 121
 072 level of perception and decision-making: in both 122
 073 cases, the agent must integrate partial, first-person 123
 074 observations over time to implicitly construct a 124
 075 latent representation. We therefore characterize 125
 076 the unified problem as learning an egocentric-to- 126
 077 allocentric cognitive transformation. On the other 127
 078 hand, from the perspective of Markov Decision 128
 079 Processes (MDPs): 129

$$080 \quad \arg \max_{a \in \mathcal{A}} P(\mathcal{S}_{t+1} | \mathcal{S}_t = \sigma, \mathcal{A}_t = a), \quad (1)$$

081 where next state \mathcal{S}_{t+1} depends solely on the cur- 130
 082 rent state-action pair (σ, a) . Under our unified 131
 083 formulation, the state \mathcal{S}_t corresponds to the egocen- 132
 084 tric visual observation at step t , while the action 133
 085 space \mathcal{A} encompasses interactions in either virtual 134
 086 interfaces or physical environments. The transi- 135
 087 tion dynamics satisfy the Markov property in both 136
 088 domains. Consequently, we formalize GUI and 137
 089 embodied navigation as a single *Navigation Agent* 138
 090 problem. 139

091 To this end, as shown in Fig. 1, NaviMaster 140
 092 incorporates three key advancements (1) We pro- 141
 093 pose a trajectory collection pipeline that unifies 142
 094 both GUI and embodied navigation within a visual- 143
 095 target paradigm, enabling joint training on mix 144
 096 data and improving generalization. (2) We build 145
 097 a unified reinforcement learning training frame- 146
 098 work applicable to both navigation types. Policies 147
 099 optimized on a single MDP tend to overfit to task- 148
 100 specific correlations. By unifying GUI and em- 149
 101 bodied navigation under a distribution over MDPs, 150
 102 we enable the policy to learn generalizable struc- 151
 103 tural representations such as visual object perma- 152
 104 nence, relative spatial reasoning, and affordance 153
 105 grounding. Specifically, we extend the training 154
 106 strategy to estimate task-specific advantages, en- 155
 107 abling a single policy to adapt effectively across 156
 108 multiple tasks. Furthermore, the framework lever- 157
 109 ages prior reasoning steps and actions as historical 158
 110 context. Given the history and current observa- 159
 111 tions as inputs, the model predicts the next action. 160
 112 This formulation unifies the I/O representation and 161
 113 enables the history to guide precise high-level ac- 162
 114 tions in long-horizon navigation. (3) We employ a 163
 115 distance-aware dense reward in the reinforcement 164
 116 learning framework, which enhances training effi- 165
 117 ciency compared to a sparse binary reward. 166
 118 We evaluate NaviMaster on OOD GUI and 167

embodied navigation benchmarks. On test sets 119
 that differ significantly from the training domain, 120
 NaviMaster outperforms state-of-the-art baselines, 121
 achieving superior results across multiple datasets. 122
 These findings demonstrate strong generalization 123
 capability and robustness to distributional shifts. In 124
 summary, our key contributions are as follows: 125

1. We propose NaviMaster, the first unified navi- 126
 gation agent that jointly handles both GUI and 127
 embodied navigation within one framework. 128
2. We develop a visual-target trajectory collec- 129
 tion pipeline that aggregates high-quality tra- 130
 jectories from both GUI and embodied navi- 131
 gation, thereby increasing data diversity and 132
 enhancing model generalization capability. 133
3. We design a distance-aware dense reward 134
 and a unified reinforcement learning pipeline, 135
 which together enhance data efficiency and 136
 further strengthen model grounding ability. 137

2 Related Work 138

2.1 Navigation Agent 139

GUI navigation agents aim to autonomously oper- 140
 ate applications by perceiving UI elements and issu- 141
 ing precise point-level actions (Wang et al., 2025). 142
 Recent efforts have adopted the data-driven train- 143
 ing paradigm such as OS-Atlas, UI-Tars (Wu et al., 144
 2025; Qin et al., 2025). They employ large-scale 145
 datasets in a multi-stage training pipeline to further 146
 enhance their UI grounding precision and planning 147
 capability. Despite this progress, most existing 148
 GUI agents rely heavily on supervised fine-tuning 149
 (SFT) with large amounts of human-annotated data, 150
 limiting their generalization capacities. To address 151
 this, models like UI-R1 and GUI-R1 (Luo et al., 152
 2025; Lu et al., 2025) incorporate reinforcement 153
 learning (RL) inspired by DeepSeek-R1. However, 154
 their scope remains restricted to GUI-only settings 155
 and lack the capacity to do embodied navigation. 156

Embodied navigation agents control physical or 157
 simulated agents to follow language instructions 158
 in 3D spaces, requiring multimodal perception and 159
 long-horizon planning (Gao et al., 2024). Analo- 160
 gous to GUI navigation tasks, works on embodied 161
 navigation typically employ a multi-stage SFT strat- 162
 egy on large datasets to adapt open-source MLLMs 163
 for navigation tasks (e.g., RoboPoint (Yuan et al., 164
 2024), SpaceLLaVa (Foutter et al., 2025)). Their 165
 scope remains restricted to a single domain, which 166
 enforces a monolithic action space, thereby lim- 167

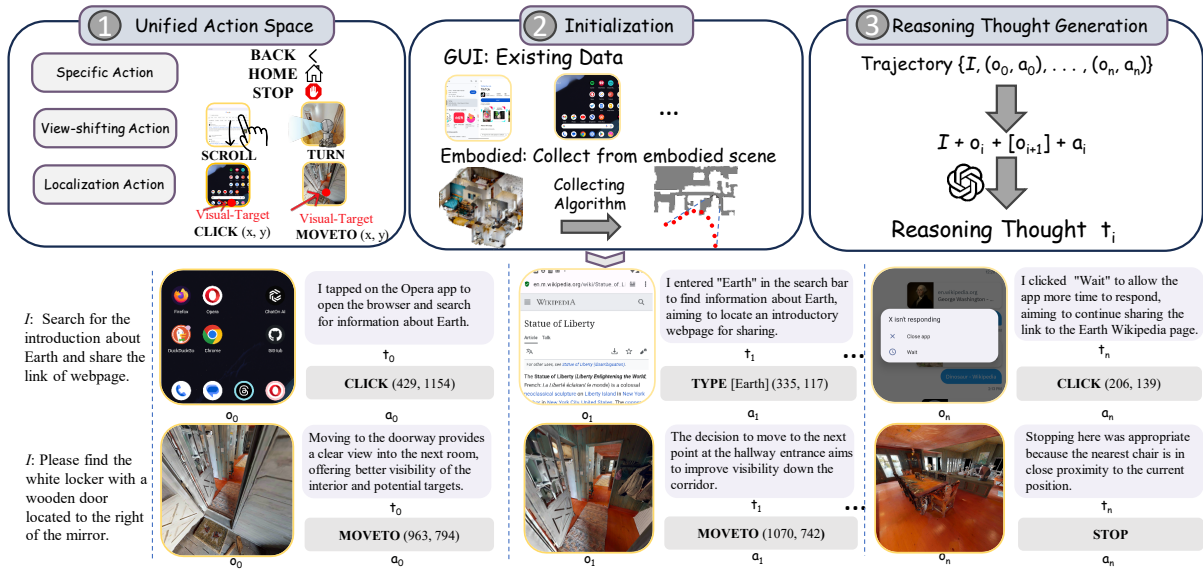


Figure 2: Visual-Target Trajectory Collection contains three parts. First, we unify the GUI and the Embodied action space by introducing a visual target at each step. Next, we initialize the trajectories using existing datasets or scenes. Last, we generate a first-person reasoning thought t_i with GPT-4o. Finally, we get our visual-target trajectories τ .

168 iting the agents’ capacity to generalize when the
 169 action space changes.

170 Recently, Embodied Web Agent (EWA) (Hong
 171 et al., 2025) is the first work that unifies physical
 172 embodiment with live web interfaces. Although
 173 EWA unifies web and embodied tasks, it lacks an
 174 emphasis on grounding capabilities and fails to es-
 175 tablish a comparable action space between the two
 176 navigation agent types. It also relies on zero/few-
 177 shot MLLMs without a unified navigation training
 178 paradigm, limiting its value for developing general-
 179 purpose navigation agents.

180 2.2 Reinforcement Fine-Tuning on MLLM

181 Visual-RFT (Liu et al., 2025b) performs reinforce-
 182 ment fine-tuning on LVLMS using their own reason-
 183 ing traces together with rule-based, verifiable
 184 visual rewards—e.g., IoU for detection and CLS ac-
 185 curacy for classification. UI-R1 (Lu et al., 2025) in-
 186 troduces a unified, rule-based reward that measures
 187 the click-coordinate accuracy within the ground-
 188 truth bounding box, thereby enhancing the pre-
 189 cision of GUI action prediction. GUI-R1 (Luo
 190 et al., 2025) also adopts a similar reward design,
 191 but places greater emphasis on high-level GUI na-
 192 vigation capabilities. However, their reward design
 193 is strictly binary; only responses that fall within the
 194 ground truth bounding box receive a positive score.
 195 This leads to many rollouts in GRPO yielding zero
 196 reward, making the training process less effective
 197 (Zheng et al., 2025a). Differently, we adopt a dense
 198 reward approach for grounding training in naviga-
 199 tion agents. Unlike prior work that relies on bi-

200 nary rewards, our method assigns scores based on
 201 the proximity of the response to the ground truth,
 202 thereby improving grounding performance while
 203 promoting more efficient and stable training.

204 3 NaviMaster

205 Our proposed NaviMaster consists of three key
 206 components, including (1) the visual-target trajec-
 207 tory collection to reformulate the GUI and embod-
 208 ied navigation trajectories into a unified form with
 209 historical information, (2) the unified reinforce-
 210 ment learning framework to optimize the cross-
 211 scenario at the same time, and (3) the distance-
 212 aware reward to update the model parameters by
 213 additionally considering the distance between out-
 214 put points and target points.

215 3.1 Visual-Target Trajectory Collection

216 As Fig. 2 shows, the visual-target trajectory collec-
 217 tion has three parts, including unified action space
 218 definition, unified trajectory initialization, and rea-
 219 soning thought generation.

