

Appendix for TrackVLA: Embodied Visual Tracking in the Wild

Contents

| | | |
|-----|--|-----------|
| 537 | 1 Introduction | 1 |
| 538 | 2 Related Works | 2 |
| 539 | 3 Method | 3 |
| 540 | 3.1 TrackVLA Architecture | 4 |
| 541 | 3.2 Implementation Details | 5 |
| 542 | 4 Data Collection | 5 |
| 543 | 4.1 Embodied Visual Tracking Data | 5 |
| 544 | 4.2 Video Question Answering Dataset | 6 |
| 545 | 5 Experiments | 6 |
| 546 | 5.1 Experiment Setups | 6 |
| 547 | 5.2 Quantitative Comparison | 7 |
| 548 | 5.3 Qualitative Results in Real-World | 7 |
| 549 | 5.4 Ablation Study | 8 |
| 550 | 6 Conclusions | 8 |
| 551 | 7 Limitations | 9 |
| 552 | A Training Details | 16 |
| 553 | B Inference Details | 16 |
| 554 | C EVT-Bench | 16 |
| 555 | C.1 Episode Generation | 16 |
| 556 | C.2 Evaluation | 17 |
| 557 | C.3 Metric Definitions | 17 |
| 558 | C.4 Humanoid Avatar Gallery | 17 |
| 559 | C.5 Visualization of Training Data | 18 |
| 560 | C.6 Qualitative Results on EVT-Bench | 18 |
| 561 | D Detailed Experiments of Gym-UnrealCV | 18 |
| 562 | D.1 Evaluation | 18 |
| 563 | D.2 Metric Definitions | 18 |
| 564 | D.3 Testing Scenes | 18 |
| 565 | D.4 Baselines | 18 |
| 566 | D.5 Experiment Results | 19 |
| 567 | D.6 Visualization of Humanoid Avatar in Gym-UnrealCV | 19 |
| 568 | D.7 Qualitative Results on Gym-UnrealCV | 20 |
| 569 | E Visual Recognition Experiment | 20 |
| 570 | E.1 Baselines | 20 |
| 571 | E.2 Evaluation | 20 |
| 572 | F More Ablation Study | 20 |
| 573 | F.1 Action Model Architecture | 20 |
| 574 | F.2 History Window Length | 21 |
| 575 | F.3 Future Trajectory Horizon | 21 |
| 576 | F.4 Human Recognition Dataset | 21 |
| 577 | G Real-world Deployment | 22 |
| 578 | G.1 Robot Platform | 22 |
| 579 | G.2 Real-world System Architecture | 22 |
| 580 | H Real-world Experiments | 22 |

A Training Details

Similar to conventional vision-language models (VLMs), TrackVLA follows a two-stage training pipeline. In the first stage, we train the projector of the visual encoder using a large amount of image-caption data [57] to align the visual embedding space with the LLM’s latent space. In the second stage, we jointly train the visual projector, the large language model, and the action model using a mixture of the training data. During training, we truncate the diffusion schedule of the action model to at most 50 out of a total of 1000 steps to diffuse the trajectory anchors, which introduces only a small amount of noise.

TrackVLA is trained on a cluster server equipped with 24 NVIDIA H100 GPUs for approximately 15 hours, totaling 360 GPU hours. The vision encoder (EVA-CLIP [56]) and the large language model (Vicuna-7B [47]) are initialized with their respective pretrained weights, and the vision encoder remains frozen throughout the entire training process. Following standard VLM practices, we train the model for only one epoch. The training is conducted with a learning rate of $2e-5$, a total batch size of 196, and a cosine learning rate schedule with linear warm-up. We use the AdamW optimizer for optimization. See Table 5 for detailed parameter settings.

B Inference Details

During inference, each input frame is resized to 224×224 and fed into the vision encoder. After obtaining visual tokens, we organize the tokens according to the task type. For the embodied visual tracking task, we prepend a special [Track] token before the instruction tokens and perform only a single-step autoregression with the LLM. The final-layer hidden state output from the LLM is then passed to the action model. We apply 10 out of 1000 diffusion steps to the trajectory anchors and use DDIM to perform 2 denoising steps, resulting in a set of predicted trajectories and corresponding score vectors. The trajectory corresponding to the anchor with the top 1 score is selected as the final output. For the VQA task, we follow the standard autoregressive decoding process of the LLM, and the language modeling head detokenizes the predicted tokens into textual answers. See Table 5 for detailed parameter settings.

| Notation | Shape & Params. | Description |
|--------------|------------------|--|
| lr | $2e-5$ | learning rate |
| B | 196 | batch size |
| T | 1000 | total diffusion steps |
| T_{train} | 50 | number of noise addition steps during training |
| T_{infer} | 10 | number of noise addition steps during training |
| N_{step} | 2 | denoising steps during inference |
| \mathbf{X} | 224×224 | input observation size |
| N | 256 | the number of image patch |
| C | 1408 | embedding dimension of visual feature |
| α | 1 | balancing parameter 1 |
| λ | 100 | balancing parameter 2 |
| M | 40 | the number of trajectory anchors |
| N_w | 10 | the number of waypoints |

Table 5: Hyperparameters and notation used in our model.

