AURA : Multimodal Shared Autonomy for Real-World Urban Navigation

Yukai Ma[1,2] Honglin He[1] Selina Song[1] Wayne Wu[1] Bolei Zhou[1]
[1] University of California, Los Angeles [2] Zhejiang University
https://vail-ucla.github.io/aura/

Abstract

Long-horizon navigation in complex urban environments relies heavily on continuous human operation, which leads to fatigue, reduced efficiency, and safety concerns. Shared autonomy, where a Vision-Language AI agent and a human operator collaborate on maneuvering the mobile machine, presents a promising solution to address these issues. However, existing shared autonomy methods often require humans and AI to operate within the same action space, leading to high cognitive overhead. We present Assistive Urban Robot Autonomy (AURA), a new multi-modal framework that decomposes urban navigation into high-level human instruction and low-level AI control. AURA incorporates a Spatial-Aware Instruction Encoder to align various human instructions with visual and spatial context. To facilitate training, we construct MM-CoS, a large-scale dataset comprising teleoperation and vision-language descriptions. Experiments in simulation and the real world demonstrate that AURA effectively follows human instructions, reduces manual operation effort, and improves navigation stability, while enabling online adaptation. Moreover, under similar takeover conditions, our shared autonomy framework reduces the frequency of takeovers by more than 44%. Demo video and more detail are provided in the project page.

1. Introduction

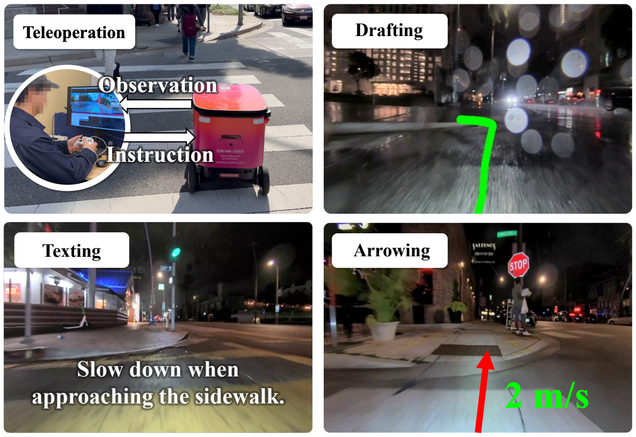


Figure 1. **Shared Autonomy for Urban Navigation.** We introduce AURA, a dual-system VLA for shared autonomy in urban navigation. AURA not only follows instructions but also enables human users to guide and correct a robot in real time through various visual and language instructions.

Despite rapid progress in autonomous driving on roads, AI remains significantly challenging to safely operate mobile machines in public urban spaces, such as sidewalks, parks, and school campuses, due to the complexity of the surroundings and the diverse range of human activities. Thus, many existing sidewalk mobile machines (e.g., assistive wheelchairs and food delivery bots) still rely heavily on a human-in-the-loop approach. For example, remote workers may teleoperate or closely supervise delivery bots to accomplish last-mile tasks. In these settings, the human pilot must remain fully attentive to both control and situational awareness. Road hazards, such as rugged terrain and fragmented curbs, as well as unexpected pedestrian behavior and other human-made errors, pose significant risks to mobile machines and their human pilots.

To address these challenges, recent studies have explored the paradigm of shared autonomy, where AI agents assist and augment human pilots in maneuvering machines in both training [11, 12, 46] and test-time phases [60]. For long-horizon navigation tasks, researchers are increasingly seeking AI assistant models that can effectively reduce human workload, allowing humans to remain in the loop primarily for monitoring and failure recovery [22, 35, 51, 55].

Deploying navigation autonomy models in the real world for long-horizon navigation typically involves switching between full human operation and full AI control [11, 12, 22, 35, 46, 55]. However, these approaches assume that the human and AI operate in the same low-level action space that directly controls the driving wheels and pedals, thereby requiring the human to operate at the same frequency as the AI. This coupling becomes inefficient and cognitively demanding for long-horizon tasks such as urban sidewalk food delivery. We believe a more effective paradigm is a *division of labor* between humans and AI by decomposing urban navigation into hierarchical levels: (i) humans provide high-level instructions, such as reasoning about corner cases and proposing alternative routes, and (ii) AI agents make low-level executions, such as lane keeping and obstacle avoidance. By delegating low-level control to the AI while reserving high-level strategic decisions for the human, this human-AI shared autonomy framework will achieve improved efficiency, safety, and robustness in long-horizon urban navigation.

In this work, we propose **A**ssistive **U**rban **R**obot **A**utonomy (**AURA**), a multi-modal shared autonomy framework designed to understand human instructions and handle low-level control, thereby significantly reducing human operation costs. Specifically, AURA is a Vision-Language-Action (VLA) model with a dual-system architecture. Acting as an "autonomous assistant," AURA can be seamlessly integrated into existing delivery robots without any hardware modifications. For high-level instruction, the system enables various human instructions through a vision-and-language interface, where concise natural-language or visual prompts on live video streams replace laborious joystick control. It enables safe, scalable sidewalk autonomy while substantially reducing operator workload. As illustrated in the center of Fig. 1, AURA can interpret multiple forms of human instructions: (i) texting, where users describe the intent; (ii) drafting, where users draw a rough path on the observation view; and (iii) arrowing, where users demonstrate the desired speed and direction.

A key challenge in shared autonomy is interpreting ambiguous human instructions and grounding them within the surrounding spatial context. To address this, we introduce a Spatial-Aware Instruction Encoder (SIE) that explicitly aligns textual instructions with both the semantic layout and the geometric structure of the scene. This design addresses the spatial-understanding limitations of standard vision–language models (VLMs), enabling the model to reason about the user's intent and where those instructions can be executed, thereby improving robustness across diverse real-world environments.

To support this model training, we construct a multi-modal video dataset, MM-CoS, on top of our sidewalk teleoperation dataset CoS [19] that includes a total of 50 hours of high-quality teleoperation data collected on real-world sidewalks. The dataset spans a diverse range of scenarios, including different cities, weather conditions, and lighting variations. We further curate relevant clips using behavior- and ego-state-based filters to ensure that the resulting human control behaviors are representative. By combining human instructions inferred from VLMs with ground-truth trajectories from human operators, we obtain a unified dataset in which multimodal human instructions serve as inputs and expert trajectories serve as outputs.

Experiments show that our framework accurately interprets and executes human instructions, achieving over 15% lower L2 error and reducing human operation costs by more than 70% compared to baselines. These results highlight the effectiveness of our shared-autonomy design in enabling efficient and human-aligned navigation. In summary, our contributions are threefold:

- A multi-modal shared autonomy framework that integrates human instruction following with low-level control via a unified VLM encoder and diffusion policy.
- We introduce Spatial-Aware Instruction Encoder designed for instruction understanding, which is trained on a new dataset with multi-modal instructions to connect human teleoperation and VLM-based intention inference.
- Simulated and real-world experiments demonstrate the effectiveness of our approach in following instructions, improving stability, and reducing human operation cost.

2. Related Work

Human-in-the-loop Learning and Shared Autonomy. It is crucial to ensure safety when deploying robots in urban environments, and incorporating human preferences into decision-making offers a promising way toward trustworthy, human-centered autonomy [11, 46]. Prior works have incorporated human feedback during training to improve policy alignment or to correct undesired behaviors, a paradigm often referred to as learning with human involvement or human-in-the-loop learning [1, 11, 45, 46, 62, 64, 67]. In this paradigm, humans actively intervene or provide demonstrations when the robot exhibits unsafe or suboptimal behaviors, allowing the policy to learn corrective actions and refine itself through shared autonomy. During deployment, it is equally important for the robot to understand implicit and explicit human intentions and preferences, expressed through multimodal instructions such as language descriptions, visual cues, etc., as demonstrated in recent advances [6, 15, 25, 28, 37, 53, 62, 64]. However, language-based guidance is typically limited to high-level task specifications and cannot support safety-critical, high-frequency interactions required for applications such as urban navigation. In these scenarios, the policy must be able to rapidly adjust its behavior in response to human instructions such as visual paths or corrective joystick inputs, enabling fine-grained shared autonomy that complements language.

Urban Navigation. Navigation is an important problem in robot learning, requiring autonomous agents to perceive, plan, and act safely in complex real-world environments. It poses challenges for understanding human behavior and ensuring safety and compliance in densely populated urban spaces. Early approaches focused primarily on map-based navigation [16, 41, 57], relying on accurate maps and localization techniques such as SLAM [42] to estimate position and plan collision-free paths. However, these methods often assume static environments and struggle to adapt to dynamic, socially interactive settings like urban sidewalks.

Recent works [13, 18, 40] based on reinforcement learning (RL) [33, 56] have demonstrated promising results in mapless navigation by eliminating the dependency on maps. However, these methods often have limited generalizability, particularly in visual navigation [47, 52, 59], due to the simulation-to-real gap [44, 58, 65]. Inspired by the success of scaling laws in language modeling [2, 27], many recent works have proposed various vision-based navigation foundation models [9, 20, 35, 50, 51, 55], leveraging massive video data for improved generalizability across different robot platforms and camera configurations.

Robotic Instruction Following. Advances in large language models (LLMs) [2, 27] and instruction alignment [43] suggest that neural networks can align with human preferences. Inspired by this, embodied AI asks whether similar principles let robots follow instructions and generalize across tasks. In perception, vision-language models (VLMs) [5, 7, 38, 39] show strong multimodal reasoning, motivating VLA policies that integrate perception, language, and low-level control [3, 10, 14, 24, 30], enabling instruction following across platforms [10, 21, 24].

However, language instructions alone encode high-level intent and are insufficient for the fine-grained, high-frequency corrections required in dynamic navigation. The bottleneck arises from how instruction alignment is performed. Following InstructGPT and RLHF [43], most VLA models rely on offline datasets. Such alignment contrasts with navigation, where safety and robustness often depend on *real-time* human feedback. While recent systems explore interactive manipulation with visual prompting [26, 63], these approaches primarily target short-horizon interactions and discrete action spaces, making them fundamentally different from the continuous, long-horizon trajectory reasoning required by navigation. In this work, we propose a novel shared-autonomy framework that addresses these limitations by integrating high-level human intent with real-time low-level control during inference.

3. AURA Framework

We present AURA, an end-to-end shared autonomy framework for long-horizon navigation. AURA takes diverse human instructions as input and predicts future waypoints to control mobile machines. Figure 2 provides an overview of the framework and its key components. As follows, we first formulate the shared autonomy problem in urban navigation and specify the input and output representations. Then, we detail the model architecture in Section 3.1 and introduce the key component, the SIE in Section 3.2. Next, we describe the dataset and annotation process to support training in Section 3.3. Finally, we give the model training strategies in Section 3.4.

