Urbex: Agentic Spatial Grounding for City-Scale 3D Scenes

Yu Wang[1,2] **Rui Dai**[2*] **Wei Chen**[1] **Yi Wang**[2] **Bo Dang**[1]
Kaikui Liu[2] **Xiangxiang Chu**[2†] **Yansheng Li**[1†]
[1]Wuhan University [2]DreamX Team, Alibaba Group

Abstract

City-scale 3D grounding is commonly formulated as candidate ranking, which requires precomputed instance proposals and becomes inefficient in large outdoor scenes. We propose Urbex, an agentic vision-language framework that instead casts grounding as active spatial search. Urbex represents a city-scale 3D scene as an interactive multi-view environment, allowing a VLM agent to query landmarks, zoom into local regions, render oblique views, and commit a grounded bounding box. We optimize this tool-use policy with reinforcement learning on CityRefer, using smooth localization rewards and lightweight shaping rewards for efficient exploration. Experiments show that Urbex achieves the best Hit@1 and IoU on CityRefer, supports candidate-free grounding without ground-truth boxes, and is substantially faster than the strongest city-scale ranking baseline. Zero-shot results on STPLS3D-Refer further indicate improved cross-scene robustness. The code and dataset are publicly available at: https://github.com/AMAP-ML/Urbex.

1 Introduction

Grounding objects in 3D scenes from natural language descriptions [1] is a core capability for embodied intelligence, with applications in autonomous driving [2], robotic navigation [3], and aerial inspection [4]. While indoor 3D grounding has been extensively studied [5, 6], city-scale 3D grounding is still underexplored [7]. Outdoor urban scenes are much larger than indoor scans, often containing hundreds of thousands to tens of millions of points per scene [8]. The associated referring expressions are also more abstract, as a query may describe a small object through a named building, a road direction, or a distant landmark, e.g., "a white car parked next to the orange car to the left of the West Midlands Police Custody Suite." Solving such queries requires both fine-grained visual recognition and reasoning over long-range spatial cues.

Existing 3D grounding methods designed for indoor scenes [5, 9] typically downsample raw point clouds to a fixed size, such as 50,000 points. When applied to city-scale scenes, this strategy discards substantial fine-grained geometry and often leads to suboptimal performance, as illustrated in Figure 1b. To avoid processing the entire point cloud, recent city-scale methods instead formulate grounding as candidate ranking [7, 10], as shown in Figure 1c. They first obtain object proposals from ground-truth annotations or an off-the-shelf instance segmentation pipeline [11], and then compute text-instance matching scores for each candidate. Although effective in reducing input scale, this formulation remains tightly coupled to precomputed candidates: its accuracy depends on proposal quality, while its inference cost grows with the number of candidates to be scored [12]. More importantly, it reduces grounding to a static retrieval problem, whereas localization in a vast urban environment is naturally sequential and exploratory [13, 14].

*Project leader.
†Corresponding authors.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

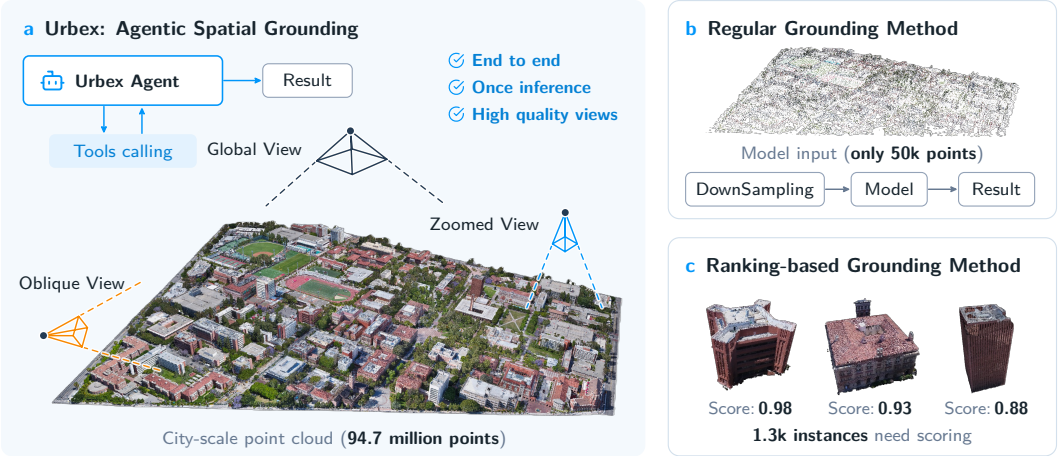


Figure 1: Overview of Urbex. Unlike regular grounding methods that downsample city-scale point clouds or ranking-based methods that score many precomputed instances, Urbex formulates grounding as agentic spatial search. A VLM agent actively interacts with the scene through tool calls, moving between global, zoomed, and oblique views before committing the final target location.