C EVT-Bench

C.1 Episode Generation

For each episode, we first sample a motion trajectory for the target humanoid avatar within the navigable area. Each trajectory consists of a start point, a random number of intermediate waypoints

611 (0–2), and an end point. The distance between any two consecutive waypoints must exceed a pre-
 612 defined minimum threshold $d_{min} = 3$ m. After generating the trajectory for the target avatar, the
 613 agent is placed near the target’s starting point, with its initial orientation roughly facing the target
 614 but perturbed by a random offset within 30° . For the *DT* and *AT* tasks, distractors are initialized
 615 near the target’s trajectory, and their paths are designed to intersect with the target’s trajectory as
 616 much as possible to enhance the level of distraction.

617 C.2 Evaluation

618 In each episode, the target humanoid and distractors move along their predefined trajectories. The
 619 evaluated algorithm receives the agent’s observation at each time step and performs inference to
 620 generate the corresponding control command, consisting of linear and angular velocities of the agent.
 621 The agent then moves according to the speed command. The episode terminates when the target
 622 humanoid reaches its destination or when the agent collides with the humanoid.

623 C.3 Metric Definitions

- 624 • **Success Rate (SR):** This metric evaluates the agent’s tracking ability. An episode is considered
 625 successful if, by its end, the agent remains oriented toward the target and maintains a safe
 626 distance of 1–3 meters. The success rate is defined as the proportion of successful episodes
 627 over the total number of episodes.
- 628 • **Tracking Rate (TR):** This metric evaluates the tracking quality of the agent. It is defined as
 629 the proportion of steps S where the agent successfully tracks the target to the total number of
 630 steps L , i.e., $TR = S/L$.
- 631 • **Collision Rate (CR):** This metric evaluates the safety of the agent. It is defined as the propor-
 632 tion of episodes that terminate due to a collision between the agent and the target humanoid
 633 avatar.

634 C.4 Humanoid Avatar Gallery

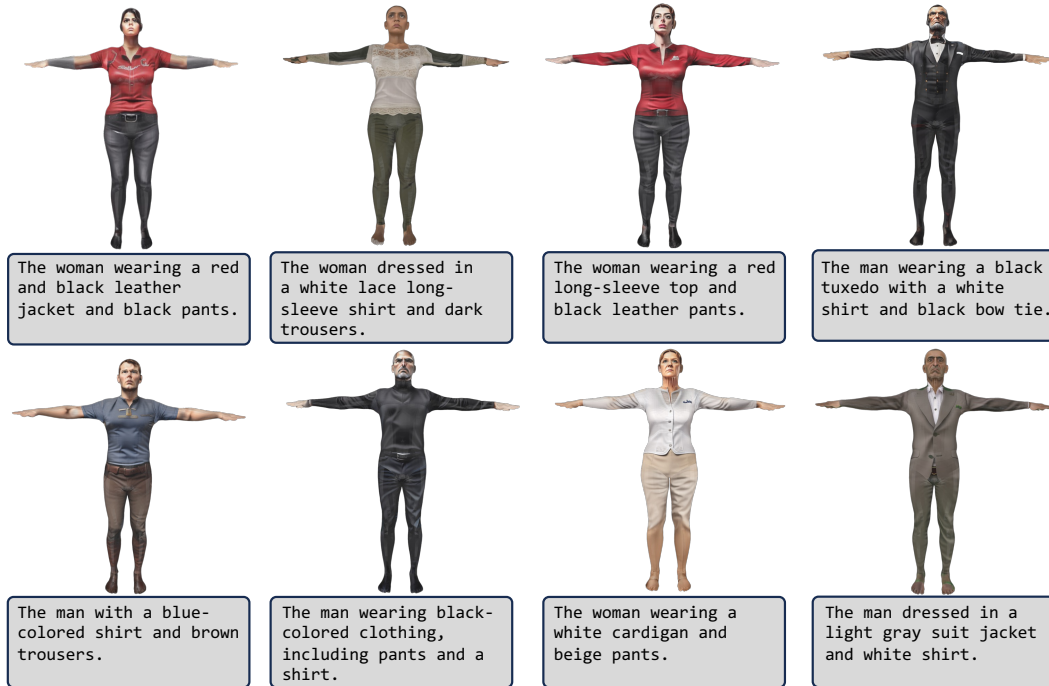


Figure 7: Visualization of the custom humanoid avatars with captions.