Problem Formulation. We aim to design and train a shared-autonomy system for mapless goal-directed visual navigation that supports various types of human instructions across diverse urban environments. The system relies solely on egocentric RGB images to perceive its surroundings. AURA provides two navigation modes: (i) `Autopilot`: AURA takes sparse GPS waypoints as input and supports basic capabilities such as sidewalk following and obstacle avoidance. ii) `Takeover`: When GPS signals are unreliable, goal locations are ambiguous, or autopilot encounters corner cases it cannot handle, a human can intervene by providing guidance via texting instructions, drafting future paths, or arrowing demonstrations to prompt AURA. This design eliminates the need for pre-built maps or explicit localization modules and frames navigation as a sequential decision-making problem.

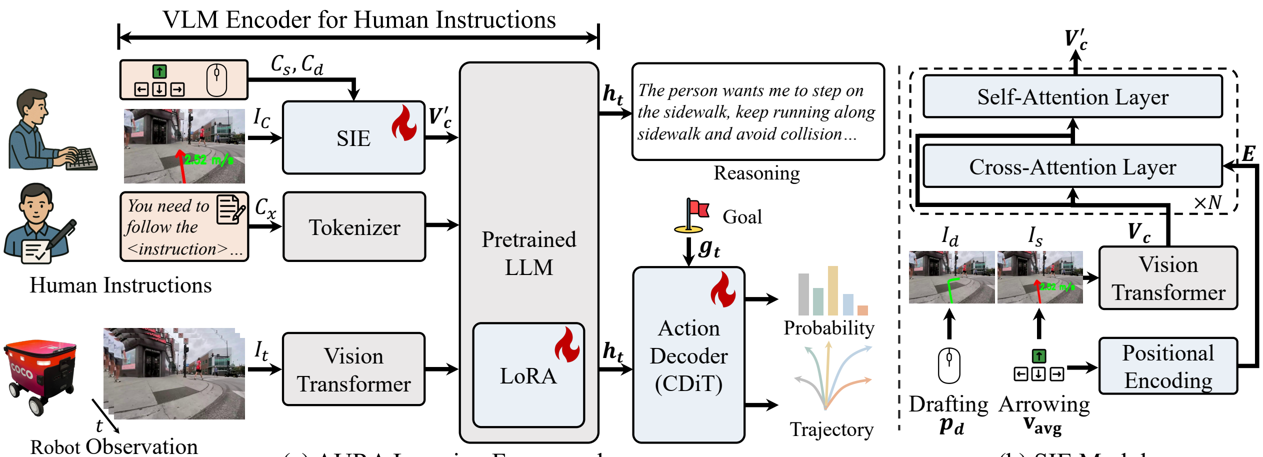
At each timestep t, the agent receives a history of the past 3 frames of RGB observations $\boldsymbol{I}_t$. In Autopilot, a sub-goal or route $\boldsymbol{g}_t$ in egocentric coordinates is also provided. In Takeover, the sub-goal is replaced by a human instruction $C_{t'}$, provided at time t', where the instruction type $c \in \{C_x, C_d, C_s\}$, with $t - t' \leq t^*$, and t^* denotes the maximum duration for which a human instruction can provide guidance. The agent $\mathcal{M}_\theta$ takes these inputs and outputs the action $\boldsymbol{a}_t$ to control the robot.

3.1. Model Architecture

We describe the dual-system architecture of AURA, as illustrated in Figure 2. AURA comprises two main components: a multi-modal encoder that encodes observations and multi-modal instructions, and a diffusion-based policy executor. Specifically, AURA leverages anchor-based regression and classification to learn future trajectory generation. A diffusion transformer (DiT) encodes the robot's sensor configuration and target point, which are then cross-attended with high-level instruction features from the VLM backbone to produce denoised motor actions. We provide a detailed description of each module below.

DiT Action Decoder. AURA first generates robot actions in a purely autonomous setting using a diffusion-based policy. Given input context features, the DiT decoder produces multi-modal future trajectories conditioned on the robot's observations and navigation goal. Instead of starting from Gaussian noise, we initialize the diffusion process from m=64 trajectory anchors representing motion primitives (e.g., straight, turn, stop) clustered from the *MM-CoS* dataset. Building on our prior work, MIMIC [19], a lightweight transformer decoder conditions the denoising process on context features $\boldsymbol{h}_t$, navigation goal $\boldsymbol{g}_t$, and diffusion-timestep embeddings t_d, producing refined trajectories along with their confidence scores.

VLM for Human Instructions. To incorporate human guidance, we augment the base policy with high-level in-



(a) AURA Learning Framework (b) SIE Module

Figure 2. **Overview of the AURA shared autonomy framework.** (a) AURA takes front-camera RGB observations and optional human guidance (e.g., texting, drafting, or arrowing). Observations are encoded by a ViT, while human inputs are processed by the SIE and tokenized; all tokens are fused in a pretrained LLM (with LoRA adapters) to produce context features. A diffusion-based action decoder then predicts a distribution over future trajectories via anchor proposals. (b) The SIE converts drafting/arrowing inputs into instruction tokens: it renders the human input as visual prompts, encodes the control points/vectors, and fuses them with visual features to produce instruction embeddings that are injected into the LLM via ⟨instruction⟩.

structions using the InternVL3-2B [68] VLM backbone. Past and current RGB images are resized to 448 × 448, processed by the visual encoder, and projected via an MLP to produce 256 image token embeddings per frame. We retain a special ⟨image⟩ token to represent each image context in the textual input.

Human instructions are injected via an additional ⟨instruction⟩ token, whose embeddings are produced by our SIE (see Section 3.2). The resulting *vision-language-instruction* embeddings $\boldsymbol{h}_t$ combine both image and instruction features. During inference and policy training, we extract intermediate representations from the 12th layer, balancing inference speed and representational quality. In addition to providing features for control, we attach a lightweight text head to decode interpretable reasoning traces for language supervision.

Finally, these embeddings are cross-attended by the DiT action decoder, conditioning continuous trajectory generation on both robot observations and human instructions, enabling safe and flexible shared autonomy.

3.2. Spatial-Aware Instruction Encoder

SIE is a key component of the VLM encoder (Figure 2(b)), designed to embed the spatial information in human instructions. For the Takeover mode, human instructions $C \in \{C_x, C_d, C_s\}$ provide explicit guidance through visual overlays. This requires the model to infer the human's instruction in an immediate prompt and generate sequential actions over a period of time. To do so, the model must understand both the semantic and geometric information in the human instruction. Fortunately, ViT already possesses strong semantic understanding, which improves prompts.

Specifically, we render the instructions (e.g., trajectory lines or steering arrows) on the observation and encode the instruction image $I_C \in \{I_d, I_s\}$ with the same vision encoder to obtain instruction visual features $\boldsymbol{V}_c \in \mathbb{R}^{N_v \times d_v}$.

However, VLM is not inherently sensitive to spatial information. To effectively ground the instructions in their geometric context, we introduce modality-specific embeddings. For the drafting trajectory instruction, we sample K pixel coordinates $\boldsymbol{p}_d = \{(u_i, v_i)\}_{i=1}^{K}$ along the projected trajectory line in normalized image space. Following the design of Segment Anything [31], we apply Fourier-based positional encoding with learnable frequency basis:

$$\mathrm{PE}(\boldsymbol{p}_{d,i}) = [\sin(\boldsymbol{w}^\top \boldsymbol{p}_{d,i}), \cos(\boldsymbol{w}^\top \boldsymbol{p}_{d,i})] \qquad (1)$$

where $\boldsymbol{w} \in \mathbb{R}^{2 \times d_p}$ is a learnable Gaussian random matrix. To preserve point ordering along the trajectory, we add learnable index embeddings: $\boldsymbol{E}_d^{(i)} = \mathrm{PE}(\boldsymbol{p}_{d,i}) + \mathrm{PosEmbed}(i)$. All point embeddings are concatenated and processed through MLP and self-attention to obtain the final encoding $\boldsymbol{E}_d \in \mathbb{R}^{d_v}$.

For arrowing instruction $\mathbf{v}_{\text{avg}} = (v, \theta)$ where v is speed and θ is heading angle, we use a rotation-invariant encoding that handles both forward and backward motion:

$$\boldsymbol{E}_s = \mathrm{MLP}([\cos(\theta'), \sin(\theta'), \log(1 + |v|)]) \qquad (2)$$

where $\theta' = \theta + \pi \cdot \mathbb{1}_{v<0}$ adjusts heading for backward motion. This representation is processed through MLP and self-attention for the final embedding $\boldsymbol{E}_s \in \mathbb{R}^{d_v}$.

The geometric embeddings $\boldsymbol{E} \in \{\boldsymbol{E}_d, \boldsymbol{E}_s\}$ are fused with instruction visual features $\boldsymbol{V}_c$ via cross-attention with

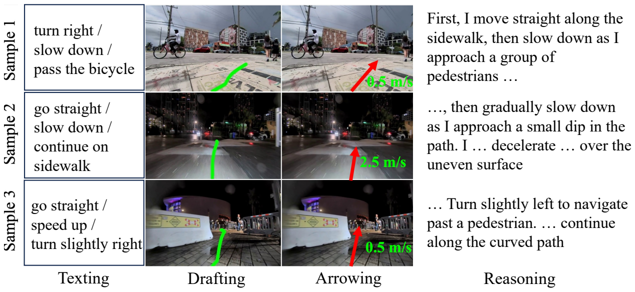

Sample 1
turn right / slow down / pass the bicycle
0.5 m/s
First, I move straight along the sidewalk, then slow down as I approach a group of pedestrians …
Sample 2
go straight / slow down / continue on sidewalk
2.5 m/s
…, then gradually slow down as I approach a small dip in the path. I … decelerate … over the uneven surface
Sample 3
go straight / speed up / turn slightly right
0.5 m/s
… Turn slightly left to navigate past a pedestrian. … continue along the curved path
Texting
Drafting
Arrowing
Reasoning

Figure 3. **Samples from the auto-labeling pipeline.** Each frame is annotated with three training labels produced by our auto-labeling pipeline: (1) the texting command expressed as a short verb phrase (e.g., "go straight", "slow down", "speed up"), (2) the drafting, visualized as a rendered path from the ground-truth future trajectory, and (3) the arrowing input, represented by instantaneous speed. The rightmost panel shows the reasoning traces used to supervise drafting and arrowing prediction.

a residual connection, then refined through a 4-head self-attention layer and a final MLP, yielding instruction-aware features $\boldsymbol{V}'_c$ that are injected into the language model via special $\langle\texttt{instruction}\rangle$ tokens.

3.3. Auto-Labeling for Human Instruction

To enable the training of the AURA framework, we require data that provides high-quality explanations for actions as text instructions C_{text}, as well as precise odometry and camera parameters for generating visual instructions C_d and C_s. The corresponding sequential trajectories serve as outputs in urban navigation scenarios.