220 **Unified Action Space Definition.** Existing GUI
 221 and embodied trajectory datasets exhibit substantial
 222 differences in their action spaces. We categorize ac-
 223 tions into three types: specific action, view-shifting
 224 action, and localization action. First, specific ac-
 225 tions have predefined, context-independent seman-
 226 tics (e.g., [BACK] in GUI, [STOP] in embodied
 227 tasks) and are directly integrated into the unified ac-
 228 tion space. Second, view-shifting actions adjust the
 229 agent’s viewpoint to locate targets outside the cur-
 230 rent field of view. This class encompasses actions

like [SCROLL] in GUI and [TURN] in embodied environments. We standardize these transformations into four directions for each domain: [up, down, left, right] for GUI and [left, right, around, back] for embodied agents. Finally, localization actions show the greatest divergence. In GUI tasks, the localization action is performed through the [CLICK (x, y)] action, where (x, y) denotes a specific target position on the screenshot. In contrast, embodied navigation tasks achieve localization via the [MOVEFORWARD] action, which does not require an explicit target. These differences reflect distinct interaction paradigms between the two localization actions. GUI actions \mathcal{A}_{gui} depend on precise, target-oriented operations (e.g., clicking specific UI elements), whereas embodied actions \mathcal{A}_{emb} focus on egocentric motion control (e.g., navigating without explicit target selection).

The discrepancy in localization actions (i.e., with or without a target) creates a challenge for unifying both tasks. To address this challenge, we propose the visual-target trajectory by introducing a localization action with an explicit target into the embodied navigation task. As shown in the left part of Fig. 2, we define a visual target within the observation at each step of the trajectory. Consequently, the localization action in embodied navigation is reformulated from [MOVEFORWARD] to [MOVETO (x, y)], where (x, y) denotes target location. The complete action space is in Appendix C.

Trajectory Collection Initialization. We consider a long-horizon task consisting of n steps, represented as $\{I, (o_0, a_0), \dots, (o_n, a_n)\}$, where I denotes the user-provided instruction, o_i is the observation from GUI screenshot or physical environment at step i , and a_i is the corresponding action at step i ($0 \leq i \leq n$). This representation is consistent with standard formulations of GUI trajectories, enabling direct reuse of existing GUI datasets. In our experiments, we leverage GUI-Odyssey (Lu et al., 2024) to obtain trajectory data.

However, the existing embodied navigation datasets typically provide only the initial and the target positions without specifying the intermediate trajectories. For an embodied navigation dataset (e.g., Matterport 3D dataset with the Habitat simulator (Yadav et al., 2023; Savva et al., 2019)), we extract the set of trajectory points along the shortest path from the initial to the target position, denoted as (s_0, s_1, \dots, s_m) , which can be mapped by performing the A* search method (Hart et al., 1968). Each trajectory point s_k ($0 \leq k \leq m$) is a 3D

coordinate in the global coordinate system.

Then, based on the point set, we collect observation images and generate visual-target actions for embodied navigation. The initialization procedure for trajectory collection is summarized in Algorithm 1. The first step is to align the next position with current observation. Given the current position $s_k(u_k, v_k, w_k)$ (global coordinate system), its camera rotation r_k and the next position $s_{k+1}(u_{k+1}, v_{k+1}, w_{k+1})$ (global coordinate system), $s'_{k+1}(u'_{k+1}, v'_{k+1}, w'_{k+1})$ (s_k coordinate system) can be obtained with following equation:

$$s'_{k+1} = r_k^{-1} \times (s_{k+1} - s_k) \times r_k. \quad (2)$$

After that, we project s'_{k+1} onto the current camera observation o_i :

$$p(x_i, y_i) = \left(\frac{W}{2} + f \cdot \frac{u'_{k+1}}{w'_{k+1}}, \frac{H}{2} + f \cdot \frac{v'_{k+1}}{w'_{k+1}} \right), \quad (3)$$

where $p(x_i, y_i)$ represents the pixel coordinates of the next position in the current observation o_i , (W, H) is the width and height of the image, and f is the camera focal length.

Due to the limitations of the camera’s pitch angle and field of view, the projected coordinates may not appear within the observation. To address this, we define several custom actions to adjust the camera angle, including the left-right and up-down turning actions in embodied tasks: [TURN left], [TURN right], [TURN around], [TURN down] and implement them as Algorithm 1. Let w'_{k+1} denote the depth of s'_{k+1} in the current observation. A negative value ($w'_{k+1} < 0$) indicates that s'_{k+1} lies behind the camera. After getting the observation o_i with the target at each position, we represent each embodied navigation trajectory with the same action space and style as in GUI navigation.

Reasoning Thought Generation. Historical information has been shown to be beneficial for agent performance (Yang et al., 2025c). Most existing approaches (Xu et al., 2025) take the implemented actions as history, which may lead to ambiguity. For example, the action [CLICK (x, y)] does not specify the context or purpose of the interaction. By contrast, pairing the reasoning thought “I should first open Chrome to start my search” with the corresponding action “[CLICK (x, y)]” explicitly indicates that agent opens the app Chrome. Some works (Qin et al., 2025) demonstrates that augmenting each step in the trajectory with its associated reasoning allows the model to articulate its decision-making process more transparently.

Algorithm 1 Trajectory Collection Initialization for Embodied Task

Input: $I, (s_0, s_1, \dots, s_m)$
Output: Trajectory

- 1: Initialize Trajectory with (I, i) with 0
- 2: **for** each $k \in [0, m]$ **do**
- 3: **if** $k < n$ **then**
- 4: Calculate $s'_{k+1}(w'_{k+1}, v'_{k+1}, w'_{k+1})$
- 5: **while** $p(x_i, y_i) \notin [0, W] \times [0, H]$ **do**
- 6: $o_i \leftarrow$ observation at (s_k, r_k)
- 7: Update $p(x_i, y_i)$
- 8: **if** $w'_{i+1} < 0$ **then**
- 9: $a_j \leftarrow$ TURN [around]
- 10: **else if** $x_i < 0$ **then**
- 11: $a_i \leftarrow$ TURN [left]
- 12: **else if** $x_i > W$ **then**
- 13: $a_i \leftarrow$ TURN [right]
- 14: **else if** $y_i > H$ **then**
- 15: $a_i \leftarrow$ TURN [down]
- 16: **end if**
- 17: Append (o_i, a_i) to Trajectory
- 18: Update $r_k \leftarrow$ execute a_i ; $i \leftarrow i + 1$
- 19: **end while**
- 20: $a_i \leftarrow$ MOVETO (x_i, y_i)
- 21: Append (o_i, a_i) to Trajectory; $i \leftarrow i + 1$
- 22: **else**
- 23: $o_i \leftarrow$ observation at (s_k, r_k) , $a_i \leftarrow$ STOP
- 24: Append (o_i, a_i) to Trajectory
- 25: **end if**
- 26: **end for**
- 27: **return** Trajectory

To enhance reasoning capability and optimize memory usage, we generate thought for each action in the trajectory, as illustrated in the bottom part of Fig. 2. Given an initialized trajectory, we construct the data generation pipeline as follows:

$$\langle I, o_i, a_i, [o_{i+1}] \rangle \xrightarrow{\mathcal{M}} t_i. \quad (4)$$

where the task instruction I , observation o_i , and action a_i are provided to the large language model \mathcal{M} , which produces an intention t_i from a first-person perspective to explain the rationale behind action a_i . In our experiments, \mathcal{M} corresponds to GPT-4o (OpenAI et al., 2024). The optional observation o_{i+1} is included only in GUI trajectories, where the target observation serves as a reference for data generation. The prompts used to generate these reasoning thoughts are provided in Appendix A.

Consequently, the visual-target trajectory for both GUI and embodied navigation tasks is represented as: $\tau = \{I, (o_0, t_0, a_0), \dots, (o_n, t_n, a_n)\}$.

3.2 Unified Reinforcement Learning Framework

Since reinforcement learning (RL) generally exhibits stronger generalization capabilities than supervised fine-tuning (SFT), we adopt the R1-Zero training strategy, directly training on our collected dataset with Group Relative Policy Optimization (GRPO). As illustrated in Fig. 3, given an n-step trajectory, we take step i as a

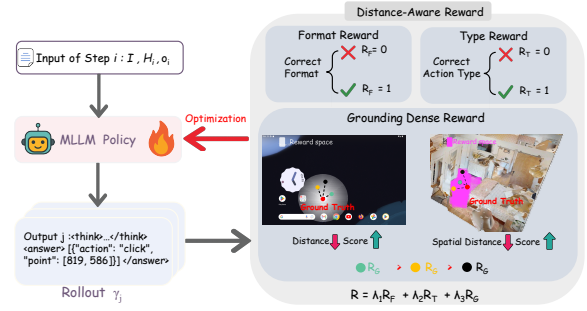


Figure 3: Overview of unified reinforcement learning framework. MLLM policy is optimized using GRPO with format, type and grounding dense reward.

data sample. Each sample consists of the user instruction I , the current observation o_i , reasoning thoughts and actions in history $H_i = \{(t_0, a_0), (t_1, a_1), \dots, (t_{i-1}, a_{i-1})\}$. NaviMaster then learns a unified policy with GRPO. Specifically, for the input queries $\{I, H_i, o_i\}$, we operate on G samples $\{\gamma_j = \pi_{\theta_{old}}(a_i | I, H_i, o_i)\}_{j=1}^G$ produced by the policy model π_{θ} . We also incorporate the depth map h_i of observation o_i as a critical prior for grounding in spatial space; for pure 2-D images h_i is set to the zero matrix. The advantage Adv is computed as follows:

$$R(i, j) = R(\gamma_j, a_i, h_i), \quad (5)$$

$$Adv = \frac{R(i, j) - \text{mean}(\{R(i, j)\}_{j=1}^G)}{\text{std}(\{R(i, j)\}_{j=1}^G)}, \quad (6)$$

where $R(i, j)$ denotes the reward function of the response. It will be detailed in the next section.

3.3 Distance-Aware Reward

The criteria for successful task execution are three-fold: (1) the model output must be correctly parsed into an executable action, (2) the type of the executed action must match the ground truth, (3) the action’s arguments must fall within valid bounds. Accordingly, we decompose the reward into three components: format, type, and grounding. While most existing reward designs employ binary success/failure signals, our approach aims to capture relative preferences even among unsuccessful rollouts. For instance, among unsuccessful rollouts, some may still be “better” than others. Specifically, we design a distance-aware dense reward for the grounding component based on the distance to the ground-truth point. Therefore, our reward consists three components, including format reward, type reward and grounding dense reward.

Format Reward. This reward $R_F(i, j)$ enforces the formatting of the output. Each response must

396 first provide a reasoning phase, followed by a final
 397 answer, where the answer must be a valid JSON
 398 string. The required structure is: “ $\langle think \rangle \dots$
 399 $\langle /think \rangle \langle answer \rangle json\ string \langle /answer \rangle$ ”. If
 400 a rollout satisfies this format, $R_F(i, j)$ will be set
 401 to 1. Otherwise, $R_F(i, j)$ will be set to 0.