635 We provide visualizations of several custom-designed humanoid avatars paired with their descriptive
636 captions, as shown in Fig. 7

637 C.5 Visualization of Training Data

638 We provide visualizations of the training set of self-built EVT-Bench, as shown in Fig. 11

639 C.6 Qualitative Results on EVT-Bench

640 We provide visual results of TrackVLA on EVT-Bench, shown in Fig. 12.

641 D Detailed Experiments of Gym-UnrealCV

642 D.1 Evaluation

643 The evaluation setting of Gym-UnrealCV follows [6], where each episode has a maximum length of
644 500 steps. The agent’s tracking region is defined as a 90-degree fan-shaped sector with a radius of
645 750 cm. Success is achieved if the agent keeps the target within this region for the entire episode. A
646 failure occurs if the target remains outside the region for more than 50 consecutive steps.

647 D.2 Metric Definitions

- 648 • **Episode Length (EL)**: The average number of steps per episode over 100 episodes, reflecting
649 the model’s long-term tracking capability under predefined termination conditions.
- 650 • **Success Rate (SR)**: The percentage of successful episodes out of the total 100 episodes, mea-
651 suring the model’s robustness in active visual tracking.

652 D.3 Testing Scenes

- 653 • **SimpleRoom**: A basic environment designed to verify the model’s fundamental tracking capa-
654 bility.
- 655 • **Parking Lot**: An environment featuring occlusions and low-light conditions.
- 656 • **UrbanCity**: A typical urban street scene with reflective road surfaces.
- 657 • **UrbanRoad**: Similar to UrbanCity with fewer obstacles.
- 658 • **Snow Village**: A challenging terrain with uneven surfaces and complex backlighting.

659 D.4 Baselines

- 660 • **DiMP [71]**: Utilizes a pre-trained video tracker to generate target bounding boxes as scene
661 representations and applies a PID controller for motion control.
- 662 • **SARL [24]**: An online reinforcement learning (RL) approach that encodes RGB observations
663 into latent visual features and trains an end-to-end Conv-LSTM policy via RL.
- 664 • **AD-VAT [3]**: Introduces an asymmetric dueling mechanism and trains an RL-based tracker
665 with a learnable adversarial target to improve robustness.
- 666 • **AD-VAT+ [4]**: An enhanced version of AD-VAT that incorporates a two-stage training scheme,
667 aiming to improve performance in cluttered and obstacle-rich environments.
- 668 • **TS [5]**: A teacher-student framework that extends behavior cloning by leveraging a pose-based
669 teacher to provide real-time supervision for the vision-based student policy during interaction.
- 670 • **EVT [6]**: An offline RL-based framework designed for dynamic target following, which inte-
671 grates vision foundation models to enhance perception and robustness.
- 672 • **IBVS [70]**: A model-based method that takes the target bounding box as input and applies a
673 Kalman filter-based visual servoing algorithm to follow the target.

- **PoliFormer** [26]: A reinforcement learning-based navigation framework that explicitly encodes target bounding boxes into the observation space to enhance tracking accuracy.
- **Uni-NaVid** [14]: A unified vision-language-action (VLA) model designed for general navigation tasks, including human following.

D.5 Experiment Results

Single Human Tracking Evaluation The single human tracking task spans five distinct environments mentioned above, covering a wide range of variations in lighting conditions, viewpoints, and scene layouts. As shown in Table 6, TrackVLA achieves state-of-the-art performance across all five environments and successfully passes all test cases. Notably, TrackVLA is trained without any data from this simulator, highlighting its strong generalization under a **zero-shot** transfer setting.

| Methods | SimpleRoom | Parking Lot | UrbanCity | UrbanRoad | Snow Village | Mean |
|---------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| DiMP | 500/1.00 | 327/0.48 | 401/0.66 | 308/0.33 | 301/0.43 | 367/0.58 |
| SARL | 500/1.00 | 301/0.22 | 471/0.86 | 378/0.48 | 318/0.31 | 394/0.57 |
| AD-VAT | 500/1.00 | 302/0.20 | 484/0.88 | 429/0.60 | 364/0.44 | 416/0.62 |
| AD-VAT+ | 500/1.00 | 439/0.60 | 497/0.94 | 471/0.94 | 365/0.44 | 454/0.76 |
| TS | 500/1.00 | 472/0.89 | 496/0.94 | 480/0.84 | 424/0.63 | 474/0.86 |
| RSPT | 500/1.00 | 480/0.80 | 500/1.00 | 500/1.00 | 410/0.80 | 478/0.92 |
| EVT | 500/1.00 | 484/0.92 | 500/1.00 | 496/0.96 | 471/0.87 | 490/0.95 |
| Ours | 500/1.00 | 500/1.00 | 500/1.00 | 500/1.00 | 500/1.00 | 500/1.00 |

Table 6: **Quantitative results compared with baselines in unseen environments.** The two metrics of each cell represent the Average Episode Length (EL) and Success Rate (SR).