However, constructing such a dataset presents two main challenges. (i) **Lack of urban sidewalk data:** Prior datasets are primarily collected in campuses, indoor environments, or plazas, leaving a gap for real-world deployment. To mitigate this, we repurpose the large-scale real-world sidewalk teleoperation logs from our prior sidewalk-autopilot study [19], which already cover diverse urban sidewalks and long-horizon navigation behaviors. (ii) **Annotation quality:** Existing urban datasets often lack high-quality textual explanations that justify actions. We address this by generating action-grounded language explanations for each trajectory using a VLM-based captioning pipeline (Appendix B.2), producing higher-quality and more consistent textual supervision.

Recent advances in VLMs [8, 68] and large-scale urban navigation datasets [4, 22, 29, 48] have made it increasingly feasible to train such models effectively. To further address these limitations, we collect a diverse 50-hour teleoperation dataset comprising 3,040 trajectories captured by a wheeled robot across various real-world urban environments. Finally, we augment our collected data with RECON [48], SCAND [29], and EgoWalk [4] to construct our training dataset, *MM-CoS*.

As illustrated in Figure 3, we provide examples of the multi-modal instructions generated by our labeling pipeline. It follows a two-stage strategy to determine which frames should be annotated. First, a pre-trained VLM (InternVL3-8B [68]) scores video frames based on visual complexity, such as pedestrian interactions, obstacles, or terrain variations, producing an "interestingness" prior. Next, motion statistics (e.g., acceleration and turning rate) are computed within sliding windows and fused with the VLM scores to obtain weighted motion saliency. Frames are then ranked using this combined score, ensuring that those with rich dynamics or meaningful interactions are prioritized. Finally, multimodal human instructions are synthesized from ground-truth trajectories, and Qwen2.5VL-72B [8] is used to generate corresponding textual prompts.

After selecting informative frames, we generate three complementary types of annotations to provide rich supervisory signals for navigation learning. Building such supervision at scale is non-trivial: each training sample must align (i) noisy real-world robot trajectories, (ii) calibrated camera geometry for accurate projection, and (iii) intent descriptions that are consistent with both the scene context and future motion. Importantly, these annotations are designed to mirror the human interfaces in shared autonomy: they let users intervene at different abstraction levels (sketching a route, nudging velocity, or stating intent) without requiring continuous low-level teleoperation. To balance clarity in the main paper and reproducibility, we summarize each modality below and defer implementation details (e.g., projection, sampling, prompting) to the Appendix B.3.

CMD `Drafting` A user can quickly *draw a rough path* on the live observation to indicate where the robot should go (e.g., "go around the group" or "take the right side"), which is often easier than issuing continuous joystick commands. We therefore represent this interface as a trajectory overlay with sampled pixel points, providing explicit spatial guidance for the policy.

CMD `Arrowing` When only a brief correction is needed (e.g., slow down, gently turn left), a low-bandwidth arrowing signal is a natural and lightweight intervention. We encode this interface with compact speed and heading supervision (and its visualization), enabling the model to react to short corrective nudges while still handling low-level stabilization.

CMD `Texting` Natural language is convenient for expressing high-level intent and constraints (e.g., "yield to pedestrians" or "stay close to the curb"), especially under limited attention or communication latency. We use a vision-language model (Qwen2.5-VL-72B [8]) to generate a command-style instruction plus a longer description, training the policy to align execution with human intent and scene interactions.

3.4. Model Training

We employ a two-stage training strategy to efficiently learn the share-autonomy model.

Instruction-conditioned VLM Adaptation. In the first stage, we adapt the InternVL3-2B [68] to incorporate human instruction conditioning. We freeze the vision encoder and the original vision-to-language projection MLP, and only train the newly introduced SIE modules. Additionally, we employ LoRA [23] for efficient adaptation of the language model. The model is trained using a language modeling loss on the generated trajectory captions, enabling the VLM to understand and encode semantic-spatial instruction signals through natural language grounding.

End-to-End Diffusion Policy Learning. In the second stage, we train the diffusion-based policy network while keeping the multi-modal encoder frozen. The diffusion decoder and auxiliary encoders (goal, camera, trajectory anchor encoders) are trained from scratch. The training loss combines mode classification loss $\mathcal{L}_{cls}$ and trajectory regression loss $\mathcal{L}_{reg}$: $\mathcal{L} = \mathcal{L}_{cls} + \mathcal{L}_{reg}$, where $\mathcal{L}_{cls}$ selects the mode closest to GT via cross-entropy, and $\mathcal{L}_{reg}$ minimizes L2 distance between the predicted and GT trajectories.

4. Experiments

We evaluate AURA from three aspects: (i) its ability to follow human instructions (Section 4.1), (ii) the efficiency of shared control (Section 4.2) and (iii) the ablation study to evaluate the effectiveness of components in the framework and the dataset (Section 4.3). Finally, we conduct a pilot study that deploys the system in real-world sidewalk environments (Section 4.4). Additional qualitative results are provided in Appendix A.2.

4.1. Validation on Instruction Following

We first evaluate our approach on instruction-following in an open-loop manner, where predicted trajectories are compared against ground-truth future trajectories from the *MM-CoS* test set. For evaluation, we adopt the standard open-loop metrics proposed in prior works [17, 61]. During testing, we provide our model with different input modalities, including texting, drafting, arrowing, and point. Motivated by the limited exploration of human instruction in urban navigation, we also adopt point-goal [22, 36] and image-goal navigation models [49, 51, 55] as baselines for comparisons, since they provide more precise guidance compared to instruction-based approaches.

Table 1 shows that AURA consistently outperforms all baselines across instruction modalities. In particular, the arrowing-guided variant 🎮 achieves the lowest L2 error (0.150 at 1s and 0.473 at 2s), outperforming CityWalker* (0.353 at 1s and 0.786 at 2s) by 39.8% at 2s. The drafting-guided version ✏️ yields the best mAP (0.844) while maintaining strong L2 performance (0.557 at 2s), whereas the text-guided version 📝 is slightly worse in both mAP and L2. Overall, geometric instructions (drafting and arrowing) provide stronger spatial guidance for precise trajectory generation than purely linguistic instructions. Some baselines omit mAP due to deterministic single-trajectory outputs, and omit L2$_{2s}$ due to shorter prediction horizons.

Table 1. **Open-loop evaluation on MM-CoS.** * denotes models re-trained on our dataset.

	minADE$_{1s}$ ↓	minFDE$_{1s}$ ↓	mAP ↑	L2$_{1s}$ ↓	L2$_{2s}$ ↓
GNM‡ [49]	0.594	0.988	-	0.988	-
ViNT‡ [51]	0.638	1.056	-	1.056	-
NoMaD‡ [55]	0.523	0.858	0.216	1.072	2.182
MBRA [22]	0.617	1.019	-	1.019	2.034
CityWalker [35]	0.648	1.125	-	1.125	-
ViNT* [51]	0.247	0.450	-	0.425	0.925
CityWalker* [35]	0.180	0.353	-	0.353	0.786
AURA 📍	0.125	0.218	0.699	0.266	0.670
AURA 📝	0.125	0.238	0.683	0.259	0.673
AURA 🎮	0.108	0.220	0.750	0.150	0.473
AURA ✏️	0.122	0.218	0.844	0.244	0.557

We provide qualitative results in Figure 4, illustrating how AURA responds to three distinct categories of human instructions. Under drafting or arrowing input as instruction, the model produces the geometrically aligned predictions in different scenarios, closely matching the user intention. With texting instructions, AURA converts high-level semantic intent into a coherent motion plan that respects both the instruction behavior and the surrounding scene, *i.e.*, following the sidewalk direction. These examples demonstrate the model's ability to interpret diverse human input and produce safe, intent-aligned trajectories for urban navigation.

4.2. Shared Control Efficiency Analysis

We aim to evaluate the efficiency of different shared-control methods. Specifically, we examine how robust each agent is to noisy waypoints and quantify the amount of time an operator needs to spend on takeover.

Pseudo-simulation testing for shared control. Closed-loop evaluation is feasible in simulation or the real world but hard to scale because human takeovers add randomness, so we use a pseudo-simulation shared-control testing pipeline. The pipeline includes (i) a navigation model that takes sequential video frames and noisy target points (random angle in $[-90°, 90°]$ and distance in $[0, 10]\,\mathrm{m}$) to predict trajectories and (ii) a judgment module that decides per frame whether takeover is needed and records intervention frames (Appendix C.1).

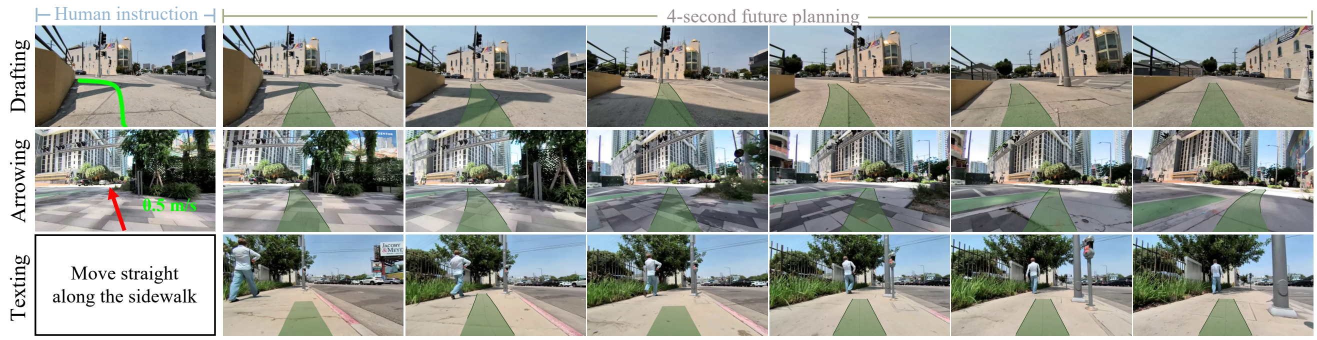


Figure 4. **Visualization of offline inference in *MM-CoS*.** We illustrate three types of human instructions. The green polygon denotes the future trajectories predicted by AURA.

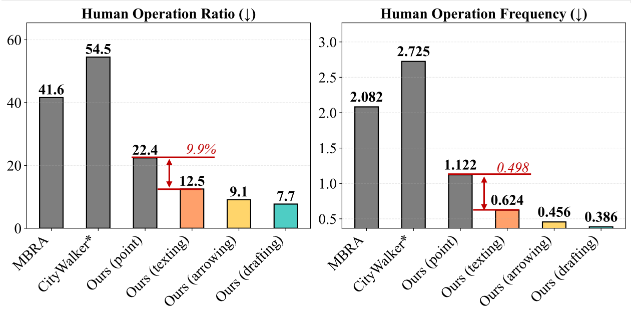


Figure 5. **Human Cost Evaluation in Pseudo-simulation.** We compare the human intervention cost of our model against prior methods [35, 40], as well as across different modes of instruction guidance within our own framework.