In this work, we instead formulate city-scale grounding as an active spatial search problem. Given a referring expression and an initial global observation, a model should parse the target attributes and spatial cues, choose where to inspect next, acquire new evidence, and decide when the evidence is sufficient to commit an answer. Recent tool-using language and vision-language agents [15, 16] provide a natural substrate for this formulation, as they can reason over visual observations and interact with external tools [17–19]. However, generic tool use is not sufficient. City-scale 3D grounding requires a task-specific interface that exposes the scene at the right level of abstraction, allowing the agent to move between global context and local detail without exhaustively enumerating all 3D instances.

We propose Urbex, an agentic spatial grounding framework for city-scale 3D scenes. As illustrated in Figure 1a, Urbex starts from a global bird's-eye view (BEV), uses landmark queries to establish coarse spatial anchors, zooms into promising regions for fine-grained inspection, renders oblique views for 3D confirmation, and finally commits a box that is projected back to world coordinates. Urbex represents each scene as an interactive multi-view environment built around a BEV image and a consistent view-to-world coordinate mapping. The VLM reasons over the accumulated interaction history and either calls another tool or commits a final bounding box in a known BEV-aligned view. The system then projects the box back to world coordinates and evaluates it against the target instance. We train Urbex with reinforcement learning on CityRefer [7]. The grounding episode is modeled as a multi-step decision process. A terminal reward encourages high overlap with the target and provides a dense center-distance signal when overlap is initially sparse. In addition, tool-level shaping rewards guide intermediate behavior, such as zooming toward the target region. This objective optimizes both final localization accuracy and the agent's intermediate search policy. Our contributions are summarized as follows:

- We propose Urbex, an agentic framework that reformulates city-scale 3D grounding from exhaustive candidate ranking to active spatial search.
- We design a city-aware interactive tool interface that enables a VLM agent to search large urban scenes through landmark querying, local zooming, and oblique-view verification, and optimize this multi-step policy with reinforcement learning.
- Urbex achieves the best Hit@1 and IoU on CityRefer, substantially improves online inference efficiency over the strongest ranking baseline, and demonstrates strong zero-shot robustness on STPLS3D-Refer.

2 Related work

2.1 3D Scene Grounding

3D scene grounding aims to localize a specific object within a 3D environment based on a natural-language referring expression. A large body of prior work has predominantly focused on indoor scenes, where early methods typically encode entire point clouds or multi-view images, fuse them with a language encoder, and rank candidate instances via cross-modal matching [5, 6, 20]. Subsequent research has significantly advanced this paradigm by incorporating relation modeling, multi-view context, and transformer-based architectures [9, 21], and exploring proposal-free, bottom-up localization to avoid error accumulation from segmentation models [12, 22]. Recent zero-shot and open-world methods further exploit foundation models, scene parsing, multi-view reasoning, or scene graphs to reduce dependence on task-specific supervised training [23–26]. These methods are highly relevant to the goal of reducing proposal dependence, but they are primarily designed for indoor or object-centric 3D scenes and do not directly address city-scale landmark reasoning, very large search spaces, or online efficiency over dense urban instances.

City-scale 3D grounding presents challenges that are substantially different from those in indoor scenes. Outdoor urban environments cover much larger spatial extents, contain many repetitive instances such as vehicles and buildings, and often require reasoning over long-range spatial cues and named landmarks. As a result, existing city-scale methods typically adopt a static candidate-ranking pipeline, where instance boxes are obtained in advance, multimodal features are extracted offline, and a scoring model ranks all candidates during inference [7, 10]. Although this formulation avoids directly processing the entire city-scale point cloud, it remains constrained by the quality of precomputed proposals, requires costly feature preprocessing, and becomes inefficient when many candidates must be scored. More importantly, it treats localization as a one-shot retrieval problem. In contrast, Urbex frames city-scale grounding as sequential spatial interaction, actively searching the scene through a small number of targeted observations.