Distraction Robustness Evaluation In this experiment, distractors with appearances identical to the target are introduced, requiring the agent to consistently track the first observed target. TrackVLA addresses this challenge using instructions such as “Follow the first person you see”. Experimental results in Table 7 show that TrackVLA achieves state-of-the-art performance across all scenarios, demonstrating its strong capability in understanding and reasoning about human motion.

| Methods | Parking Lot (2D) | UrbanCity (4D) | ComplexRoom (4D) |
|---------|------------------|-----------------|------------------|
| DiMP | 271/0.24 | 348/0.32 | 307/0.26 |
| SARL | 237/0.12 | 221/0.16 | 263/0.15 |
| AD-VAT | 232/0.13 | 204/0.06 | 223/0.16 |
| AD-VAT+ | 166/0.08 | 245/0.11 | 262/0.18 |
| TS | 331/0.39 | 381/0.51 | 401/0.54 |
| EVT | 425/0.63 | 472/0.92 | 479/0.88 |
| Ours | 467/0.90 | 476/0.92 | 479/0.91 |

Table 7: **Evaluating the distraction robustness in the environment with distractors.** (4D) represents that there are 4 distractors in the environment.

Unseen Object Generalization Evaluation We further evaluate the object-level generalization ability of TrackVLA using the Gym-UnrealCV benchmark. Specifically, in the *SimpleRoom* environment, we test the model’s tracking performance on four unseen animal categories: horse, dog, sheep, and pig. As shown in Table 8, TrackVLA successfully tracks all four categories, consistent with its performance on the single-person tracking task. This demonstrates its strong generalization capability to novel object types.

D.6 Visualization of Humanoid Avatar in Gym-UnrealCV

We showcase several humanoid avatars used in Gym-UnrealCV in Fig. 8.

| Methods | Horse | Dog | Sheep | Pig |
|---------|-----------------|-----------------|-----------------|-----------------|
| EVT | 500/1.00 | 469/0.90 | 471/0.93 | 472/0.94 |
| Ours | 500/1.00 | 500/1.00 | 500/1.00 | 500/1.00 |

Table 8: **Evaluating the generalization on the unseen category of the target in SimpleRoom.** We directly adopt the agent on the unseen animals: horse, dog, sheep, and pig.



Figure 8: Examples of humanoid avatars used in Gym-UnrealCV.

697 D.7 Qualitative Results on Gym-UnrealCV

698 We provide visual results of TrackVLA on Gym-UnrealCV, shown in Fig. 13.

699 E Visual Recognition Experiment

700 E.1 Baselines

- 701 • **RexSeek** [74]: A Multimodal Large Language Model (MLLM) designed to detect people or
702 objects in images based on natural language descriptions.
- 703 • **LISA++** [75]: A Multimodal Large Language Model capable of both language understanding
704 and mask generation.
- 705 • **SoM+GPT-4o** [72, 73]: A visual prompting method that guides large multimodal models like
706 GPT-4o to perform visual grounding by overlaying segmented image regions with identifiable
707 marks.

708 E.2 Evaluation

709 During testing, each test image contains two *unseen* persons positioned on the left and right sides,
710 and two corresponding descriptions are provided for each person. Given the differing output formats
711 of the evaluated methods, we define task-specific evaluation criteria. RexSeek is an object detection
712 model that outputs bounding boxes; we evaluate its performance by checking whether the predicted
713 box correctly selects the target person. LISA++ is an instance segmentation model that outputs a
714 mask for the target; we assess whether the mask covers the correct individual. The SoM+GPT-
715 4o pipeline first performs image segmentation, then uses SoM to overlay numerical marks on the
716 original image at the location of each segmentation mask. The annotated image is then passed to
717 GPT-4o, which selects the number that best matches the given description. For this method, we
718 evaluate whether the mask corresponding to the selected mark covers the correct individual. As for
719 TrackVLA, which outputs a future trajectory, we determine correctness by checking whether the
720 trajectory direction aligns with the corresponding target person.

721 F More Ablation Study

722 F.1 Action Model Architecture

723 The architectures evaluated in this ablation study include Multi-Layer Perceptrons (MLPs) with
724 3 and 6 layers, respectively, as well as diffusion transformers of varying scales. The hidden state
725 dimensions for the two MLPs are set to 1024 and 4096. The base diffusion transformer is configured

with a depth of 12, hidden size of 768, and 12 attention heads, while the small diffusion transformer uses a depth of 6, hidden size of 384, and 4 attention heads.

| Model | Params. | SR \uparrow | TR \uparrow | CR \downarrow | time(ms) \downarrow |
|----------------|---------|---------------|---------------|-----------------|-----------------------|
| Autoregressive | 131M | 42.6 | 56.9 | 11.7 | 460 |
| MLP (3-Layers) | 7M | 45.8 | 59.9 | 10.1 | 0.5 |
| MLP (6-Layers) | 89M | 52.7 | 61.9 | 9.42 | 0.8 |
| DP-Base | 89M | 17.9 | 33.8 | 27.7 | 65 |
| Ours-Small | 13M | 49.8 | 60.2 | 6.67 | 8 |
| Ours-Base | 89M | 57.6 | 63.2 | 5.80 | 13 |

Table 9: Comparison of different action models.