Results. We report two key metrics: Human Operation Ratio (percentage of time under human control) and Human Operation Frequency (number of frames under human control per total duration). We report two key metrics: Human Operation Ratio (percentage of time under human control) and Human Operation Frequency (frames under human control per total duration). Results are shown in Figure 5, assuming each human takeover lasts 2 seconds. Two observations follow. First, our method achieves a human operation frequency of 0.96 and a 19.2% lower Human Operation Ratio compared to the baseline [46], indicating improved robustness in point navigation. Second, when humans provide only high-level instructions, they require fewer control interventions, yielding a 9.9% reduction in human operation time and a human operation frequency of 0.498 (44% reduction). During testing, we use a 5Hz control rate, so the maximum human operation frequency is 5. More results for other takeover durations are in the Appendix C.2.

4.3. Ablation Study

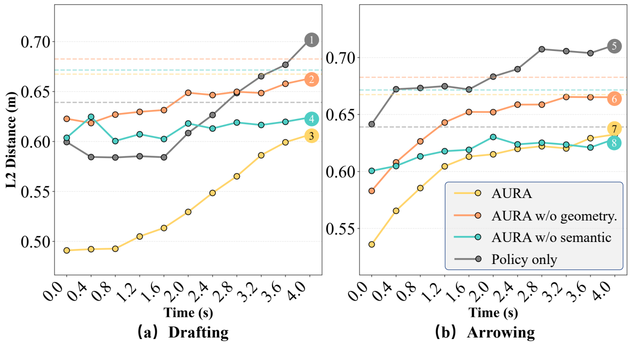
To valid the dataset and the SIE we designed. To validate the effectiveness of the collected dataset and the proposed Instruction Encoder, we conduct ablation studies on both instruction understanding and planning capabilities.

Table 2. **Evaluation on instruction following.**

Model	Finetune	Visual Prompt	SIE	ROUGE-L ↑	Intent Score ↑
InternVL3-2B	×	✓	×	0.167	2.019
InternVL3-8B	×	✓	×	0.184	2.818
InternVL3-2B	✓	✓	×	0.532	4.885
AURA	✓	×	✓	0.534	4.842
AURA	✓	✓	✓	**0.581**	**5.446**

Instruction following. We evaluate the model's ability to reason correctly based on human instructions. Specifically, we report ROUGE-L [34] and Qwen Score. The Qwen Score is computed using QwenVL2.5-72B [8], a vision-language evaluation model. These metrics measure the quality of the model's reasoning outputs compared to human-provided references. The results are presented in Table 2. After fine-tuning on our synthesis label dataset, the InternVL3-2B model demonstrates significantly improved instruction following compared to its non-fine-tuned counterpart, and even surpasses the larger InternVL3-8B model in both accuracy and fluency. Using only the proposed projector module, the model achieves comparable performance to the original VLM with visual prompting. When combining both visual prompts and the projector, the model attains the best overall performance in instruction following.

End-to-end planning. To evaluate the effects of semantic and spatial awareness in our SIE, we conduct trajectory planning experiments under human-provided drafting and arrowing inputs. The horizontal axis denotes the time lag between each instruction and the current frame, with larger values indicating older instructions. We compare four models: the baseline with only the DiT action decoder, where observations and the instruction image are encoded with Dinov3 [54] using visual prompts; AURA without geometry, which uses only visual prompts; AURA without without semantic, which directly embeds the drafting and arrowing inputs without any visual prompts; and AURA with both



(a) Drafting (b) Arrowing

Figure 6. **Effectiveness of components on end-to-end planning.**

Table 3. **Real-world closed-loop navigation results.** We report Human Operation Ratio (HO), Normalized Intervention Rate (NIR, emergency interventions per 100 meters), Off-track Distance Ratio (ODR), and Time Success Rate (TSR, ratio of valid autonomous time to total autonomous time). Lower is better for HO/NIR/ODR and higher is better for TSR.

Method	HO (%) ↓	NIR ↓	ODR ↓	TSR ↑
NoMaD	9.74	43.2	11.3	89.0
CityWalker	14.56	48.29	20.0	80.3
Gemini	16.9	255.7	32.0	63.2
AURA	**1.73**	**16.99**	**10.5**	**89.3**

semantic and geometry.

Figure 6 visualizes the L2 distance at 2s for these models. Dashed lines indicate the corresponding results without any target input. Line ❶ and ❸ demonstrates the importance of semantic understanding: lacking a shared semantic representation, the model fails to interpret human instruction and performs worst when predicting instructions 4s ahead, even worse than the model without any target input. Comparing lines ❷ and ❹, and lines ❻ and ❽, we observe that geometry encoding provides similar short-term performance as visual prompts, since both convey goal direction information. However, over longer horizons, geometry encoding establishes a stable spatial structure and goal memory, whereas the explicit information in visual prompts disappears as the field of view changes, leading to performance degradation. Combining both semantic and geometric guidance achieves the best overall performance (lines ❸ and ❼), as visual prompts ensure accurate short-term tracking while geometry encoding maintains long-term spatial consistency and goal-directed memory.

4.4. Real-World Pilot Study

We conduct a pilot study by deploying the proposed shared-autonomy system on a wheeled robot in real-world sidewalk environments. We evaluate 8 scenarios covering 16 routes with a total length of about 2.8 km.

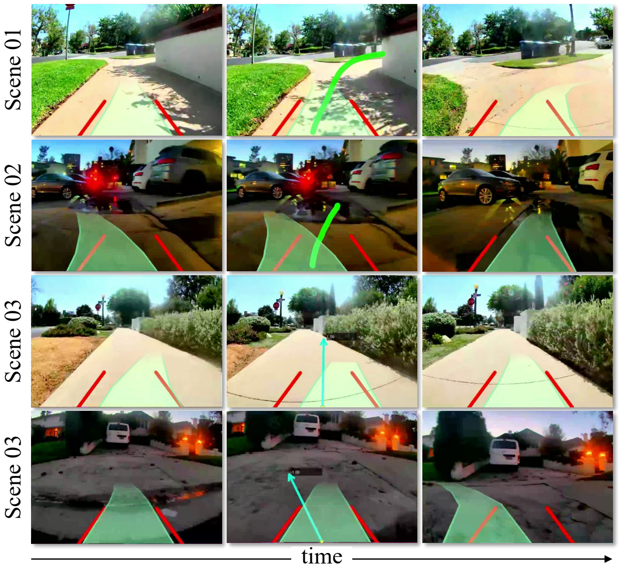


Figure 7. **Visualization of the teleoperation interface in real-world experiments.** For each row, the center frame corresponds to the moment of human intervention (teleoperation takeover), with the other frames showing the surrounding context. The green polygon denotes the predicted trajectories, and the red lines on both sides indicate the width of the robot.

As shown in Table 3, we compare against NoMaD, CityWalker, and Gemini, using four metrics tailored to shared autonomy: Human Operation Ratio (HO), Normalized Intervention Rate (NIR, emergency interventions per 100 meters), Off-track Distance Ratio (ODR), and Time Success Rate (TSR, ratio of valid autonomous time to total autonomous time). With safety-supervised human interventions, all routes can be completed; thus HO/NIR quantify human cost. AURA achieves the lowest human intervention cost across all metrics. Figure 7 presents qualitative examples from the real-world experiments. Additional visualizations are provided in the Appendix A.2.

5. Conclusion

We present an assistive robot autonomy system with hierarchical takeover to reduce effort. It pairs a multimodal encoder that aligns with human instructions with a diffusion policy for planning. Extensive experiments have validated the improved instruction-following capability and the benefits of shared control.

Acknowledgment The project was supported by the NSF Grants CCF-2344955 and IIS-2339769. Honglin is supported by the Amazon Trainium Fellowship. The project and paper were finished during Yukai Ma's visit to UCLA.

References

[1] David Abel, John Salvatier, Andreas Stuhlmüller, and Owain Evans. Agent-agnostic human-in-the-loop reinforcement learning. *arXiv preprint arXiv:1701.04079*, 2017. 2

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 3

[3] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022. 3

[4] Timur Akhtyamov, Mohamad Al Mdfaa, Javier Antonio Ramirez, Sergey Bakulin, German Devchich, Denis Fatykhov, Alexander Mazurov, Kristina Zipa, Malik Mohrat, Pavel Kolesnik, et al. Egowalk: A multimodal dataset for robot navigation in the wild. *arXiv preprint arXiv:2505.21282*, 2025. 5, 12

[5] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022. 3

[6] Reuben M Aronson and Elaine Schaertl Short. Intentional user adaptation to shared control assistance. In *Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, pages 4–12, 2024. 2

[7] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023. 3

[8] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025. 5, 7, 13, 15

[9] Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. *arXiv preprint arXiv:2412.03572*, 2024. 3

[10] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control, 2024. 3

[11] Haoyuan Cai, Zhenghao Peng, and Bolei Zhou. Predictive preference learning from human interventions. *arXiv preprint arXiv:2510.01545*, 2025. 1, 2

[12] Haoyuan Cai, Zhenghao Peng, and Bolei Zhou. Robot-gated interactive imitation learning with adaptive intervention mechanism. *arXiv preprint arXiv:2506.09176*, 2025. 1

[13] Yu Fan Chen, Michael Everett, Miao Liu, and Jonathan P How. Socially aware motion planning with deep reinforcement learning. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1343–1350. IEEE, 2017. 3

[14] An-Chieh Cheng, Yandong Ji, Zhaojing Yang, Zaitian Gongye, Xueyan Zou, Jan Kautz, Erdem Bıyık, Hongxu Yin, Sifei Liu, and Xiaolong Wang. Navila: Legged robot vision-language-action model for navigation. *arXiv preprint arXiv:2412.04453*, 2024. 3

[15] Yuchen Cui, Siddharth Karamcheti, Raj Palleti, Nidhya Shivakumar, Percy Liang, and Dorsa Sadigh. No, to the right: Online language corrections for robotic manipulation via shared autonomy. In *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*, pages 93–101, 2023. 2

[16] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006. 2

[17] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R Qi, Yin Zhou, et al. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *ICCV*, pages 9710–9719, 2021. 6

[18] Tingxiang Fan, Pinxin Long, Wenxi Liu, and Jia Pan. Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios. *The International Journal of Robotics Research*, 39(7):856–892, 2020. 3

[19] Honglin He, Yukai Ma, Brad Squicciarini, Wayne Wu, and Bolei Zhou. Learning sidewalk autopilot from multi-scale imitation with corrective behavior expansion, 2026. 2, 3, 5

[20] Noriaki Hirose, Catherine Glossop, Ajay Sridhar, Dhruv Shah, Oier Mees, and Sergey Levine. Lelan: Learning a language-conditioned navigation policy from in-the-wild videos. *arXiv preprint arXiv:2410.03603*, 2024. 3

[21] Noriaki Hirose, Catherine Glossop, Dhruv Shah, and Sergey Levine. Omnivla: An omni-modal vision-language-action model for robot navigation. *arXiv preprint arXiv:2509.19480*, 2025. 3

[22] Noriaki Hirose, Lydia Ignatova, Kyle Stachowicz, Catherine Glossop, Sergey Levine, and Dhruv Shah. Learning to drive anywhere with model-based reannotation. *arXiv preprint arXiv:2505.05592*, 2025. 1, 5, 6, 16