402 **Type Reward.** This reward $R_T(i, j) = [\hat{a}_j = a_i]$
 403 evaluates the correctness of the model’s action se-
 404 lection. \hat{a}_j denotes the predicted action type for
 405 sample γ_j and a_i denotes the ground truth action
 406 type at step i . $[\cdot]$ stands for the Iverson bracket, an
 407 indicator that equals 1 when the statement inside
 408 is true and 0 otherwise. It is a binary reward that
 409 assesses whether the predicted action type matches
 410 the ground truth within a small, discrete action
 411 space. It supervises the model’s ability to make
 412 high-level decisions aligned with task semantics.

413 **Grounding Dense Reward.** This reward R_G is de-
 414 signed to guide model’s grounding ability. Specif-
 415 ically, this ability requires selecting the correct
 416 target within a large selection space, such as a
 417 pixel-level coordinate within an image. It eval-
 418 uates the predicted location relative to the ground
 419 truth at step i . To consistently measure grounding
 420 performance across navigation tasks, we define a
 421 distance-based dense reward instead of a sparse
 422 reward. It ensures that the agent receives higher
 423 rewards when its prediction is closer to targets (UI
 424 elements or position in embodied scene). Such de-
 425 sign can provide effective guidance during training.
 426 This reward function is as follows:

$$427 \quad R_G(i, j) = (1 - \frac{d_j}{\theta_d})[d_j < \theta_d, p_j < \theta_h] \quad (7)$$

428 where θ_d and θ_h are thresholds for distance d_j and
 429 depth disparity p_j . The d_j denotes the pixel-level
 430 distance between predicted point (\hat{x}_j, \hat{y}_j) of γ_j and
 431 the corresponding ground truth point (x_i, y_i) . In
 432 embodied environments, the depth value $h_i(\hat{x}_j, \hat{y}_j)$
 433 is also incorporated to account for potential occlu-
 434 sions: two pixels that are close in the 2D image
 435 may have substantially different depths in the 3D
 436 scene. The details of thresholds are in Appendix H.
 437 The definitions of d_j and p_j are as follows:

$$438 \quad d_j = \sqrt{(\hat{x}_j - x_i)^2 + (\hat{y}_j - y_i)^2}, \quad (8)$$

$$439 \quad p_j = |h_i(\hat{x}_j, \hat{y}_j) - h_i(x_i, y_i)|, \quad (9)$$

441 The overall reward function is a weighted combina-
 442 tion of the three components described above. Here,
 443 $R(i, j) = \lambda_1 R_F(i, j) + \lambda_2 R_T(i, j) + \lambda_3 R_G(i, j)$.
 444 $\lambda_1, \lambda_2, \lambda_3 \in \mathbb{R}_+$ controlling their relative impor-
 445 tance of each term.

446 4 Experiments

447 4.1 Implementation Details

448 We trained our model using the EasyR1 frame-
 449 work (Zheng et al., 2025b), adopting Qwen2.5VL-
 450 7B model (Bai et al., 2025) as the base model. The
 451 training is conducted for three epochs on 8 NVIDIA
 452 A800 GPUs, with a global batch size of 128 and
 453 a learning rate of 1×10^{-6} . The hyperparameters
 454 λ_1, λ_2 , and λ_3 are experimentally set to 0.1, 1, and
 455 1, respectively. The training utilizes 20k samples,
 456 including 10k GUI samples from GUI-Odyssey
 457 and 10k embodied samples from Matterport 3D
 458 and RoboPoint. All the experiments of NaviMaster
 459 use the same amount of data for a fair comparison.
 460 More details are provided in Appendix D.

461 4.2 Benchmarks and Metrics

462 **GUI task.** For the evaluation of the GUI task,
 463 we employ seven distinct mobile, web, desktop
 464 and in-domain benchmarks: AC-High/Low (Li
 465 et al., 2024), AITW (Rawles et al., 2023a), GUIAct-
 466 P/W (Chen et al., 2025), Llamatouch (Zhang et al.,
 467 2024b), AITZ (Zhang et al., 2024a) OmniAct-
 468 W/D (Kapoor et al., 2024) and Odyssey (Lu et al.,
 469 2024). We follow OS-Atlas (Wu et al., 2024) to
 470 take the success rate (SR) and grounding (GR) as
 471 the primary evaluation metrics. SR measures the
 472 per-step task success rate, while GR measures the
 473 accuracy of localizing the correct click coordinates.

474 We compare our model with the following meth-
 475 ods: the proprietary GPT-4o (OpenAI et al., 2024),
 476 SFT-based models such as OS-Atlas, Aguvus (Xu
 477 et al., 2025) and Qwen2.5VL-7B* fine-tuned on
 478 our trajectory data, as well as RL-based models
 479 like GUI-R1 (Luo et al., 2025), infiGUI-R1 (Liu
 480 et al., 2025a), UI-Shift (Gao et al., 2025) and UI-
 481 AGILE (Lian et al., 2025). We note the concurrent
 482 work of OmniActor (Yang et al., 2025b), which
 483 also proposes a unified agent for both 2D and 3D
 484 tasks. However, we exclude it from comparison
 485 because it is evaluated on a subset of benchmarks
 486 and its implementation is not publicly available.

487 **Embodied task.** We assess our model’s per-
 488 formance through two distinct embodied tasks.
 489 First, we conduct spatial affordance prediction to
 490 assess the model’s spatial grounding ability on
 491 the metric of SR. Specifically, we employ Ro-
 492 boRefT (Lu et al., 2023) for object referring
 493 and Where2Place (Yuan et al., 2024), RoboSpa-
 494 tial (Song et al., 2025), RefSpatial (Zhou et al.,
 495 2025) for free space referring. The second task is

Models	Mobile												Web				Desktop		ID	
	AC-Low		AC-High		AITW		GuiAct-P		Llamatouch		AITZ		GuiAct-W		OmniAct-W		OmniAct-D		Odyssey	
	GR	SR	GR	SR	GR	SR	GR	SR	GR	SR	GR	SR	GR	SR	GR	SR	GR	SR	GR	SR
GPT-4o	38.67	28.39	30.90	21.19	35.75	26.07	26.15	26.79	46.32	38.77	27.31	20.99	45.02	41.84	42.79	34.06	63.25	50.67	14.17	5.36
Qwen2.5VL-7B	87.08	62.50	59.71	47.06	65.99	45.80	55.79	44.99	75.84	60.97	60.79	33.88	90.78	78.08	84.40	71.91	79.42	57.58	68.52	37.32
Qwen2.5VL-7B*	66.73	43.82	51.29	31.79	62.51	48.81	52.21	47.75	83.88	45.56	58.36	31.18	60.26	43.26	47.61	44.36	50.58	42.29	40.96	29.38
OS-Atlas-7B	73.37	50.94	54.90	29.83	64.89	41.38	58.52	29.43	59.25	30.11	58.30	27.58	75.61	57.02	69.35	59.15	62.87	56.73	39.74	26.96
Aguvis	76.56	57.55	72.09	49.96	69.57	53.06	63.39	42.66	75.08	60.41	62.61	36.15	37.18	27.66	54.27	48.29	26.00	23.31	-	-
inhGUI-3B	93.20	92.10	74.40	71.10	75.58	46.51	66.67	40.09	75.02	58.66	75.42	40.81	86.02	64.60	75.42	54.91	73.25	47.17	70.30	33.15
GUI-R1-7B	84.02	66.52	70.31	51.56	62.94	55.31	47.09	48.62	69.82	61.27	64.98	49.06	88.06	74.54	84.87	68.69	81.19	57.70	71.10	36.22
UI-shift	94.22	73.38	73.41	52.16	74.44	54.38	59.10	52.12	75.08	61.71	73.84	46.78	89.49	79.43	82.93	64.22	80.01	57.94	62.54	32.75
UI-AGILE	93.71	63.32	77.55	50.97	75.72	48.66	60.11	42.27	78.31	66.10	75.89	38.74	90.33	69.57	84.04	63.15	81.94	59.35	70.40	36.96
Ours (w/o Embodied)	94.40	67.98	77.85	53.47	71.31	58.05	60.67	52.34	78.68	63.79	77.57	52.56	89.96	83.68	85.00	72.63	79.96	61.47	70.08	46.38
Ours (w/o GUI)	81.42	64.97	29.47	30.05	74.95	49.17	76.08	47.55	81.72	64.03	76.81	37.37	91.56	77.80	66.07	51.70	72.09	51.35	64.75	33.92
Ours	93.90	69.46	78.15	55.89	74.92	59.72	64.39	53.27	82.54	67.39	81.14	54.00	91.95	86.17	85.30	72.99	82.16	62.47	73.60	48.35

Table 1: Results on different GUI tasks. The red background represents that the data source is in the training set of the corresponding model, while the green background represents that the test dataset is OOD for the model. **Bold** highlights the best results in the OOD setting, and underlined are the second-best.

embodied navigation, which evaluates the model’s practical application capabilities on the metric of SR and SPL (Success Rate Weighted by Inverse Path Length). We evaluate our approach on the unseen validation branch of ObjectNav (Batra et al., 2020). We adhere to the framework established in VLMNav (Goetting et al., 2024) and substitute the agent model in our experiments to assess the performance. We compare our model against the proprietary GPT-4o, open-source methods like Qwen2.5VL-7B, SpatialVLM like SpaceLLaVA (Foutter et al., 2025), the latest method RoboPoint-13B (Yuan et al., 2024). More details of benchmarks and metrics are in Appendix E and Appendix F.

4.3 Main Results

GUI Navigation. The results are shown in Table 1. To evaluate the generalization capability of NaviMaster, we employ entirely out-of-domain (OOD) test data, highlighted with a green background, which are disjoint from the training distribution. The remaining test cases, indicated with a red background, are in-domain. Compared with state-of-the-art baselines, NaviMaster demonstrates consistently superior performance across the various benchmarks, demonstrating strong generalization capability and robustness when handling OOD datasets. Additionally, compared to models trained solely on either GUI or embodied data, our model trained on the mix data achieves the highest performance across all test datasets. This demonstrates the effectiveness of our collected visual-target trajectory and unified training framework, which enable strong generalization and competitive performance with a relatively small amount of data.