727

728 F.2 History Window Length

729 Incorporating historical observations helps the model better infer the target’s motion pattern and
730 relative position. Here, we investigate how varying the length of the history observation window
731 L_{his} affects model performance. Table 10 shows that removing history observations leads to a
significant performance drop. We empirically select 32 as the optimal window length.

| L_{his} | SR \uparrow | TR \uparrow | CR \downarrow |
|-----------|---------------|---------------|-----------------|
| 0 | 29.9 | 49.6 | 6.94 |
| 32 | 57.6 | 63.2 | 5.80 |
| 64 | 56.5 | 63.3 | 6.49 |

Table 10: Comparison of different history window lengths.

732

733 F.3 Future Trajectory Horizon

734 TrackVLA predicts a future trajectory consisting of L_{traj} waypoints. In Table 11, we investigate the
735 impact of varying the number of predicted waypoints on overall performance. Experimental results
show that using 10 waypoints yields the best performance.

| L_{traj} | SR \uparrow | TR \uparrow | CR \downarrow |
|------------|---------------|---------------|-----------------|
| 1 | 44.3 | 60.6 | 14.4 |
| 10 | 57.6 | 63.2 | 5.80 |
| 20 | 51.3 | 60.2 | 7.54 |

Table 11: Comparison of different predicted waypoint lengths.

736

737 F.4 Human Recognition Dataset

738 Furthermore, we investigate the impact of different types of human recognition data on the model’s
739 recognition capability. Specifically, we categorize the data into three types: Single Human, Multiple
740 Human, and Same Human, corresponding to images containing one person, 2–3 different individ-
741 uals, and two identical individuals, respectively. For each category, we construct dedicated human
742 recognition datasets and evaluate the model’s recognition performance under each data setting. Ta-
743 ble 12 presents the model’s recognition performance under different types of human recognition
744 data. The experimental results demonstrate that the inclusion of each type of human recognition
745 data leads to improved model recognition performance.

In addition, we conduct another analysis to evaluate the impact of removing random backgrounds by replacing all the human recognition data with a plain white background. As presented in Table 12, removing the random background leads to a notable performance drop.

| Single Human | Multiple Human | Same Human | Random Background | ACC \uparrow | ACC Drop |
|--------------|----------------|------------|-------------------|----------------|--------------------|
| ✗ | ✗ | ✗ | ✗ | 62.0 | 22.9% \downarrow |
| ✓ | ✗ | ✗ | ✓ | 72.3 | 10.4% \downarrow |
| ✓ | ✓ | ✗ | ✓ | 76.7 | 4.60% \downarrow |
| ✓ | ✓ | ✓ | ✗ | 67.4 | 16.2% \downarrow |
| ✓ | ✓ | ✓ | ✓ | 80.7 | - |

Table 12: Comparison of different human recognition data.

748

749 G Real-world Deployment

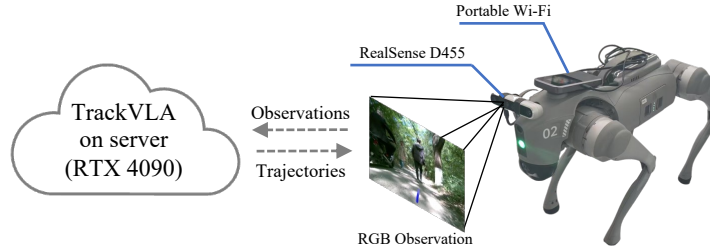


Figure 9: **Real-world system architecture.** TrackVLA is deployed on a remote server, and the robot communicates with it via the Internet.

750 G.1 Robot Platform

We provide a visualization of our robotic platform in Fig. 9. The platform is based on the Unitree GO2 quadruped robot, equipped with an Intel RealSense D455 camera. In our work, only RGB frames with a resolution of 640×480 are utilized, under a horizontal field of view (HFOV) of 90° . Additionally, a portable Wi-Fi is mounted on the back of the robot to enable communication with the remote server through the Internet.

756 G.2 Real-world System Architecture

TrackVLA is deployed on a remote server equipped with an NVIDIA RTX 4090 GPU. During tracking, the server receives the instructions and images captured by the Intel RealSense D455 camera via the Internet. To ensure efficient communication, the images are compressed before transmission. After processing the incoming images, the model performs inference and predicts the future trajectory, which is then transmitted to the quadruped robot for execution. Upon receiving the predicted trajectory, the robot employs a pure pursuit algorithm, combined with its pose information, to perform closed-loop control of its linear and angular velocities, enabling it to follow the trajectory accurately. Additionally, the robot leverages LiDAR point cloud data and implements an elastic band algorithm to achieve obstacle avoidance.