[23] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022. 6

[24] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025. 3

[25] Zhengran Ji and Boyuan Chen. Pref-guide: Continual policy learning from real-time human feedback via preference-based learning. *arXiv preprint arXiv:2508.07126*, 2025. 2

[26] Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: Robot manipulation with multimodal prompts. 2023. 3

[27] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec

Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020. 3

[28] Siddharth Karamcheti, Raj Palleti, Yuchen Cui, Percy Liang, and Dorsa Sadigh. Shared autonomy for robotic manipulation with language corrections. In *ACL Workshop on Learning with Natural Language Supervision*, 2022. 2

[29] Haresh Karnan, Anirudh Nair, Xuesu Xiao, Garrett Warnell, Sören Pirk, Alexander Toshev, Justin Hart, Joydeep Biswas, and Peter Stone. Socially compliant navigation dataset (scand): A large-scale dataset of demonstrations for social navigation. *IEEE Robotics and Automation Letters*, 7 (4):11807–11814, 2022. 5, 12

[30] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024. 3

[31] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4015–4026, 2023. 4

[32] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023. 13

[33] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015. 3

[34] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004. 7

[35] Xinhao Liu, Jintong Li, Yicheng Jiang, Niranjan Sujay, Zhicheng Yang, Juexiao Zhang, John Abanes, Jing Zhang, and Chen Feng. Citywalker: Learning embodied urban navigation from web-scale videos. *arXiv preprint arXiv:2411.17820*, 2024. 1, 3, 6, 7

[36] Xinhao Liu, Jintong Li, Yicheng Jiang, Niranjan Sujay, Zhicheng Yang, Juexiao Zhang, John Abanes, Jing Zhang, and Chen Feng. Citywalker: Learning embodied urban navigation from web-scale videos. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 6875–6885, 2025. 6, 16

[37] Kai Lu, Chenyang Ma, Chiori Hori, and Diego Romeres. Kitchenvla: Iterative vision-language corrections for robotic execution of human tasks. In *1st Workshop on Safely Leveraging Vision-Language Foundation Models in Robotics: Challenges and Opportunities*. 2

[38] Yukai Ma, Tiantian Wei, Naiting Zhong, Jianbiao Mei, Tao Hu, Licheng Wen, Xuemeng Yang, Botian Shi, and Yong Liu. Leapvad: A leap in autonomous driving via cognitive perception and dual-process thinking. *arXiv preprint arXiv:2501.08168*, 2025. 3

[39] Jianbiao Mei, Yukai Ma, Xuemeng Yang, Licheng Wen, Xinyu Cai, Xin Li, Daocheng Fu, Bo Zhang, Pinlong Cai, Min Dou, et al. Continuously learning, adapting, and improving: A dual-process approach to autonomous driving. *arXiv preprint arXiv:2405.15324*, 2024. 3

[40] Piotr Mirowski, Razvan Pascanu, Fabio Viola, Hubert Soyer, Andrew J Ballard, Andrea Banino, Misha Denil, Ross Goroshin, Laurent Sifre, Koray Kavukcuoglu, et al. Learning to navigate in complex environments. *arXiv preprint arXiv:1611.03673*, 2016. 3, 7

[41] Hans Moravec and Alberto Elfes. High resolution maps from wide angle sonar. In *Proceedings. 1985 IEEE international conference on robotics and automation*, pages 116–121. IEEE, 1985. 2

[42] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: A versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015. 2

[43] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*, 2022. 3

[44] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 3803–3810. IEEE, 2018. 3

[45] Zhenghao Peng, Zhizheng Liu, and Bolei Zhou. Data-efficient learning from human interventions for mobile robots. *arXiv preprint arXiv:2503.04969*, 2025. 2

[46] Zhenghao Mark Peng, Wenjie Mo, Chenda Duan, Quanyi Li, and Bolei Zhou. Learning from active human involvement through proxy value propagation. *Advances in neural information processing systems*, 36:77969–77992, 2023. 1, 2, 7

[47] Pranav Putta, Gunjan Aggarwal, Roozbeh Mottaghi, Dhruv Batra, Naoki Yokoyama, Joanne Truong, and Arjun Majumdar. Embodiment randomization for cross embodiment navigation. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5527–5534. IEEE, 2024. 3

[48] Dhruv Shah, Benjamin Eysenbach, Nicholas Rhinehart, and Sergey Levine. Rapid exploration for open-world navigation with latent goal models. In *5th Annual Conference on Robot Learning*, 2021. 5, 12

[49] Dhruv Shah, Ajay Sridhar, Arjun Bhorkar, Noriaki Hirose, and Sergey Levine. Gnm: A general navigation model to drive any robot. *arXiv preprint:2210.03370*, 2022. 6

[50] Dhruv Shah, Ajay Sridhar, Arjun Bhorkar, Noriaki Hirose, and Sergey Levine. Gnm: A general navigation model to drive any robot. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7226–7233. IEEE, 2023. 3

[51] Dhruv Shah, Ajay Sridhar, Nitish Dashora, Kyle Stachowicz, Kevin Black, Noriaki Hirose, and Sergey Levine. Vint: A foundation model for visual navigation. *arXiv preprint arXiv:2306.14846*, 2023. 1, 3, 6

[52] William B Shen, Danfei Xu, Yuke Zhu, Leonidas J Guibas, Li Fei-Fei, and Silvio Savarese. Situational fusion of visual representation for visual navigation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2881–2890, 2019. 3

[53] Lucy Xiaoyang Shi, Zheyuan Hu, Tony Z Zhao, Archit Sharma, Karl Pertsch, Jianlan Luo, Sergey Levine, and Chelsea Finn. Yell at your robot: Improving on-the-fly from language corrections. *arXiv preprint arXiv:2403.12910*, 2024. 2

[54] Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025. 7

[55] Ajay Sridhar, Dhruv Shah, Catherine Glossop, and Sergey Levine. Nomad: Goal masked diffusion policies for navigation and exploration. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 63–70. IEEE, 2024. 1, 3, 6

[56] Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*. MIT press Cambridge, 1998. 3

[57] Sebastian Thrun. Probabilistic robotics. *Communications of the ACM*, 45(3):52–57, 2002. 2

[58] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017. 3

[59] Joanne Truong, Denis Yarats, Tianyu Li, Franziska Meier, Sonia Chernova, Dhruv Batra, and Akshara Rai. Learning navigation skills for legged robots with learned robot embeddings. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 484–491. IEEE, 2021. 3

[60] Sumukha Udupa, Vineet R Kamat, and Carol C Menassa. Shared autonomy in assistive mobile robots: a review. *Disability and Rehabilitation: Assistive Technology*, 18(6):827–848, 2023. 1

[61] Balakrishnan Varadarajan, Ahmed Hefny, Avikalp Srivastava, Khaled S Refaat, Nigamaa Nayakanti, Andre Cornman, Kan Chen, Bertrand Douillard, Chi Pang Lam, Dragomir Anguelov, et al. Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction. In *2022 ICRA*, pages 7814–7821. IEEE, 2022. 6

[62] Zizhao Wang, Xuesu Xiao, Bo Liu, Garrett Warnell, and Peter Stone. Appli: Adaptive planner parameter learning from interventions. In *2021 IEEE international conference on robotics and automation (ICRA)*, pages 6079–6085. IEEE, 2021. 2

[63] Yinghui Xing, Qirui Wu, De Cheng, Shizhou Zhang, Guoqiang Liang, Peng Wang, and Yanning Zhang. Dual modality prompt tuning for vision-language pre-trained model. *IEEE Transactions on Multimedia*, 26:2056–2068, 2023. 3

[64] Yunkun Xu, Zhenyu Liu, Guifang Duan, Jiangcheng Zhu, Xiaolong Bai, and Jianrong Tan. Look before you leap: Safe model-based reinforcement learning with human intervention. In *Conference on Robot Learning*, pages 332–341. PMLR, 2022. 2

[65] Alan Yu, Ge Yang, Ran Choi, Yajvan Ravan, John Leonard, and Phillip Isola. Learning visual parkour from generated images. In *8th Annual Conference on Robot Learning*, 2024. 3

[66] Hao Zhang, Feng Li, Xueyan Zou, Shilong Liu, Chunyuan Li, Jianwei Yang, and Lei Zhang. A simple framework for open-vocabulary segmentation and detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1020–1031, 2023. 15

[67] Jiakai Zhang and Kyunghyun Cho. Query-efficient imitation learning for end-to-end simulated driving. In *Proceedings of the AAAI conference on artificial intelligence*, 2017. 2

[68] Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, et al. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*, 2025. 4, 5, 6, 12

Appendix

In the appendix, we provide additional details of this work. In Section A, we introduce the robot and teleoperation platform, along with more qualitative results. In Section B, we provide further details about our *MM-CoS* dataset, including data distribution and the automatic labeling pipeline. In Section C, we present additional quantitative results on efficiency under `Share Control`. In Section D, we describe the implementation details.

A. Real-World Experiments

A.1. Evaluation Platform

We evaluate the instruction-following performance of our model on a wheeled robot developed by Coco Robotics. As illustrated in Figure 8, the robot platform includes both the onboard robotic infrastructure and a teleoperation interface for monitoring and control. Notably, during testing, the inference computer is placed inside the robot's storage compartment. For the onboard robotic system, we primarily use the front fisheye camera for perception. The raw fisheye images are undistorted before being passed to the inference computer. The robot is connected to the inference computer through an Ethernet cable, and the inference computer outputs the desired linear and angular velocities generated by a PD controller, which are then sent to the robot executor. The robot performs localization using odometry based on IMU and GPS fusion. Communication with the teleoperation station is established through a 4G router. The teleoperation station consists of a computer equipped with a joystick, mouse, and keyboard. The mouse and keyboard are used for issuing instructions, while the joystick is reserved for full manual control.

A.2. More Visualization Results.

We present additional real-world results in Figure 18, showcasing three cases for each type of instruction. AURA demonstrates robust performance in real-world navigation tasks, including path selection, lane recovery, and obstacle avoidance. The captions under the images describe the robot's actions. The first three rows illustrate that the robot can accurately interpret geo-information under drafting guidance. Rows four to six show that the robot can follow arrowing instructions, effectively performing obstacle avoidance, lane recovery, and path selection. Finally, rows seven to nine demonstrate that our model can follow high-level textual instructions provided by a human.

B. Details about *MM-CoS*.

B.1. *MM-CoS*

Finally, we generate 29K annotations for our *MM-CoS* dataset from 50 hours of teleoperation data. Figure 9 illustrates the data distribution. The frequencies were calculated based on keyword occurrences, and the data are categorized by action mode and interaction behavior. Each major category contains specific subcategories, which may overlap; for example, a single scene could include both "Straight_Movement" and "Turning_Left" at the same time. The percentage values shown on each bar indicate the proportion of the dataset corresponding to that behavior. The weather distribution and time of day are illustrated in Figure 11. The *MM-CoS* dataset (50 hours) contains 3,040 videos, covering a wide range of weather and lighting conditions. Figure 10 illustrates the diversity of the *MM-CoS* dataset, covering variations in lighting, weather, and sidewalk scenes. Such data better reflect real-world scenarios and present significant challenges.