Spatial Affordance Prediction. We evaluate two types of spatial affordance predicting datasets: one is object referring, which involves identifying and localizing specific objects within a scene based

on language descriptions; the other is free space referring, which targets understanding and navigating to spatial regions or locations that are not necessarily tied to specific objects but are described in language. Table 2 summarizes the average success rate of predicted points falling within the ground-truth mask on the four spatial affordance prediction benchmarks. Compared to all baselines, NaviMaster performance best in all spatial affordance prediction tasks. These results demonstrate that NaviMaster’s fine-grained visual-spatial alignment significantly enhances performance in both object-level and free-space referring.

Models	RoboRefIt	Where2Place	RoboSpatial	RefSpatial
GPT-4o	15.28	29.00	5.70	8.40
Qwen2.5VL-7B	3.46	3.00	10.31	3.55
SpaceLLaVA	21.30	11.84	2.50	4.02
RoboPoint-13B	49.82	46.77	19.70	8.40
Ours (w/o Embodied)	67.86	35.05	14.75	16.88
Ours (w/o GUI)	76.23	43.02	19.83	18.19
Ours	77.34	52.97	21.65	19.49

Table 2: Results on spatial affordance prediction.

Embodied Navigation. Since we are the first to train a model capable of generalizing in VLMNav, there are no prior navigation models trained under VLMNav for direct comparison. We only report our results in Table 3, which reports performance on the ObjectNav benchmark. NaviMaster achieves the highest SR of 33.10% and SPL of 12.60%, representing a substantial improvement over the base model. Training exclusively on embodied data or GUI data yields slightly lower SR, indicating that the mix training strategy effectively leverages the complementary advantages of both data sources.

Runs	SR	SPL
Qwen2.5VL-7B	27.23	9.68
RoboPoint	13.01	2.80
SpaceLLaVA	14.98	2.62
Ours (w/o GUI)	31.10	11.20
Ours (w/o Embodied)	31.00	10.05
Ours	33.20	12.60

Table 3: Results on embodied navigation.

4.4 Discussions and Analysis

For all analysis experiments, we task AC-High/Low as GUI benchmarks and Where2Place, RefSpatial

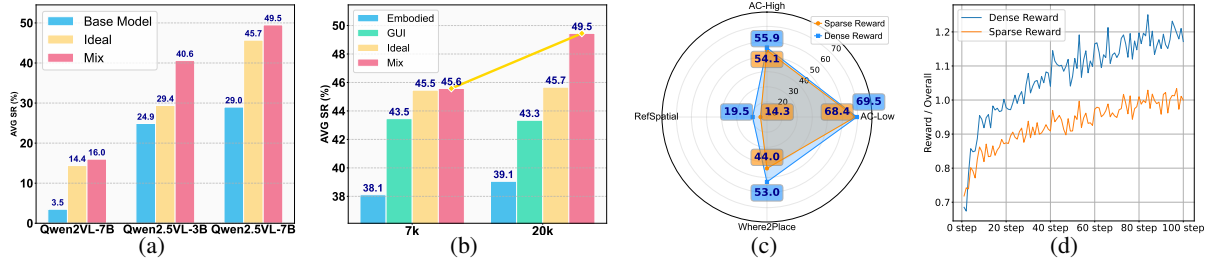


Figure 4: Ablation Studies. (a) Performance of different base models. (b) Performance of different data scales. (c) Performance of dense and sparse reward. (d) Reward curves.

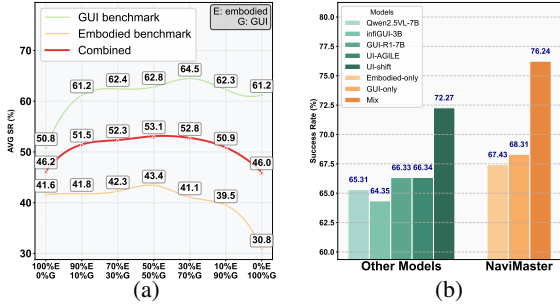


Figure 5: (a) Performance of different ratios of training data. (b) Performance on our visual-relation benchmark.

as Embodied benchmarks. More details and other ablation studies can be found in Appendix G.

Impact of data ratio. We investigate the impact of varying data mixing ratios on performance across the two tasks. As illustrated in Fig. 5a, we evaluate four distinct ratios (0:10, 1:9, 3:7 and 5:5) and report the average SR on both tasks. The results show that overall performance peaks at 5:5 ratio, demonstrating that our mixed training approach enhances cross-domain generalization. Notably, even with imbalanced mixtures, models trained on mix data outperform those trained on a single dataset.

Advantages of mixing. As stated in the introduction, to evaluate the visual reasoning enabled by our unified-training strategy, we manually constructed a visual-relation benchmark consisting of 100 instances from the AndroidControl dataset. Each instance contains a landmark UI element and a final target defined by its spatial relationship to that landmark. Importantly, the instructions require the model to infer the target location from a relational description (e.g., “Click on the 2nd button above the PUSH-UPS”) rather than directly naming the target. This design compels the model to reason over the image instead of merely learning a straightforward language-to-pixel mapping. As shown in Fig. 5b, the model trained with mix data achieves substantially higher accuracy than models trained on a single data type and also outperforms the existing RFT model with reasoning capabilities.

Base model. To verify that our improvements arise from unified training rather than a

strong base model, we evaluated the framework on Qwen2.5VL-3B (smaller parameter scale) and Qwen2VL-7B (Wang et al., 2024) (less pre-training knowledge). Fig. 4a presents the average success rate for each mode. Here, ‘Ideal’ denotes the average success rates of two specialist models, each trained and evaluated solely within its respective domain (GUI or embodied). Training on mix data consistently outperforms training on single data, regardless of the underlying model.

Data scales. We investigate the impact of data scale on our mixture strategy’s performance relative to single-task training. We evaluate two scales, 7k and 20k samples, while keeping the same training setting for all models. As shown in Fig. 4b, the mix-data approach consistently outperforms single-task training, demonstrating its benefits in both large-data regimes and relatively few samples.

Reward Design. We assess the effectiveness of the proposed grounding dense reward by replacing it with a sparse alternative. Specifically, the threshold is set to $\hat{\theta}_d = 20$ in the sparse reward setting, whereas $\theta_d = 200$ is used in the dense reward setting. For the sparse reward, if the predicted point’s distance to the ground-truth point is less than $\hat{\theta}_d$, the reward is 1; otherwise, it is 0. The results in the left of Fig. 4c show that the model trained with the dense reward consistently outperformed the one trained with the sparse reward. Moreover, the reward curve under dense setting in Fig. 4d rises more rapidly, indicating more efficient training.

5 Conclusion

We introduce NaviMaster, the first agent unifying GUI and embodied navigation in a single RL framework by reformulating both tasks into a visual-target trajectory format. This unification enables joint training and cross-task generalization. We propose a distance-aware dense reward to improve learning efficiency and spatial grounding capability. Experiments demonstrate NaviMaster achieves superior OOD generalization over prior methods.

636 Limitations

637 While our NaviMaster improves the performance
638 of both GUI and embodied navigation tasks signifi-
639 cantly, especially in OOD scenes, it still treats GUI
640 and embodied navigation as two different tasks.
641 Our trajectory dataset lacks any single trajectory
642 that interleaves GUI and embodied navigation tasks
643 due to the collection difficulty. Future work should
644 be constructing a navigation agent that supports in-
645 teracting with the GUI and embodied environments
646 at the same time.

647 Broader Impacts

648 The GUI or embodied data may leak personal in-
649 formation such as phone number or face. The data
650 collection pipeline we proposed will not introduce
651 any significant privacy such as personal informa-
652 tion. It will not contain any real information as
653 all the data source are virtual or from open-source
654 datasets.

655 The navigation agent will interact with the OS
656 system or real-world environments. This will po-
657 tentially affect the functioning of the system or take
658 risky actions to damage the environment. However,
659 all the settings in our experiments are in virtual
660 environments or on a monitor. We do not view this
661 as a concern.

662 References

663 Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wen-
664 bin Ge, Sibao Song, Kai Dang, Peng Wang, Shi-
665 jie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu,
666 Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei
667 Wang, Wei Ding, Zheren Fu, Yiheng Xu, and 8 oth-
668 ers. 2025. [Qwen2.5-vl technical report](#). *Preprint*,
669 arXiv:2502.13923.

670 Dhruv Batra, Aaron Gokaslan, Aniruddha Kembhavi,
671 Oleksandr Maksymets, Roozbeh Mottaghi, Manolis
672 Savva, Alexander Toshev, and Erik Wijmans.
673 2020. [Objectnav revisited: On evaluation of em-
674 bodied agents navigating to objects](#). *Preprint*,
675 arXiv:2006.13171.

676 Wentong Chen, Junbo Cui, Jinyi Hu, Yujia Qin, Junjie
677 Fang, Yue Zhao, Chongyi Wang, Jun Liu, Guirong
678 Chen, Yupeng Huo, Yuan Yao, Yankai Lin, Zhiyuan
679 Liu, and Maosong Sun. 2025. [Guicourse: From gen-
680 eral vision language models to versatile gui agents](#).
681 *Preprint*, arXiv:2406.11317.

682 Matthew Foutter, Daniele Gammelli, Justin Kruger,
683 Ethan Foss, Praneeet Bhoj, Tommaso Guffanti, Si-
684 mona D’Amico, and Marco Pavone. 2025. [Space-
685 llava: a vision-language model adapted to extrater-
686 restrial applications](#). *Preprint*, arXiv:2408.05924.