766 H Real-world Experiments

To further evaluate the tracking capability of TrackVLA, we conducted extensive real-world experiments comparing a quadruped robot powered by TrackVLA with a leading commercial tracking

769 drone (DJI Flip). We tested the following three levels of tracking scenarios with increasing difficulty,
770 each repeated 10 times:

- 771 • *Easy*: tracking in open outdoor environments without obstacles;
- 772 • *Medium*: tracking in complex environments with occlusions such as walls;
- 773 • *Hard*: tracking a target moving at high speed.

774 The results are shown in Table 13. Both TrackVLA and DJI Flip achieved a 100% success rate in
775 the *Easy* setting. However, as task difficulty increased, the performance of DJI Flip dropped signifi-
776 cantly, falling well below that of TrackVLA. Figure 10 further illustrates several representative cases
777 where TrackVLA succeeded while DJI Flip failed. Additional details of the real-world experiments
778 are provided in the supplementary video.

| Method | Easy | Medium | Hard |
|----------|-------------|------------|------------|
| DJI Flip | 100% | 70% | 50% |
| TrackVLA | 100% | 90% | 70% |

Table 13: **Real-world tracking experiments.** We compare TrackVLA with the commercial tracking drone.

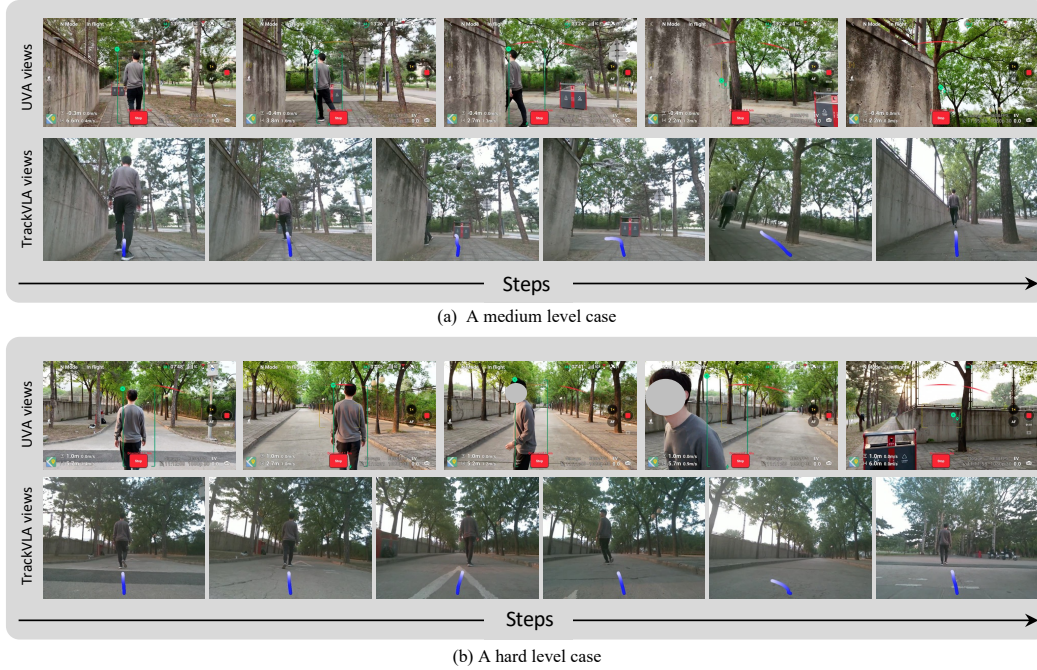


Figure 10: **Visualization of the real-world experiments.** TrackVLA demonstrates robust tracking performance under challenging conditions such as occlusions and fast target motion, outperforming existing commercial tracking drones.