In addition, the *MM-CoS* dataset is further augmented with 18K annotations from open-source datasets [4, 29, 48].

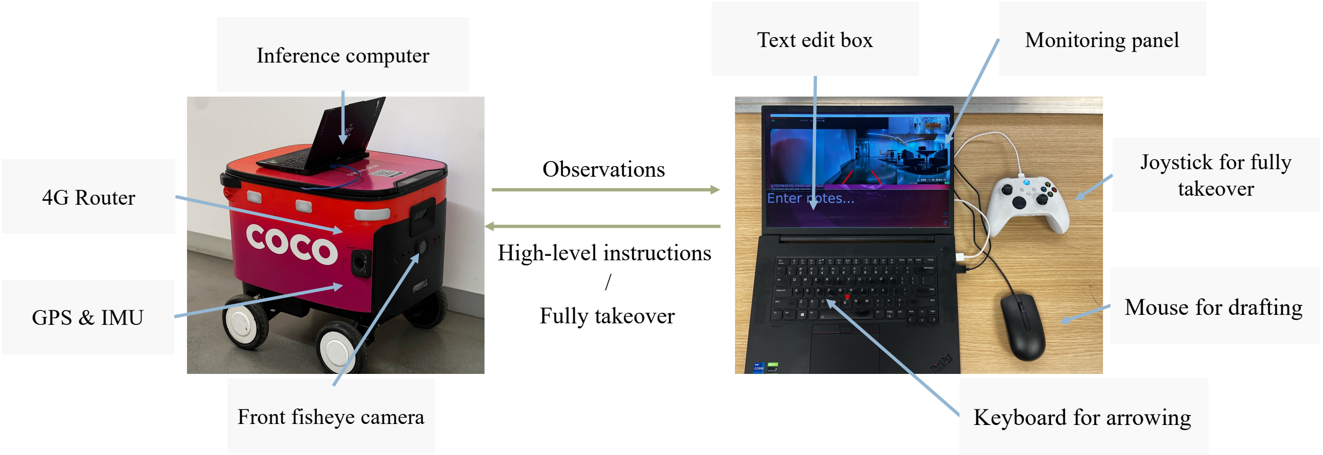
B.2. Auto labeling pipeline

As illustrated in Figure 12, we employ a two-stage selection strategy to identify informative frames for annotation. This approach combines VLM-based assessment with trajectory-based motion analysis, corresponding to the "behavior and state filtering" step shown in the figure.

First, we use a pre-trained VLM (InternVL3-8B [68]) to classify each video segment as either "interesting" or "boring" based on scene complexity, such as pedestrian interactions, obstacles, and terrain changes. This produces a temporal interestingness map $\mathbf{I} = [i_1, i_2, \dots, i_n] \in 1, 2^n$ for the n segments of each clipped trajectory, where $i_j = 1$ denotes an interesting segment and $i_j = 2$ denotes a boring one. The prompts used for this classification are shown in Figure 13.

Second, we perform motion analysis on the robot trajectory using a sliding window approach. For each window, we compute acceleration and turning scores as weighted combinations of motion statistics: $S_{accel} = w_1\alpha_{avg} + w_2\alpha_{max} + w_3\sigma_\alpha^2$ and $S_{turn} = w_4\theta_{avg} + w_5\theta_{max} + w_6\sigma_\theta^2$, where α and θ denote acceleration and turning angle respectively. To prioritize semantically meaningful scenes, we apply priority-based weighting: $S' = w_p \cdot S$ where $w_p = w_{\text{int}} > 1$ for interesting segments, $w_p = w_{\text{bor}} < 1$ for boring segments, and $w_p = 1$ otherwise. We then rank windows by their weighted scores and select the top-k frames with high acceleration or turning, ensuring temporal diversity by filtering adjacent candidates.

Once keyframes are identified (Figure 12), we construct the VLM inputs by overlaying the robot's future trajectory and frame-wise speed cues on the front-view images. With the prompts in Figures 14 and 15, the VLM generates (i) a short command-style instruction and (ii) a detailed description of the underlying reasons. In parallel, we derive geometric/control supervision (drafting and arrowing) from the same trajectory and speed signals; formal definitions are provided in Section B.3. Notably, for all appendix prompts, the purple text explains the input signals and the pink text specifies rules to follow so that the output satisfies the requirements shown in blue.



(a) A wheeled robot developed by Coco Robotics

(a) Teleoperation station

Figure 8. **Overview of the real-world experiment setup.** Left: the robot hardware platform; Right: the teleoperation interface.

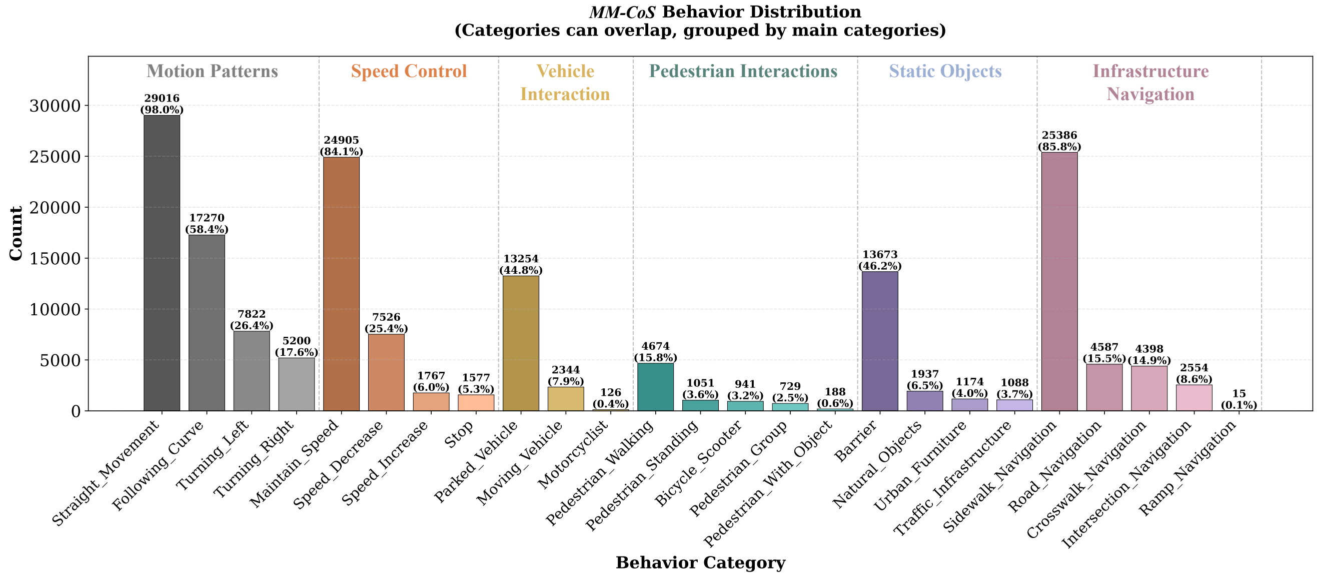


Figure 9. **Data distribution of our *MM-CoS* dataset (50h)**. Behaviors are divided into six subclasses based on their outcomes and causes. Bars of the same color represent different specific actions within each subclass. The top of each bar indicates the number of occurrences of the behavior and its proportion of the total annotations.

B.3. Multi-Modal Annotation Generation

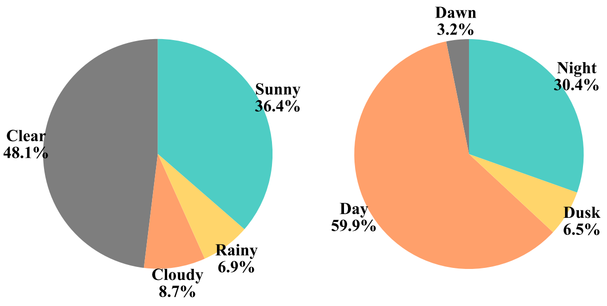
For each selected frame, we generate three complementary types of annotations to provide rich supervisory signals for navigation learning.

CMD `Drafting.` We project the robot's planned 4-second future trajectory from 3D world coordinates onto the 2D image plane using a pinhole camera model with intrinsic parameters $[f_x, f_y, c_x, c_y]$ and the corresponding 6-DoF camera poses. The projected waypoints are rendered as green trajectory lines overlaid on the RGB image $\boldsymbol{I}_d$, showing the intended path relative to the observed scene. Along this line, we uniformly sample n pixel points $\boldsymbol{p}_d$ to obtain the visual instruction $C_d = (\boldsymbol{I}_d, \boldsymbol{p}_d)$.

CMD `Arrowing.` We compute the average speed and directional control signals from the trajectory waypoints. Specifically, given the first N waypoints $\{\mathbf{p}_i\}_{i=1}^{N}$, the average velocity vector is calculated as $\mathbf{v}_{\text{avg}} = \frac{1}{N-1}\sum_{i=1}^{N-1}(\mathbf{p}_{i+1} - \mathbf{p}_i) \cdot \text{fps}$, and the corresponding heading angle as $\theta = \arctan 2(v_y, v_x)$. The velocity vector $\mathbf{v}_{\text{avg}}$ is then projected onto the front-view camera image and visualized as a fixed-length arrow indicating direction, with the absolute speed value overlaid as text, producing the image $\boldsymbol{I}_s$. The resulting arrowing instruction is defined as $C_s = (\boldsymbol{I}_s, \mathbf{v}_{\text{avg}})$.