Longxi Gao, Li Zhang, and Mengwei Xu. 2025. [Uishift: Enhancing vlm-based gui agents through self-supervised reinforcement learning](#). *Preprint*, arXiv:2505.12493. 687-688-689-690

Peng Gao, Peng Wang, Feng Gao, Fei Wang, and Ruyue Yuan. 2024. [Vision-language navigation with embodied intelligence: A survey](#). *Preprint*, arXiv:2402.14304. 691-692-693-694

Dylan Goetting, Himanshu Gaurav Singh, and Antonio Loquercio. 2024. [End-to-end navigation with vision language models: Transforming spatial reasoning into question-answering](#). *Preprint*, arXiv:2411.05755. 695-696-697-698-699

Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. 1968. [A formal basis for the heuristic determination of minimum cost paths](#). *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107. 700-701-702-703

Yining Hong, Rui Sun, Bingxuan Li, Xingcheng Yao, Maxine Wu, Alexander Chien, Da Yin, Ying Nian Wu, Zhecan James Wang, and Kai-Wei Chang. 2025. [Embodied web agents: Bridging physical-digital realms for integrated agent intelligence](#). *Preprint*, arXiv:2506.15677. 704-705-706-707-708-709

Raghav Kapoor, Yash Parag Butala, Melisa Russak, Jing Yu Koh, Kiran Kamble, Waseem Alshikh, and Ruslan Salakhutdinov. 2024. [Omniact: A dataset and benchmark for enabling multimodal generalist autonomous agents for desktop and web](#). *Preprint*, arXiv:2402.17553. 710-711-712-713-714-715

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. [Efficient memory management for large language model serving with pagedattention](#). In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*. 716-717-718-719-720-721-722

Wei Li, William Bishop, Alice Li, Chris Rawles, Foflawiyo Campbell-Ajala, Divya Tyamagundlu, and Oriana Riva. 2024. [On the effects of data scale on ui control agents](#). *Preprint*, arXiv:2406.03679. 723-724-725-726

Shuquan Lian, Yuhang Wu, Jia Ma, Yifan Ding, Zihan Song, Bingqi Chen, Xiawu Zheng, and Hui Li. 2025. [Ui-agile: Advancing gui agents with effective reinforcement learning and precise inference-time grounding](#). *Preprint*, arXiv:2507.22025. 727-728-729-730-731

Bingqian Lin, Yunshuang Nie, Khun Loun Zai, Ziming Wei, Mingfei Han, Rongtao Xu, Minzhe Niu, Jianhua Han, Liang Lin, Cewu Lu, and Xiaodan Liang. 2025. [Evolvenav: Self-improving embodied reasoning for llm-based vision-language navigation](#). *Preprint*, arXiv:2506.01551. 732-733-734-735-736-737

Yuhang Liu, Pengxiang Li, Congkai Xie, Xavier Hu, Xiaotian Han, Shengyu Zhang, Hongxia Yang, and Fei Wu. 2025a. [Infogui-r1: Advancing multimodal gui agents from reactive actors to deliberative reasoners](#). *Preprint*, arXiv:2504.14239. 738-739-740-741-742

743	Ziyu Liu, Zeyi Sun, Yuhang Zang, Xiaoyi Dong, Yuhang Cao, Haodong Duan, Dahua Lin, and Jiaqi Wang. 2025b. Visual-rft: Visual reinforcement fine-tuning . <i>Preprint</i> , arXiv:2503.01785.	Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. 2019. Habitat: A platform for embodied ai research . <i>Preprint</i> , arXiv:1904.01201.	799
744			800
745			801
746			802
747	Quanfeng Lu, Wenqi Shao, Zitao Liu, Fanqing Meng, Boxuan Li, Botong Chen, Siyuan Huang, Kaipeng Zhang, Yu Qiao, and Ping Luo. 2024. Gui odyssey: A comprehensive dataset for cross-app gui navigation on mobile devices . <i>Preprint</i> , arXiv:2406.08451.	Chan Hee Song, Valts Blukis, Jonathan Tremblay, Stephen Tyree, Yu Su, and Stan Birchfield. 2025. Robospatial: Teaching spatial understanding to 2d and 3d vision-language models for robotics . <i>Preprint</i> , arXiv:2411.16537.	803
748			804
749			805
750			806
751			807
752	Yuhao Lu, Yixuan Fan, Beixing Deng, Fangfu Liu, Yali Li, and Shengjin Wang. 2023. Vl-grasp: a 6-dof interactive grasp policy for language-oriented objects in cluttered indoor scenes . <i>Preprint</i> , arXiv:2308.00640.	Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. 2024. Qwen2-vl: Enhancing vision-language model’s perception of the world at any resolution . <i>Preprint</i> , arXiv:2409.12191.	808
753			809
754			810
755			811
756	Zhengxi Lu, Yuxiang Chai, Yaxuan Guo, Xi Yin, Liang Liu, Hao Wang, Han Xiao, Shuai Ren, Guanqing Xiong, and Hongsheng Li. 2025. Ui-r1: Enhancing efficient action prediction of gui agents by reinforcement learning . <i>Preprint</i> , arXiv:2503.21620.		812
757			813
758			814
759			815
760			
761	Run Luo, Lu Wang, Wanwei He, and Xiaobo Xia. 2025. Gui-r1 : A generalist r1-style vision-language action model for gui agents . <i>Preprint</i> , arXiv:2504.10458.	Shuai Wang, Weiwen Liu, Jingxuan Chen, Yuqi Zhou, Weinan Gan, Xingshan Zeng, Yuhan Che, Shuai Yu, Xinlong Hao, Kun Shao, Bin Wang, Chuhan Wu, Yasheng Wang, Ruiming Tang, and Jianye Hao. 2025. Gui agents with foundation models: A comprehensive survey . <i>Preprint</i> , arXiv:2411.04890.	816
762			817
763			818
764	OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, and 401 others. 2024. Gpt-4o system card . <i>Preprint</i> , arXiv:2410.21276.		819
765			820
766			821
767			
768			822
769			823
770			824
771	Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, Wanjun Zhong, Kuanye Li, Jiale Yang, Yu Miao, Woyu Lin, Longxiang Liu, Xu Jiang, Qianli Ma, Jingyu Li, and 16 others. 2025. Ui-tars: Pioneering automated gui interaction with native agents . <i>Preprint</i> , arXiv:2501.12326.	Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, and Yu Qiao. 2024. Os-atlas: A foundation action model for generalist gui agents . <i>Preprint</i> , arXiv:2410.23218.	825
772			826
773			
774			827
775			828
776			829
777			830
778			831
779			832
780			
781			833
782			834
783			835
784			836
785			837
786			
787	Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. 2021. Habitat-matterport 3d dataset (HM3d): 1000 large-scale 3d environments for embodied AI . In <i>Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track</i> .	Yiheng Xu, Zekun Wang, Junli Wang, Dunjie Lu, Tianbao Xie, Amrita Saha, Doyen Sahoo, Tao Yu, and Caiming Xiong. 2025. Aguvis: Unified pure vision agents for autonomous gui interaction . <i>Preprint</i> , arXiv:2412.04454.	838
788			839
789			840
790			841
791			842
792			843
793			
794			844
795			845
796			846
797			847
798			848
799			
800			849
801			850
802			851
803			852
804			853
805			
806			
807			
808			
809			
810			
811			
812			
813			
814			
815			
816			
817			
818			
819			
820			
821			
822			
823			
824			
825			
826			
827			
828			
829			
830			
831			
832			
833			
834			
835			
836			
837			
838			
839			
840			
841			
842			
843			
844			
845			
846			
847			
848			
849			
850			
851			
852			
853			

854 Rui Yang, Hanyang Chen, Junyu Zhang, Mark
855 Zhao, Cheng Qian, Kangrui Wang, Qineng Wang,
856 Teja Venkat Koripella, Marziyeh Movahedi, Manling
857 Li, Heng Ji, Huan Zhang, and Tong Zhang. 2025c.
858 Embodiedbench: Comprehensive benchmarking
859 multi-modal large language models for vision-driven
860 embodied agents. *Preprint*, arXiv:2502.09560.

861 Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert
862 Pumacay, Ranjay Krishna, Adithyavairavan Murali,
863 Arsalan Mousavian, and Dieter Fox. 2024. **Robo-
864 point: A vision-language model for spatial affordance
865 prediction for robotics.** *Preprint*, arXiv:2406.10721.

866 Jiwen Zhang, Jihao Wu, Yihua Teng, Minghui Liao,
867 Nuo Xu, Xiao Xiao, Zhongyu Wei, and Duyu Tang.
868 2024a. **Android in the zoo: Chain-of-action-thought
869 for gui agents.** *Preprint*, arXiv:2403.02713.

870 Li Zhang, Shihe Wang, Xianqing Jia, Zhihan Zheng,
871 Yunhe Yan, Longxi Gao, Yuanchun Li, and Meng-
872 wei Xu. 2024b. **Llamatouch: A faithful and scal-
873 able testbed for mobile ui task automation.** *Preprint*,
874 arXiv:2404.16054.

875 Haizhong Zheng, Yang Zhou, Brian R. Bartoldson,
876 Bhavya Kailkhura, Fan Lai, Jiawei Zhao, and Beidi
877 Chen. 2025a. **Act only when it pays: Efficient re-
878 inforcement learning for llm reasoning via selective
879 rollouts.** *Preprint*, arXiv:2506.02177.

880 Yaowei Zheng, Junting Lu, Shenzhi Wang, Zhangchi
881 Feng, Dongdong Kuang, and Yuwen Xiong. 2025b.
882 Easyr1: An efficient, scalable, multi-modality rl train-
883 ing framework. [https://github.com/hiyouga/
884 EasyR1](https://github.com/hiyouga/EasyR1).

885 Enshen Zhou, Jingkun An, Cheng Chi, Yi Han, Shanyu
886 Rong, Chi Zhang, Pengwei Wang, Zhongyuan Wang,
887 Tiejun Huang, Lu Sheng, and Shanghang Zhang.
888 2025. **Roborefer: Towards spatial referring with
889 reasoning in vision-language models for robotics.**
890 *Preprint*, arXiv:2506.04308.

A Prompts for reasoning thought generation

Here are our prompts for generating reasoning thoughts.

Embodied Thought Generation Prompt

You are a robot in an unfamiliar environment. Now I want you to give the reason for your action. Your action can be in the following list:

- Based on the image, predict the optimal location to move next to finish the task. Use the coordinates (x, y) (x is the pixel from left to right and y is the pixel from top to bottom) to indicate where you want to move to: {"action_type": "move", "x": <position in horizontal (width)>, "y": <position in vertical (height)>}.
 - Turn left: {"action_type": "turn_left"}.
 - Turn right: {"action_type": "turn_right"}.
 - Turn around: {"action_type": "turn_around"}.
 - Move the camera angle downward: {"action_type": "look_down"}.
- Based on the image, if you find the target and the target is close enough, please stop to indicate that you want to stop: {"action_type": "stop"}.

You will be given the view before you performed the action (which has a text label "before" on the bottom right), the action you chose, and the task.

This is the action you performed: <action>
This is the task: <task/question> (the picture and action is one of the steps to finish the task)

By inspecting the picture and the action performed, give a brief reason of this step. You should carefully inspect the environment and give your analysis for why to do such action rather than other actions.

If moving to a position, explain why moving to that position based on the current environment. Avoid generic reasons like "get closer to the target."

NOTICES:

1. Coordinates are absolute coordinates (a center point defined by top-left and bottom-right coordinates).
2. If the action type is "move", the point will be labeled as "Next point" in the before image.
3. Remember that you should give the answer from a first-person perspective and keep it around 60 words and in a single line.
4. Don't limit yourself to begin with "I...". try any other possible sentence structure (like the position of exchanging description and target) if not influence the meaning."

GUI Thought Generation Prompt

You are an agent who can operate an Android phone on behalf of a user. Now I want you to give the reason for your action.