2.2 Agentic Vision-Language Models

Modern vision-language models (VLMs) have demonstrated exceptional capabilities in visual question answering, fine-grained reasoning, and instruction following [15, 16, 27–29]. RL further improves reasoning capabilities [30]. Building upon these foundation models, recent research has evolved from passive perception to active, agentic systems capable of calling tools, manipulating graphical interfaces, and interpreting complex visual environments [19, 31–33]. Instead of outputting a single-pass prediction, such agentic frameworks interact with their surroundings through a continuous loop of observation, reasoning, action, and feedback [17, 18, 34–37].

However, much of the agentic VLM literature focuses either on 2D tasks, such as web navigation and open-ended image reasoning, or on indoor embodied navigation [38, 39]. Recent work has begun to formulate 3D grounding itself as an agentic or sequential process, including dynamic 2D-to-3D reconstruction with visual tools [40], active cognition over open-world object lookup structures [23], graph-guided VLM reasoning over 3D scene structure [24], and task-oriented sequential grounding in indoor scenes [41]. While conceptually close to our active-search view, these works operate in indoor RGB-D or graph-structured settings and are not optimized for city-scale 3D environments. Urbex specializes the agentic VLM paradigm for this challenging domain by providing city-aware spatial tools and an RL objective tailored for efficient exploration.

3 Method

Urbex formulates city-scale 3D grounding as active spatial search rather than static candidate retrieval. The central design choice is to expose a large 3D scene through a small number of coordinate-aligned visual observations that a VLM can inspect and act upon. We first describe how the environment converts city-scale geometry into a VLM-operable multi-view space, then define the tool-use reasoning loop over these views, and finally present the reinforcement learning objective that turns prompted tool use into a learned search policy.



Figure 2: Architecture of Urbex: (left) a multi-view environment exposing global BEV, zoomed, and oblique views; (middle) the policy's reasoning trace and tool calls over observation o_t and interaction history h_t; (right) RL under a terminal reward (IoU and center-distance) plus tool-level shaping.

3.1 Interactive environment for city-scale 3D grounding

Given a city-scale 3D scene $\mathcal{S}$ and a natural-language query q, the goal is to localize the referred object $o^\star$ and output its spatial extent. A conventional candidate-ranking method assumes a candidate set $\mathcal{C} = \{o_i\}_{i=1}^{N}$ and learns a score $s(q, o_i)$ for each candidate, making the quality and size of $\mathcal{C}$ central to both accuracy and efficiency [7, 10]. In contrast, Urbex receives no candidate set and exposes the scene through views $\mathcal{V}$ that the agent can extend through tool calls. It predicts a free-form box $\hat{b}_v$ in a known BEV-aligned view v, which the environment maps to a world-space extent. Ground-truth boxes provide training rewards and evaluation targets, but are not inference inputs.

View representation. The initial view is a global bird's-eye-view image I_{global}. Each BEV-aligned view $v \in \mathcal{V}$ contains an image observation I_v and a mapping T_v from the view coordinate frame to world coordinates. The agent expresses boxes in a normalized 2D coordinate frame. The environment maintains the corresponding view-to-world transform, so a box predicted in any such view can be projected back to world coordinates. This design gives the VLM a simple and consistent spatial interface while preserving metric grounding in the original 3D scene.

City-aware tools. The environment provides three complementary tools for global anchoring, local detail recovery, and 3D verification, keeping the action space compact. *Landmark Query* resolves a named landmark from the query, such as a building or road name, and returns its world position and its projection in a selected view. This tool provides global spatial anchors when the expression contains proper nouns. This design is motivated by the geography-aware nature of city-scale referring expressions, where named buildings, roads, and other landmarks provide strong global anchors [7, 10, 42]. *Zoom-in* maps a selected box to the original high-resolution BEV and returns a local crop with a new view identifier and transform. The original BEV has a long side typically exceeding 10,000 pixels, so cropping recovers genuine detail absent from the downsampled global observation, where small objects occupy only a few pixels. *Oblique Rendering* renders a selected candidate region from a specified cardinal direction, producing a 3D-aware view that helps verify object shape, height, and spatial placement. The oblique image is used as supporting evidence rather than as the coordinate frame for the final submission. Detailed tool definitions and argument schemas are provided in Table 5.