CMD `Texting.` We employ a pre-trained vision-language model, Qwen2.5-VL [8], deployed with vLLM [32] for efficient infer-

Figure 10. **A thumbnail montage showing a subset of the *MM-CoS* videos collected from real world environments for shared autonomy.**



(a) Weather Distribution

(b) Time of Day Distribution

Figure 11. **Data distribution across weather conditions and time of day.** The dataset contains 3,040 scenes in total, with weather conditions primarily consisting of clear (48.1%) and sunny (36.4%) conditions, while cloudy (8.7%) and rainy (6.9%) conditions are less represented. Time-of-day distribution shows a strong bias toward daytime scenes (59.9%), followed by nighttime scenes (30.4%), with transitional periods of dawn (3.2%) and dusk (6.5%) being relatively scarce.

ence, to generate two levels of language supervision: (i) a short command-style instruction (verb phrase) describing the maneuver (e.g., “go straight”, “slow down”, “speed up”), and (ii) a more detailed natural-language description used to supervise reason-

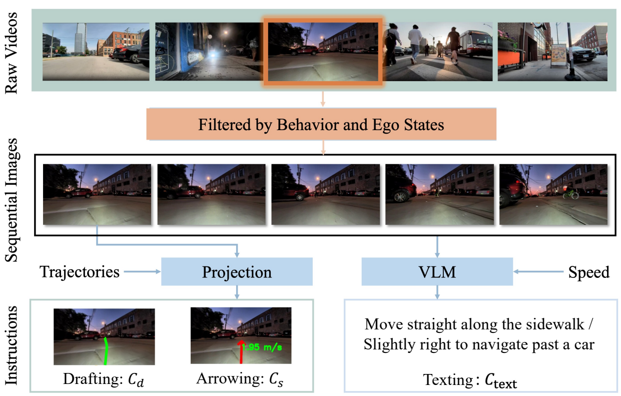
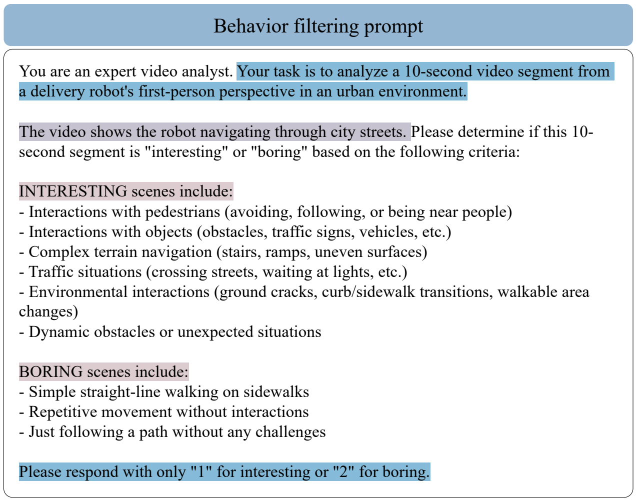

Raw Videos
Filtered by Behavior and Ego States
Sequential Images
Trajectories
Projection
VLM
Speed
Instructions
Drafting: C_d
Arrowing: C_s
Move straight along the sidewalk / Slightly right to navigate past a car
Texting : C_{text}

Figure 12. **Auto labeling pipeline.** We first select interesting clips based on behavioral cues and ego state information, focusing on frames that show rich interactions with objects or the environment. We then feed the front view images with visual prompts of the projected future trajectory and the speed of each frame into the VLM to generate textual instructions. Finally, we use the projected future trajectory on the front images to derive the drafting and arrowing instructions.



Behavior filtering prompt

You are an expert video analyst. Your task is to analyze a 10-second video segment from a delivery robot's first-person perspective in an urban environment.

The video shows the robot navigating through city streets. Please determine if this 10-second segment is "interesting" or "boring" based on the following criteria:

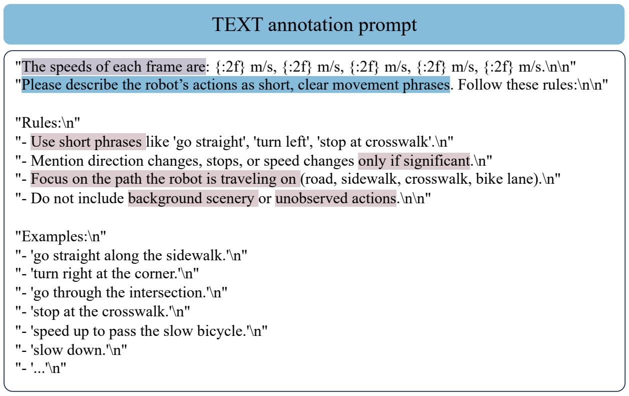
INTERESTING scenes include:
- Interactions with pedestrians (avoiding, following, or being near people)
- Interactions with objects (obstacles, traffic signs, vehicles, etc.)
- Complex terrain navigation (stairs, ramps, uneven surfaces)
- Traffic situations (crossing streets, waiting at lights, etc.)
- Environmental interactions (ground cracks, curb/sidewalk transitions, walkable area changes)
- Dynamic obstacles or unexpected situations

BORING scenes include:
- Simple straight-line walking on sidewalks
- Repetitive movement without interactions
- Just following a path without any challenges

Please respond with only "1" for interesting or "2" for boring.

Figure 13. **Prompt for behavior filter.** We use this prompt to let the VLM determine whether the robot in the clip is interacting with objects or the environment.

ing about the underlying scene and interactions. We sample future observation frames covering 4 seconds and overlay the corresponding 4-second trajectory visualizations on each frame. The VLM processes these temporally ordered frames together with frame-wise speed measurements to produce the instruction and the detailed description, covering directional changes, speed adjustments, obstacle avoidance, and interactions with the environment. These language annotations provide high-level semantic insights into navigation behaviors, complementing the geometric and control-level annotations.



TEXT annotation prompt

"The speeds of each frame are: {:2f} m/s, {:2f} m/s, {:2f} m/s, {:2f} m/s, {:2f} m/s.\n\n"
"Please describe the robot's actions as short, clear movement phrases. Follow these rules:\n\n"

"Rules:\n"
"- Use short phrases like 'go straight', 'turn left', 'stop at crosswalk'.\n"
"- Mention direction changes, stops, or speed changes only if significant.\n"
"- Focus on the path the robot is traveling on (road, sidewalk, crosswalk, bike lane).\n"
"- Do not include background scenery or unobserved actions.\n\n"

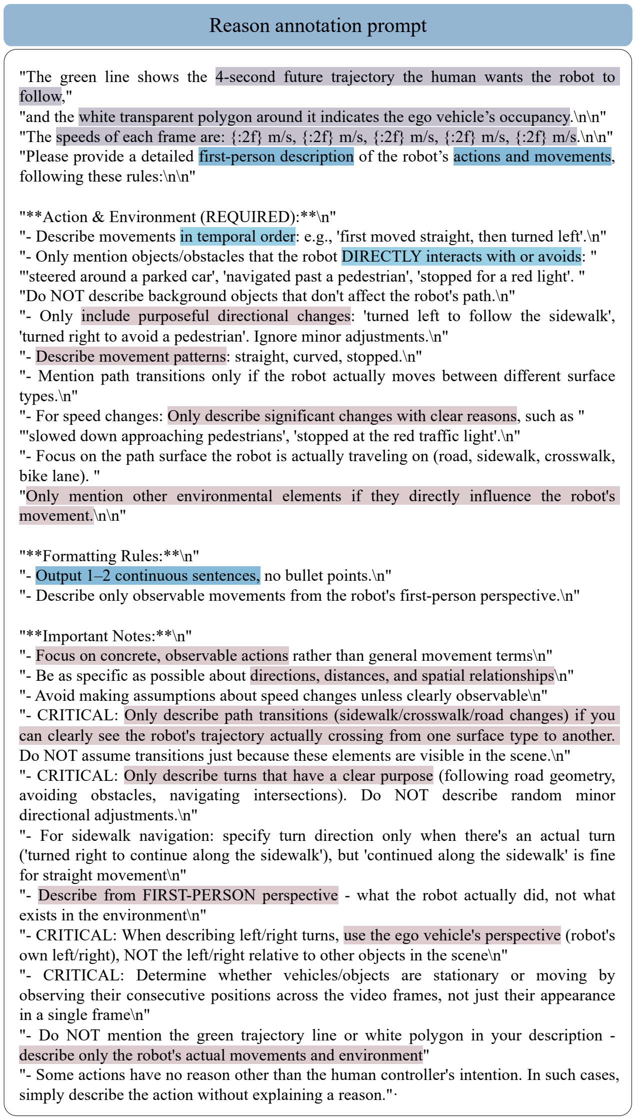
"Examples:\n"
"- 'go straight along the sidewalk.'\n"
"- 'turn right at the corner.'\n"
"- 'go through the intersection.'\n"
"- 'stop at the crosswalk.'\n"
"- 'speed up to pass the slow bicycle.'\n"
"- 'slow down.'\n"
"- '...'\n"

Figure 14. **Prompt for generating texting instructions.** We expect the VLM to produce short, simple instructions that guide the robot at a high level.

C. Details in Shared Control

C.1. The Judgment Module.

As shown in Figure 16, we employ both rule-based and VLM-based criteria to determine whether a human takeover is required.



Reason annotation prompt

"The green line shows the 4-second future trajectory the human wants the robot to follow."
"and the white transparent polygon around it indicates the ego vehicle's occupancy.\n\n"
"The speeds of each frame are: {:2f} m/s, {:2f} m/s, {:2f} m/s, {:2f} m/s, {:2f} m/s.\n\n"
"Please provide a detailed first-person description of the robot's actions and movements, following these rules:\n\n"

"**Action & Environment (REQUIRED):**\n"
"- Describe movements in temporal order: e.g., 'first moved straight, then turned left'.\n"
"- Only mention objects/obstacles that the robot DIRECTLY interacts with or avoids: "
"'steered around a parked car', 'navigated past a pedestrian', 'stopped for a red light'. "
"Do NOT describe background objects that don't affect the robot's path.\n"
"- Only include purposeful directional changes: 'turned left to follow the sidewalk', 'turned right to avoid a pedestrian'. Ignore minor adjustments.\n"
"- Describe movement patterns: straight, curved, stopped.\n"
"- Mention path transitions only if the robot actually moves between different surface types.\n"
"- For speed changes: Only describe significant changes with clear reasons, such as "
"'slowed down approaching pedestrians', 'stopped at the red traffic light'.\n"
"- Focus on the path surface the robot is actually traveling on (road, sidewalk, crosswalk, bike lane). "
"Only mention other environmental elements if they directly influence the robot's movement.\n\n"

"**Formatting Rules:**\n"
"- Output 1–2 continuous sentences, no bullet points.\n"
"- Describe only observable movements from the robot's first-person perspective.\n"

"**Important Notes:**\n"
"- Focus on concrete, observable actions rather than general movement terms\n"
"- Be as specific as possible about directions, distances, and spatial relationships\n"
"- Avoid making assumptions about speed changes unless clearly observable\n"
"- CRITICAL: Only describe path transitions (sidewalk/crosswalk/road changes) if you can clearly see the robot's trajectory actually crossing from one surface type to another. Do NOT assume transitions just because these elements are visible in the scene.\n"
"- CRITICAL: Only describe turns that have a clear purpose (following road geometry, avoiding obstacles, navigating intersections). Do NOT describe random minor directional adjustments.\n"
"- For sidewalk navigation: specify turn direction only when there's an actual turn ('turned right to continue along the sidewalk'), but 'continued along the sidewalk' is fine for straight movement\n"
"- Describe from FIRST-PERSON perspective - what the robot actually did, not what exists in the environment\n"
"- CRITICAL: When describing left/right turns, use the ego vehicle's perspective (robot's own left/right), NOT the left/right relative to other objects in the scene\n"
"- CRITICAL: Determine whether vehicles/objects are stationary or moving by observing their consecutive positions across the video frames, not just their appearance in a single frame\n"
"- Do NOT mention the green trajectory line or white polygon in your description - describe only the robot's actual movements and environment"
"- Some actions have no reason other than the human controller's intention. In such cases, simply describe the action without explaining a reason.".