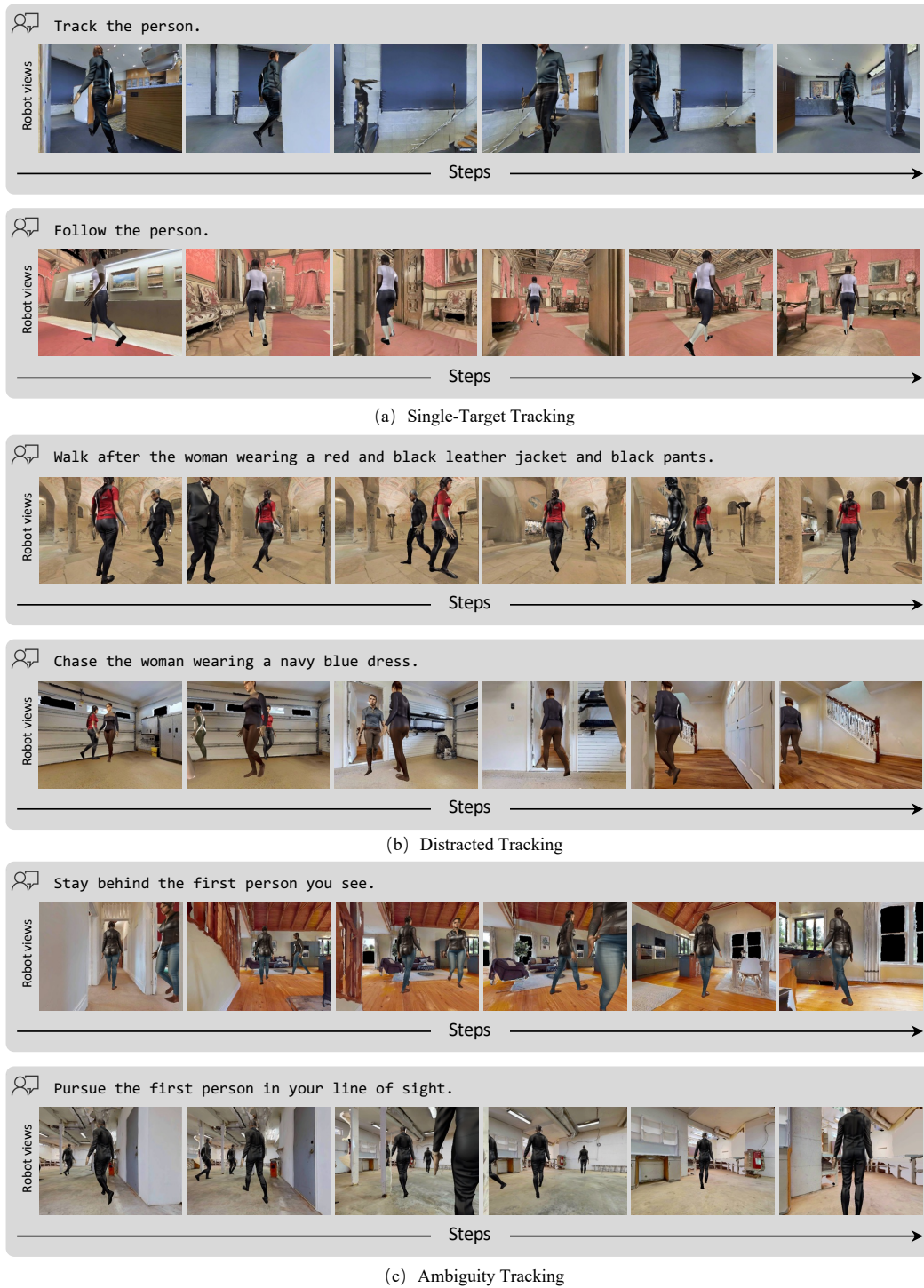


Figure 11: Visualization of the training set of EVT-Bench.