You will be given the screenshot before you performed the action (which has a text label "before" on the bottom right), the action you chose (together with the reason), and the screenshot after the action was performed (which has a text label "after" on the bottom right).

This is the action you picked: <q_text>

This is the task: <task> (The screenshots and action are one of the steps to finish the task)

This is the instruction: <instruction> (The instruction to solve the task)

This is the related apps: <apps> (Apps in the reason you output cannot go beyond the range of the app list) By comparing the two screenshots and the action performed, give a brief reason of this step. The reason should include the detailed description for the action and the target to do so, but avoid any description related to the after screenshot.

Requirements:

- Use first-person perspective.
- Keep the response around 60 words and in a single line.
- Do not begin every sentence with "I"; feel free to vary the structure as long as the meaning remains clear.

896

897

898

B Prompts for training

Here is our prompts for training NaviMaster.

Spatial-referring Prompt

Your answer should be formatted as a tuple, i.e. [x, y], where the tuple contains the x and y coordinates of a point satisfying the conditions above. Output the thinking process in <think> ... </think> tags, and the final answer in:

```
<answer>["action": "moveto", "point": [x, y]]</answer>
```

Note: The coordinates should be between the size of picture, indicating the absolute pixel locations of the points in the image.

Example:

```
["action": "moveto", "point": [123, 300]]
```

899

Navigation Prompt

"You are a Navigation Robot in an unfamiliar environment. In this photo <image>, the task is '{text}', with the history being '{history}'

You need to use your prior knowledge about where items are typically located within a home.

Please predict next action to find target item.

Your action can be in the following list:

Basic Action(move to a point on the ground in the picture):

- Based on the image, predict the optimal location to move next to finish the task, use the coordinates (x,

900

y)(x is the pixel from left to right and y is the pixel from top to bottom) to indicate where you want to move to:

```
[{"action": "moveto", "point": [x(position in horizontal(width)), y(position in vertical(height))]}].
```

View Adjustment Actions(adjust view as current photo does not have suitable position):

- Executes a 90-degree rotation to the left from the current facing direction. Ideal for navigating around obstacles on the right, aligning with a leftward path, or adjusting the view to inspect the left side of the environment. Use this when the task requires a lateral shift to the left:

```
[{"action": "turn_left"}].
```

- Rotates the perspective 90 degrees to the right. This action is useful when the target object or destination is positioned on the right, or when you need to change the direction to follow a rightward route:

```
[{"action": "turn_right"}].
```

- Performs a 180-degree rotation, flipping the orientation to face the opposite direction. This is valuable for finding a possible way if there is no path in front of you:

```
[{"action": "turn_around"}].
```

- Adjusts the camera view to look downwards, without physically moving the position. This is particularly useful for examining details on the ground, such as identifying objects, reading markings, or inspecting lower-level structures:

```
[{"action": "look_down"}].
```

Stop Action:

- Carefully inspect the environment and judge from history, if you find your target in your view and has been close enough for about 1 meter, stop at current position:

```
[{"action": "stop"}].
```

Output the thinking process in <think></think> tags, and the final answer in <answer></answer> tags as follows:

```
<think>... </think> <answer>answer here </answer>
```

Note: The 'point' should contain the coordinates of the next destination. Coordinates are absolute coordinates(a center point defined by top-left and bottom-right coordinates). Ensure the predicted Example:

```
{"action": "moveto", "point": [123, 300]}
```

901

GUI Prompt

A conversation between User and Assistant. The user asks a question, and the Assistant solves it step by step. The assistant first thinks about the reasoning process in the mind and then provides the user with the answer.

At each step, you will be given the current screenshot and the history of the conversation (include screenshot and action in each step). Based on these pieces of information and the goal, you must give the whole content of what you think and then choose to perform one of the actions in the following list (action description followed by the JSON format) by outputting the action in the correct JSON format. Click/tap on an element on the screen. We have defined the width and height of the screenshot, use the coordinates (x, y) (x is the pixel from left to right

902

and y is the pixel from top to bottom) to indicate which element you want to click, both x and y are integers:

```
["action": "click", "point": [x(position in horizontal(width)), y(position in vertical(height))]]
```

Long press on an element on the screen, similar with the click action above, use the coordinates (x, y) to indicate which element you want to long press:

```
["action": "long_press", "point": [x(position in horizontal(width)), y(position in vertical(height))]]
```

Type text into a text field (this action contains clicking on the target field, typing in the text and pressing the enter), use the coordinates (x, y) to indicate which element you want to click, both x and y are integers:

```
["action": "input_text", "text": <text_input>, "point": [x(position in horizontal(width)), y(position in vertical(height))]]
```

Navigate to the home screen:

```
["action": "navigate_home"]
```

Navigate back:

```
["action": "navigate_back"]
```

Scroll the screen or a scrollable UI element from start point to end point, use the coordinates (x, y) to indicate the two points you want to scroll:

```
["action": "scroll", "start_point": [<start position in horizontal(width)>, <start position in vertical(height)>], "end_point": [<end position in horizontal(width)>, <end position in vertical(height)>]]
```

NOTICES: 1.Coordinates are absolute coordinates (a center point defined by top-left and bottom-right coordinates). 2.The reasoning process and answer are enclosed within `<think> ...</think>` and `<answer> ...</answer>` tags, respectively. Example:

```
<think> reasoning process here </think>
<answer>["action": "click", "point": [378, 871]]</answer>
```

C Action Space

In our trajectory, the action space \mathcal{A}_{gui} for GUI task is defined as:

$$\mathcal{A}_{gui} = \begin{cases} \text{CLICK}(x, y) \\ \text{SCROLL}[up, down, right, left] \\ \text{LONGPRESS}(x, y) \\ \text{TYPE}[TEXT](x, y) \\ \text{HOME} \\ \text{BACK} \end{cases} \quad (10)$$

The action space \mathcal{A}_{emb} for embodied task is defined as:

$$\mathcal{A}_{emb} = \begin{cases} \text{MOVETO}(x, y) \\ \text{TURN}[left, right, around, down] \\ \text{STOP} \end{cases} \quad (11)$$

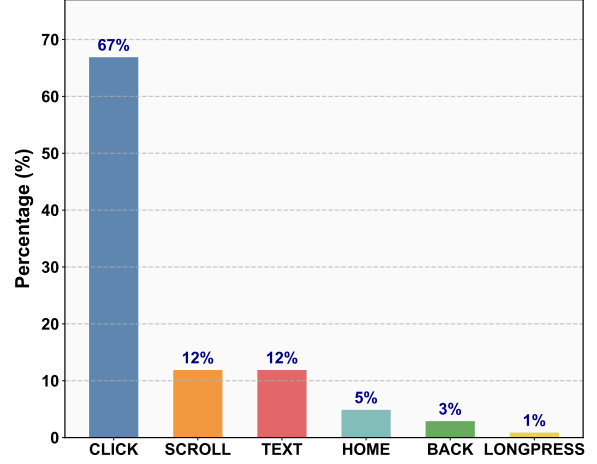


Figure 6: Data distribution of GUI-Odyssey

D Training Settings

D.1 Training Hyperparameter

To ensure the fairness of all comparative and ablation experiments, we maintained consistent hyperparameter settings throughout the training process, as detailed in Table 4.

Hyperparameter	Value
learning_rate	from 1e-6 to 0
temperature	1.0
num_generations	5
num_train_epochs	3
max_prompt_length	7000
max_response_length	1024
per_device_train_batch_size	4
gradient_accumulation_steps	16
KL coefficient	0.01
Reward coefficients $\lambda_1, \lambda_2, \lambda_3$	0.1, 1, 1

Table 4: Hyperparameter settings used for all reinforcement learning training.

In our implementation, whenever the concatenated trajectory exceeds 7000 tokens, we iteratively remove the minimum number of oldest steps until the prompt length drops just below the 7000 limit.

D.2 Sampling strategy

Selection bias could significantly affect the results. To avoid this, our sampling strictly follows the original distribution of GUI-Odyssey. We first computed the action-type distribution in the full 173k dataset. As shown in Fig. 6, when sampling the subset, we preserved this distribution, ensuring that the sampled data statistically matches the source dataset. Therefore, no additional bias was introduced during the sampling process.

E GUI Benchmark and Metrics Details

All the GUI benchmark are the test set of open-source dataset. The in-domain testing is from our GUI data source. We sample 4800 cases to test our in-domain performance.

In GUI task, we follow the settings in OS-Atlas, where a correct type prediction is considered accurate if the predicted action type matches the ground truth. For predictions involving grounding, an action is deemed correct if the predicted location falls within 14% of the image size relative to the ground truth.

Here, we argue that the metric of Type (the accuracy of the predicted action type) is unreliable due to dataset bias. As shown in Table 5, a naive model that predicts all actions as [CLICK] can still achieve a high Type prediction accuracy. Despite this limitation, we report Type results for completeness in Table 6.

Metric	AC-High/Low	AITW	GUIAct-Phone	LlamaTouch	AITZ
Type	59.7	57.2	58.0	64.4	55.9

Table 5: The bias of action types in testing datasets.

F Embodied Benchmark and Metrics Details

For details of benchmark:

Where2Place. This benchmark contains 100 real-world images to evaluate free space referring.

RoboSpatial. There are three branch in the benchmark: “Configuration”, “Context” and “Compatibility”. We take the “Context” branch to test free space referring.

RefSpatial. We take the unseen set of RoboSpatial. This set comprises 77 samples from the Location/Placement task.

Roboreft. We take the testA set of Roboreft.

For metrics, we introduce the average success rate of predicted points with in the groundtruth mask to evaluate the spatial grounding accuracy in the spatial referring task. This metric directly assesses the model’s ability to accurately localize the target based on the natural language description. For the navigation task, consistent with prior works, we utilize Success Rate (SR) and Success Rate Weighted by Inverse Path Length (SPL) as our metrics. SR measures the percentage of episodes that are successfully completed. Here, we set the success threshold to 0.3, meaning that stopping within this distance from the goal will be considered a success. SPL is a measure of navigation

path efficiency, which quantifies the agent’s performance by considering both task success and the path efficiency relative to the optimal path.

G Detailed Analysis

G.1 Statistical Stability and Error Statement

Our results are based on multiple runs and are empirically stable. All reported results are averaged over multiple trials, and to reduce variance during inference, we set the vLLM (Kwon et al., 2023) temperature to 0. Moreover, the success criteria for our tasks are range-based rather than relying on a single exact output. Therefore, even if model predictions vary slightly due to stochasticity, the evaluation outcome is generally unaffected.