Figure 15. **Prompt for generating reasoning.** We expect the VLM to generate a detailed description of the robot's behavior in the video, and to provide supervision signals (drafting and arrowing) for interpreting human instructions.

There are two primary situations that trigger a takeover. The first is collision risk. Specifically, we project the predicted trajectory together with the ego vehicle's occupancy polygon onto the front camera view to obtain a mask M_p. Using OpenSeed [66], we segment all pixels corresponding to impassable regions (such as obstacles or walls) as mask M_f. If the overlap between M_p and M_f exceeds 10% of M_p, the system flags the frame as requiring a takeover. The second is instruction compliance. To assess whether the predicted trajectory aligns with the human's original instructions, we input two rendered images, one with the predicted trajectory and one with the ground-truth trajectory, into QwenVL2.5-72B [8]. The model evaluates whether the prediction follows the intended goal, for example when the agent deviates from the cor-

Table 4. **Pseudo simulation evaluation on share control**.

	HO% ↓	TF ↓	OF ↓	HO_{1s}% ↓	TF_{1s} ↓	OF_{1s} ↓	HO_{2s}% ↓	TF_{2s} ↓	OF_{2s} ↓	HO_{3s}% ↓	TF_{3s} ↓	OF_{3s} ↓
MBRA [22]	15.1	0.757	0.757	30.5	0.305	1.523	41.6	0.211	2.082	48.1	0.164	2.407
CityWalker* [36]	19.6	0.978	0.978	40.3	0.404	2.016	54.5	0.276	2.725	60.7	0.207	3.037
AURA 📍	6.0	0.298	0.298	14.9	**0.151**	0.744	22.4	**0.116**	1.122	28.0	**0.098**	1.400
AURA 📝	7.2	0.359	0.359	9.0	0.195	0.450	12.5	0.170	0.624	15.0	0.152	0.751
AURA 🎮	5.0	0.248	0.248	6.4	0.169	0.320	9.1	0.152	0.456	11.2	0.144	0.562
AURA ✏	**4.5**	**0.226**	**0.226**	**5.7**	0.163	**0.284**	**7.7**	0.150	**0.386**	**9.7**	0.142	**0.483**

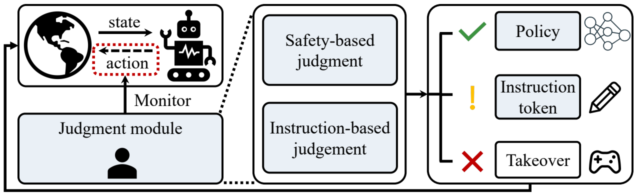


Figure 16. **Judgment module.** After the policy updates its state and receives the observation, it produces a candidate action through a forward pass of the model, which is temporarily cached before execution. We then apply a judgment module that evaluates the action using both safety-based criteria and instruction-based constraints. Based on this evaluation, the system decides whether to (i) execute the action, (ii) inject an instruction token to guide the policy toward human instruction, or (iii) trigger a full human takeover when the candidate action violates critical safety conditions.

rect path or remains stationary while the human intends to move.

C.2. More Results

These experiments evaluate how much human intervention time can be reduced during teleoperation. To ensure reproducibility, we conduct all evaluations in a pseudo-simulation environment. We select 29 scenarios from the test set in which every clip is annotated as *interesting*.

Additional results on Human Cost Evaluation in pseudo-simulation are shown in Table 4. Here, HO denotes the human operation ratio, defined as the percentage of time under human control. TF represents the takeover frequency, calculated as the total number of takeover events divided by the total duration. OF indicates the operation frequency, defined as the number of operation actions per unit time. The first three columns without footnotes assume that human takeover occurs instantaneously. In our actual testing setup, the system runs at 5 Hz, meaning that a full takeover requires 0.2 seconds of human control. The footnoted columns report results under the assumption that each full takeover lasts 1, 2, or 3 seconds.

The results indicate a clear trend: as humans intend to sustain longer takeover periods, the total amount of required intervention decreases. Notably, when the human remains in control for longer durations, more potential takeover events are absorbed within that interval and therefore do not trigger additional interventions.so the takeover frequency are reducing while the human takeover time increase.

Judgement module prompt

"You are a delivery robot safety judge. Based on the following images:\n\n"
"1. Predicted trajectory from the autonomous system (first image).\n"
"2. Reference trajectory from the human operator (second image).\n\n"
"Green line = future robot path.\n"
"White polygon = robot occupancy at each future pose.\n"
"Red points with timestamps = future positions.\n"
"If no green line/polygon, the robot is stopped (no collision risk).\n\n"

"Decide if the human operator should take over.\n\n"
"Rules:\n"
"- Predicted trajectory should generally follow the reference direction and path, though length may differ.\n"
"- Predicted trajectory should generally follow the reference route, though timestamp and the end point may differ.\n"

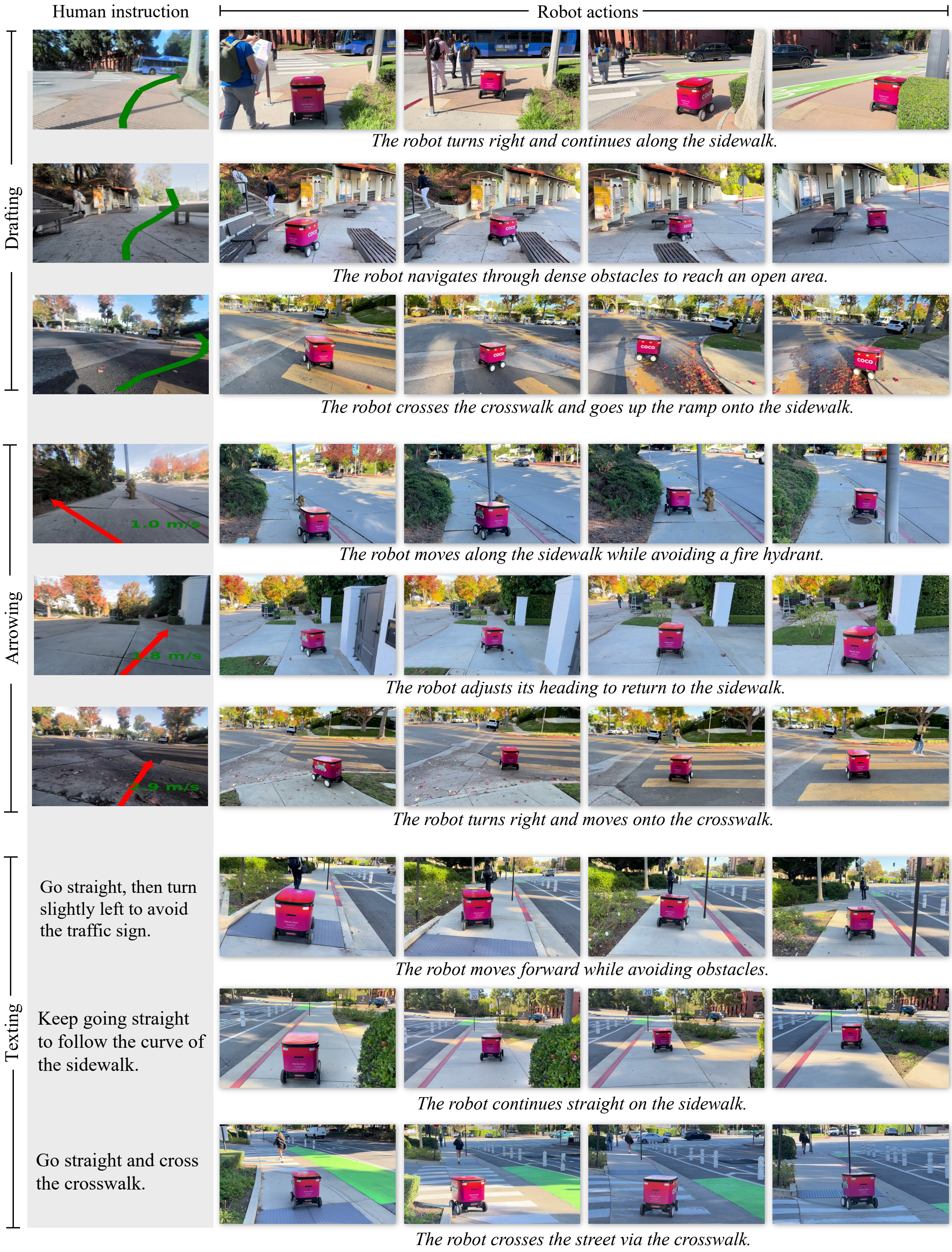
"- A takeover is required if:\n"
" • White polygon overlaps obstacles, pedestrians, or vehicles.\n"
" • Trajectory goes into invalid areas (buildings, roads, restricted zones).\n"
" • Trajectory clearly deviates from the reference path (e.g., wrong turn, different branch).\n"
" • Predicted motion is unsafe (abrupt turns, unsafe proximity, excessive speed).\n"
"- Minor deviations in shape or length are acceptable if overall direction/path is consistent.\n"
"- Judge only on the given prediction. Do not imagine unseen events.\n\n"

"Answer with one word: 'YES' (takeover required) or 'NO' (not required)."

Figure 17. **Prompt for the VLM judge.** The judge module is required to output only yes or no. It checks whether the predicted trajectory matches the human-intended action. If the robot's intent diverges from the human intention, human intervention is needed. For example, the robot may take an incorrect path or make unnecessary stops.

D. Implementation Details

Our training pipeline consists of two stages. For stage 1, we train the SIE and the LoRA-adapted LLM to generate action captions conditioned on the drafting and the arrowing prompt. The modules are trained on eight A6000 GPUs using a cosine learning rate schedule with a base learning rate of 2×10^{-5}, weight decay of 0.05, and a warmup ratio of 0.03, for 5 epochs with a global batch size of 32. For stage 2, we freeze all other components and train only the action decoder in an end-to-end manner. The policy is trained on eight RTX 4090 GPUs using the AdamW optimizer with a learning rate of 1×10^{-4} for 100 epochs and a global batch size of 32. The AdamW hyperparameters are set to $\beta_1 = 0.9$ and $\beta_2 = 0.99$.



The robot turns right and continues along the sidewalk.

The robot navigates through dense obstacles to reach an open area.

The robot crosses the crosswalk and goes up the ramp onto the sidewalk.

The robot moves along the sidewalk while avoiding a fire hydrant.

The robot adjusts its heading to return to the sidewalk.

The robot turns right and moves onto the crosswalk.

Go straight, then turn slightly left to avoid the traffic sign.

The robot moves forward while avoiding obstacles.

Keep going straight to follow the curve of the sidewalk.

The robot continues straight on the sidewalk.

Go straight and cross the crosswalk.

The robot crosses the street via the crosswalk.

Figure 18. **Qualitative results in the real world.** The first three rows are guided by drafting, rows 4 to 6 by arrowing, and the last three rows by texting. The first column shows human instructions, while columns 2 to 5 show the actions executed by the robot.