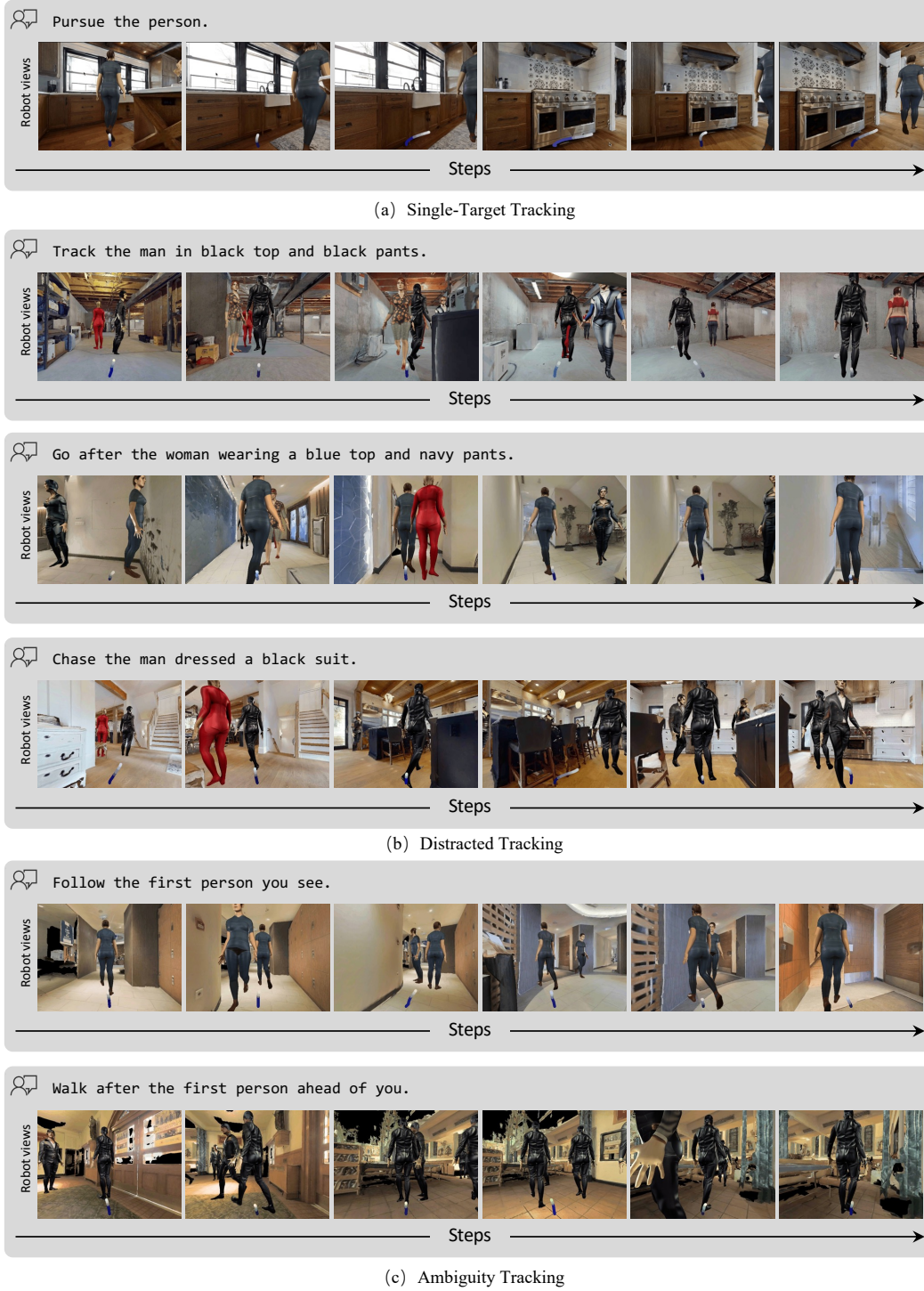
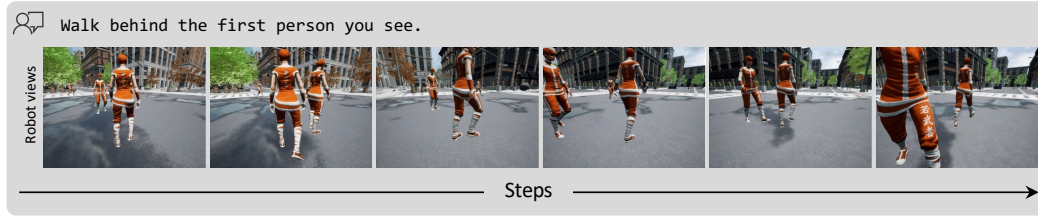
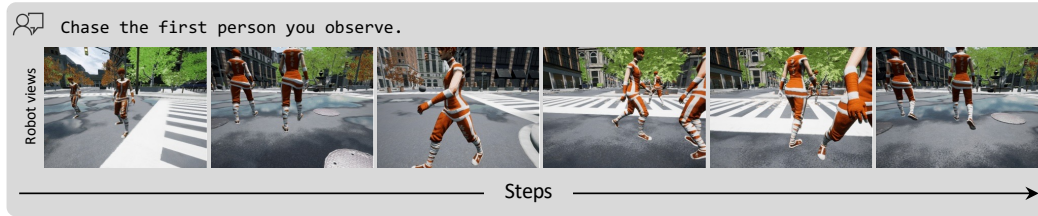


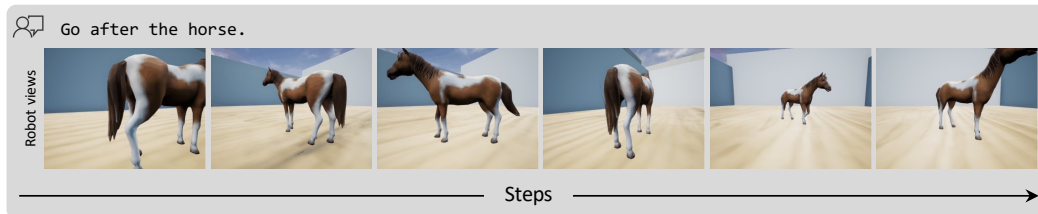
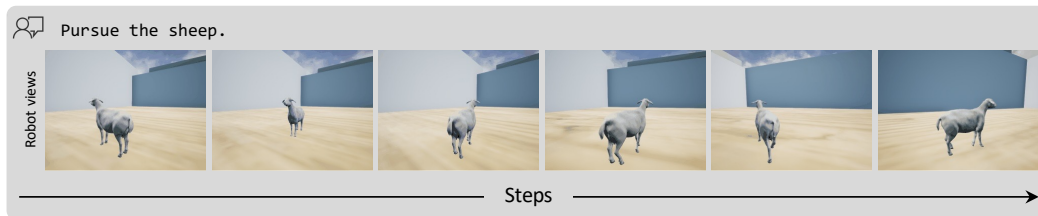
Figure 12: Visualization of TrackVLA on EVT-Bench.



(a) Single Target



(b) Distractor



(c) Unseen Objects

Figure 13: Visualization of TrackVLA on Gym-UnrealCV.