G.2 Detailed statistics

The detailed results of base model ablation, data scales ablation, data scale ablation and reward design ablation is shown in the Table.7.

G.3 Embodied Data Source

Our embodied data consists of two parts: one part is derived from the point-based data we construct (trajectory), and the other part is the spatial affordance prediction data from RoboPoint (affordance). We evaluate the model when trained on each source individually and on their union (affordance + trajectory). As shown in the Table.7, Navimaster-robopoint the embodied data source only from affordance data. Navimaster-navi represents model embodied data are only from trajectory. Results show that combining both sources under an equal total data volume yields the best training performance.

G.4 Data Usage

Regarding dataset utilization, there are primarily two strategies. One is to mix different types of data into a single training phase (Mix). The other is to adopt a multi-stage schedule, with each stage focusing on one specific task or subset of data (GUI-Embodied or Embodied-GUI). Details were shown in the Table.7, Navimaster-1gui2embodied means training on GUI data at stage 1 and then on embodied data for stage 2. Navimaster-1embodied2gui represents model trained on embodied data first and then on GUI data. The average results shown in Fig. 7, the mix training strategy generally outperforms the two-stage training strategy across various benchmarks. This suggests that training with mix

	AC-Low	AC-High	AITW	GuiAct-P	Llamatouch	AITZ	GuiAct-W	OmniAct-W	OmniAct-D	Odyssey
Ours (w/o Embodied)	81.16	69.50	71.47	71.10	88.21	64.25	91.21	95.25	95.79	78.54
Ours (w/o GUI)	80.47	71.45	53.15	51.44	90.15	45.29	83.62	93.82	90.15	71.30
Ours	81.08	73.88	67.47	68.71	86.38	61.79	92.13	94.06	95.62	77.51

Table 6: TYPE results on different GUI tasks.

Models	AC-High	AC-Low	Where2Place	RefSpatial
Qwen2.5VL-3B	64.57/34.43	85.58/58.78	2.96	3.36
Navimaster-3B	66.02/45.26	79.85/60.41	14.83	15.59
Navimaster-3B(w/o GUI)	45.89/28.75	75.21/56.25	7.00	7.79
Navimaster-3B(w/o Embodied)	45.89/28.75	75.21/56.25	7.00	7.79
Qwen2VL-7B	19.11/2.30	17.98/7.24	3.00	1.30
Navimaster-qwen2vl	24.18/14.43	31.54/18.81	22.95	7.79
Navimaster-qwen2vl(w/o GUI)	19.37/13.04	26.46/15.71	15.98	9.09
Navimaster-qwen2vl(w/o Embodied)	23.53/13.87	30.65/18.74	15.01	1.30
Navimaster-7k	76.90/52.66	93.85/70.41	41.07	18.19
Navimaster-7k(w/o GUI)	26.86/29.06	77.78/63.37	44.94	15.08
Navimaster-7k(w/o Embodied)	74.83/52.34	94.13/69.48	37.73	14.28
Navimaster-1gui2embodied	74.87/54.83	94.19/70.09	47.77	18.82
Navimaster-1embodied2gui	74.55/54.73	93.64/71.04	47.97	19.21
Navimaster-hardreward	76.19/54.07	92.56/68.39	44.01	14.28
Navimaster-robopoint	73.32/52.34	93.94/68.94	52.04	19.67
Navimaster-navi	73.64/52.44	93.90/69.37	47.99	16.49
Navimaster-1_1_1	75.03/53.20	94.20/69.31	49.96	20.14
Navimaster-0.1_1_2	77.33/54.98	93.70/69.19	47.06	22.14
Navimaster-0.1_2_1	76.04/54.10	94.06/70.72	46.06	23.34

Table 7: Detailed Statistics

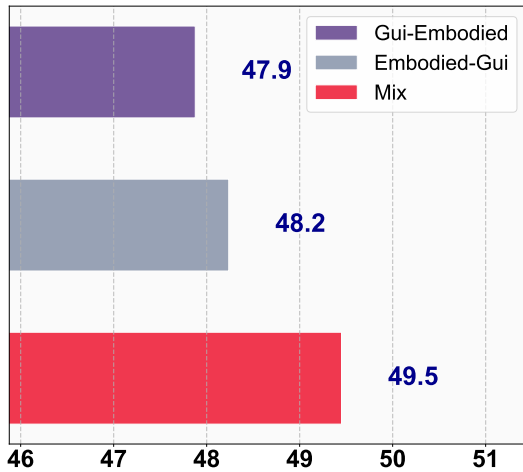


Figure 7: Performance of different data usage strategies.

data in a single phase enables the model to exploit complementary information effectively.

G.5 Hyperparameters

The ablation on hyperparameters focuses on the three reward weights, λ_1 , λ_2 , and λ_3 . We conducted three additional experiments with configurations different from the main setup: NaviMaster_0.1_1_2 ($\lambda_1 = 0.1$, $\lambda_2 = 1$, $\lambda_3 = 2$), NaviMaster_0.1_2_1 ($\lambda_1 = 0.1$, $\lambda_2 = 2$, $\lambda_3 = 1$), and NaviMaster_1_1_1 ($\lambda_1 = 1$, $\lambda_2 = 1$, $\lambda_3 = 1$). As shown in Table 7, the configuration used in our main experiments achieves the best overall performance among these variants.

G.6 Reward Components

We attribute the performance improvements to gains in both navigation accuracy and instruction

adherence. To decouple these factors, we provide the reward components ablation study in Table 8. It shows that removing the format or type reward leads to a drop (1–2 points) across all navigation benchmarks.

Reward	AC-High	AC-Low	Where2Place	RefSpatial
NaviMaster	55.89	69.46	52.97	19.49
w/o format reward	54.02	68.95	52.02	20.77
w/o type reward	55.66	70.60	48.99	18.18

Table 8: Ablation of Reward Components

H Discussion of Threshold

H.1 Standardized High Resolution

We did not use low-resolution images (e.g., 224×224 or 480×640) for embodied observations. Instead, we maintained a high-resolution standard: the minimum GUI resolution is 1280×720 , and all embodied observations are uniformly resized to 1980×1080 . This decision was deliberate, as prior works have shown that using high-resolution images during training and inference significantly enhances the model’s crucial grounding capability.

H.2 Rationale for Absolute Threshold

We chose an absolute threshold over a relative one to prevent severe reward hacking. If a relative threshold were used, on larger-resolution images, the acceptable error range would scale proportionally, making the success condition much easier. This would lead to many localization-inaccurate rollouts being incorrectly deemed "successful", which would seriously mislead the model’s training process. The absolute threshold of 200 pixels avoids this issue, ensuring that only highly accurate samples receive a positive reward, thereby preventing incorrect samples from skewing the training, even if it makes successful rollouts marginally more challenging on high-resolution data.

H.3 Alignment with Established Metrics

Furthermore, the $\theta_d = 200$ pixels threshold is not arbitrary; it is consistent with widely accepted evaluation metrics in the GUI community. Success in GUI tasks is commonly judged by whether the predicted action is within a certain area (e.g., 14%) of

the ground truth. Given our minimum resolution of 1280×720 , a 200-pixel radius circle aligns almost perfectly with this standard:

$$\pi \times (200^2) / (1280 \times 720) \approx 14\%$$

H.4 Training with relative threshold

We are prepared to conduct additional experiments using a relative threshold for reward definition. Specifically, we would normalize the model prediction and ground truth to a 0 – 1000 range and set the reward threshold to 140 for training. The Table 9 shows the advantage of an absolute reward threshold.

threshold	AC-High	AC-Low	Where2Place	RefSpatial
absolute	55.89	69.46	52.97	19.49
relative	53.61	67.56	51.99	18.48

Table 9: Comparison of relative and absolute threshold

H.5 Ablation of threshold θ_d

We provide experiments in Table 10, which evaluates different distance thresholds to further validate the robustness of our formulation.

θ_d	AC-High	AC-Low	Where2Place	RefSpatial
100 pixel	55.06	69.85	48.88	19.48
200 pixel	55.89	69.46	52.97	19.49
300 pixel	55.77	69.60	48.14	16.88
400 pixel	53.61	68.58	51.06	16.68

Table 10: Ablation of threshold θ_d

H.6 Ablation of threshold θ_h

We clarify that the introduction of depth into the reward signal is primarily intended for robustness, not for a direct enhancement of policy performance, thereby reducing the reliance on sensor priors as the main source of improvement. If the reward were based purely on 2D distance from the image projection, it would fail in corner cases—for example, when an object far away and an object nearby appear spatially aligned and close in the 2D image. Such misleading 2D calculations necessitate the introduction of depth to compute the true 3D distance, ensuring a robust and accurate proximity reward signal. We have conducted an ablation study on the effect of the depth signal on the overall performance in Table 11. These results confirm that the presence of the depth signal does not lead to a significant performance boost (it is not a primary source of gain). However, setting an overly strict depth threshold (θ_h) severely limits the reward availability and degrades performance, which

θ_h	AC-High	AC-Low	Where2Place	RefSpatial
w/o θ_h	55.09	69.46	52.16	19.49
0.1 m	51.56	67.98	36.84	16.88
1 m	55.89	69.46	52.97	19.49
10 m	55.89	69.46	52.97	19.49

Table 11: Ablation of threshold θ_h

supports our claim that depth’s role is to ensure robustness by preventing misleading 2D projection errors, not to artificially inflate the success rate.

I Case Study

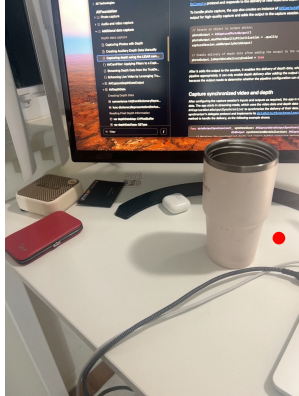
Fig. 8 illustrates the spatial referring task, where the agent must local the coordinates in the image according to the given instruction. Our case set comprises three free space referring instances and one object referring instance.

Fig. 9 illustrates the Embodied Navigation task, where the agent must follow the instruction to complete an entire navigation episode in Habitat.

Fig. 10 illustrates the AC-High task, where the agent receives a high-level user instruction and, based on the history and the current smartphone screenshot, outputs the action for the current step. Fig. 11 illustrates the general task, where the agent receives an user instruction and face both GUI and embodied environments. It outputs the thoughts and action for the current step.



In the image, there is a cup. Pinpoint point within the vacant space situated to the right of the cup.



Thought: To locate points in the vacant space to the right of the cup, I need to determine a point that is entirely to the right of the cup and within the designated area mentioned. The cup and the wires in front of it create a clear boundary for the vacant space of interest. Visually, the area to the right of the cup forms a rectangular space, and I can identify points within this space.

Action: MOVETO (1322, 1141)



Please point out the object between the white box and the farthest black pot.



Thought: The question asks for the object between the white box and the farthest black pot. The white box is located on the left side, and the farthest black pot appears to be on the right side, further back in the image. The object between these two is the blurry object in the middle, which appears to be a kitchen appliance or container

Action: MOVETO (123, 58)



Grab the green dinosaur model



Thought: The green dinosaur model is located on the left side of the chair, below the teddy bear and to the left of the two plastic cones.

Action: MOVETO (237, 276)



Locate one spot within the vacant area that is in front of the teal bowl.



Thought: The teal bowl is located towards the right side of the image. There is a vacant area in front of the teal bowl, which is the space between the bowl and the edge of the countertop. I need to identify one point within this vacant area.

Action: MOVETO (539, 340)

Figure 8: Case studies in RefSpatial, RoboSpatial, RoboRefit and Where2Place.



Navigate to the nearest sofa



Thought: From the image, I can see a wooden floor, a dining area with a table and chairs, a wall with a window and blinds, a door, and a room in the background. The dining area is likely near the central area of the house, and sofas are typically found in living rooms, which can be inferred from the hallway leading to the dining area. Since I see the dining area and a hallway leading further, I would guess that the living room is likely the next room, which may contain the sofa. The sofas are usually found in living rooms, not dining rooms or entrances. To get to the nearest sofa, I should proceed down the hallway, as the living room area is presumably the next room. I should avoid taking the actions that lead through closed doors or into areas that would take me away from the living room. The path to the living room is indicated by the hallway directly ahead, and I should move towards that direction. The labeled action that aligns with moving further down the hallway goes to position 2.

Action: MOVETO ②



Thought: In your observation, you can see a hallway with a couch and other pieces of furniture in the room ahead. The red arrow labeled 1 points to a chair, and the red arrow labeled 2 points to a general area near what appears to be the edge of the room, possibly a wall or corner. Since the task is to navigate to the nearest sofa, I should head towards the area labeled 1, as it is closer to the room with the sofa. The action that achieves this goal best is to move forward in the direction of the labeled 1.

Action: MOVETO ①



Thought: In the image, there is a living room area visible with a sofa on the right side. The sofa appears to be made of fabric and is partially covered with a blanket or throw. The red arrows and numbers in the image are pointing to different areas, but the sofa is clearly visible and within proximity of the highlighted area. Based on the visual information, the sofa is to the right of the robot's current position. The red arrow labeled "4" points closest to the sofa, indicating the direction to go for the best approach. Therefore, the robot should navigate to the right to get closer to the sofa.

Action: MOVETO ④



Thought: Based on the image, I see an indoor scene with a sitting area in the background. There are several pieces of furniture visible, but the couch is not immediately in view. The red arrow labeled "1" points towards the living area where the couch is likely to be located. The other text instructs me to turn around, which I will do to face the living area where the couch is presumably located. To achieve the task of navigating to the nearest sofa, the best action would be to move in the direction pointed towards by the red arrow labeled "1". This corresponds to moving forward into the living area.

Action: MOVETO ①



Thought: In the observation, the room contains a sofa on the left side near the entrance, a table with framed pictures, and a dining table on the right. The sofa is the closest visible piece of furniture to the entrance. To navigate to the nearest sofa and get as close to it as possible, I should move towards the left side of the image. The action labeled as 1 is the one that corresponds to the sofa's location. Therefore, taking action 1 will lead me towards the sofa.

Action: MOVETO ①

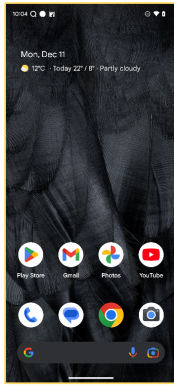


Thought: In the image, I see a living room with a section of a sofa on the right. The sofa is partially visible, and it's covered with a blanket and some pillows. The room has a window with blinds, a small table in the center, and some wall decor. The area is spacious and well-lit with natural light. The action arrows indicate possible movement paths, but since we are looking for the nearest sofa, let's focus on area 3 and 4. Action 4 seems to be leading towards the sofa, making it the best choice for approaching the nearest sofa. Therefore, my response is: {'action': 4}. This action will take me towards the nearest sofa.

Action: MOVETO ④

Figure 9: Case study in ObjectNav.

I want to search for a flight to Oxford, United Kingdom from Rotterdam for a vacation on the Omio app for the 13th of December.

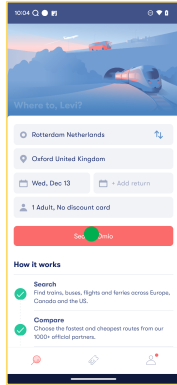


Step1



Thought: Since the Omio app is not visible on the home screen, I need to open the app drawer or search for the Omio app to proceed with the task of searching for a flight.

Action: OPENAPP [Omio]

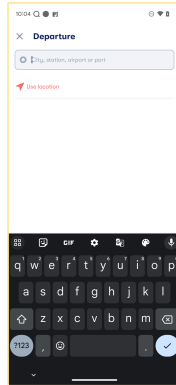


Step2



Thought: The current screen shows the departure city as Rotterdam and the destination city as Oxford, so no changes are needed in these fields. The next logical step would be to initiate the search by clicking the 'Search Omio' button.

Action: CLICK (546, 1474)

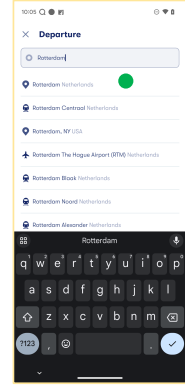


Step3



Thought: Since I need to search for a flight from Rotterdam to Oxford, my next step should be to type 'Rotterdam' as the departure city in the search bar.

Action: TYPE [Rotterdam]

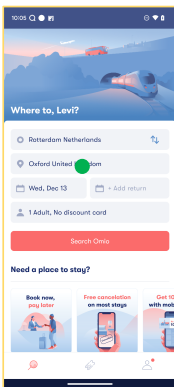


Step4



Thought: Since my goal is to book a flight from Rotterdam to Oxford, I need to select 'Rotterdam Netherlands' as the departure location to ensure the search results are accurate for flights departing from the correct city.

Action: CLICK (546, 527)

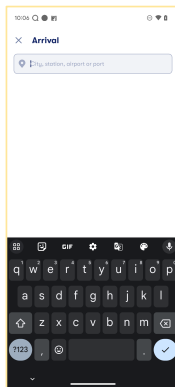


Step5



Thought: Since the departure location is already set to Rotterdam, I need to confirm the arrival location as Oxford, United Kingdom to proceed with the search for flights.

Action: CLICK (587, 986)

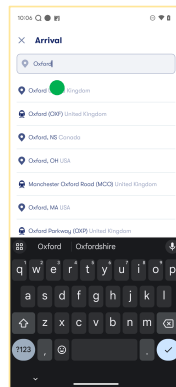


Step6



Thought: I need to enter the arrival city as 'Oxford' for the flight search, I should type 'Oxford' in the text box provided.

Action: TYPE [Oxford]

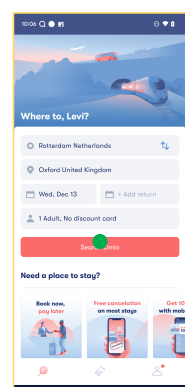


Step7



Thought: I need to select the correct suggestion that matches this destination. The 'Oxford United Kingdom', which seems to be the correct choice for my query.

Action: CLICK (387,526)



Step8



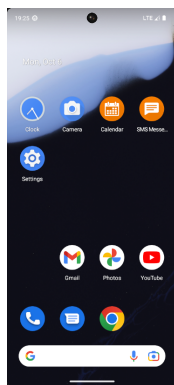
Thought: Since all the necessary details for the trip are entered correctly, I need to initiate the search to view available flights.

Action: CLICK (546, 1474)

Figure 10: Case study in AC-High.



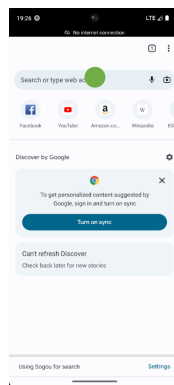
Please help me search on the browser for how to prune bonsai and navigate to the location of the plant.



Step1



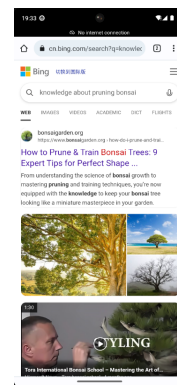
Thought: I need to open the app drawer or search for the Chrome app to proceed with the task of searching for pruning bonsai.
Action: OPENAPP [Chrome]



Step2



Thought: Now that I have entered the browser, I need to type "knowledge about pruning bonsai" in the search bar in the middle.
Action: TYPE [knowledge about pruning bonsai] (182, 153)



Step3



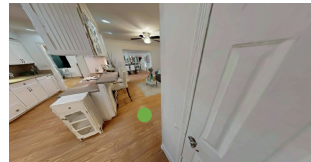
Thought: Since the image shows the search results for pruning bonsai, I have obtained the search results, and I should end the current task.
Action: STOP



Step4



Thought: To find the nearest plant, I should look for a location in the house where plants are typically kept. As the door appears to be closed, it won't allow for movement outside. I need to navigate back inside to search for a plant.
Action: TURN [left]



Step5



Thought: In the image, I see a kitchen area with white cabinets and a wooden floor. The plant is not visible in the immediate area, the best course of action is to move towards the corridor
Action: MOVETO (562, 473)



Step6



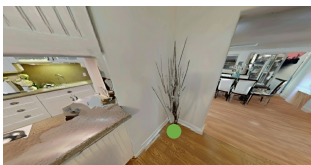
Thought: In the image, I observe a kitchen area and a living room. There are no plants in the immediate kitchen area. I need to move to living room, where a plant is more likely to be located.
Action: MOVETO (792, 460)



Step7



Thought: In the image, I see a living room with a sofa, a coffee table, and a painting of a horse on the wall. There are no visible plants in the image. To find the plant, I should move in the general direction of the dining area.
Action: MOVETO (240, 638)



Step8



Thought: There's a plant with tall dried stalks standing near the corner of the dining area, closer to the wall. To get as close as possible to the plant, I should move towards the dining area as it is the closest direction with a clear path to the plant.
Action: MOVETO (673, 541)



Step9



Thought: The image shows a close-up of a plant with long, thin leaves. The plant appears to be very close to the camera. I have finished the task.
Action: STOP

Figure 11: Case study in general task.