3.2 Agentic reasoning with multi-view tool use

The policy π_θ is a VLM that receives the current observation o_t and the interaction history h_t. The history contains the original query, previous tool calls, returned view identifiers, coordinate

conventions, and visual observations. At step t, the policy produces a reasoning trace and an action:

$$(\text{thought}_t, a_t) \sim \pi_\theta(o_t, h_t). \tag{1}$$

The action a_t is either a tool call with structured arguments or a final answer. A tool call may specify landmark names, a source view, a 2D crop box, a candidate box, or an oblique rendering direction. The environment executes the tool and appends a textual response and, when applicable, a new image observation to the history.

The resulting inference procedure naturally decomposes into global search, local observation, and candidate confirmation. The agent first parses the expression to identify target attributes and spatial anchors in the BEV, then zooms into promising regions to resolve small objects or dense local context, and finally renders oblique views when top-down evidence is insufficient to verify the 3D object. Because every tool response is appended to the interaction history, the model can refer back to previous views and maintain coordinate consistency while switching between global and local evidence. This protocol can be used directly with a prompted VLM, yielding a zero-shot tool-use baseline, but the search space remains large. The model must learn not only how to interpret each view, but also when to query, zoom, render, or stop, which motivates the RL training described next.

3.3 Reinforcement learning for active spatial search

We model an episode as a finite-horizon Markov decision process [43]. The state s_t consists of the dialogue history, the current set of views, and tool feedback. The action space includes valid tool invocations and final answer submission. Given an action and a scene, the environment transition is deterministic; trajectory-level stochasticity comes from policy sampling. The episode terminates when the agent submits an answer or reaches the maximum number of assistant turns.

Terminal reward. Let $\hat{b}$ be the submitted world-space box and $b^\star$ be the ground-truth target box. We use a smooth terminal reward:

$$R_{\text{term}} = \lambda_{\text{iou}} \,\text{IoU}(\hat{b}, b^\star) + \lambda_{\text{dist}} \exp\left(-\frac{d(\hat{b}, b^\star)}{\tau}\right) - \lambda_{\text{step}} \frac{T}{T_{\max}}, \tag{2}$$

where $d(\hat{b}, b^\star)$ is the center distance in the ground plane, T is the number of assistant turns, and $T_{\max}$ is the interaction budget. The IoU term rewards precise localization. The distance term provides a continuous learning signal even when overlap is zero, which is common early in training for vast city-scale scenes and has been shown to be useful in spatial navigation tasks [44]. The step cost discourages unnecessarily long trajectories. We do not use a discrete success bonus; the reward grows smoothly with localization quality rather than introducing a threshold cliff.

Tool-level shaping. In addition to the terminal reward, we employ lightweight process rewards [45] to shape the intermediate tool-calling behavior [46]:

$$R(\tau) = R_{\text{term}} + \sum_{t=1}^{T} R_{\text{tool}}(s_t, a_t). \tag{3}$$

For *Zoom-in*, the reward is positive when the zoom center moves closer to the target compared with the previous zoom and negative when it moves away, with a small cap to prevent it from dominating the terminal objective. For *Oblique Rendering*, the reward is proportional to the overlap between the rendered candidate region and the target region, again capped to keep it as a shaping signal. Invalid tool names, malformed arguments, impossible boxes, or tool execution errors receive negative rewards. These rewards are privileged training signals derived from ground-truth annotations, analogous to reward functions used in simulated navigation or RLVR settings. They are used only for policy optimization and are never included in the agent's text observation, so they do not leak target information at inference time.

Optimization. We train Urbex on CityRefer [7] entirely via reinforcement learning using Group Relative Policy Optimization [47]. To circumvent the prohibitive memory overhead of a separate value network (critic), it estimates the baseline directly from multiple concurrent rollouts. Specifically, for each input query and initial environment state, the behavior policy $\pi_{\theta_{\text{old}}}$ samples a group of G

trajectories $\{\tau_1, \tau_2, \ldots, \tau_G\}$. After evaluating the cumulative reward $R(\tau_i)$ for each trajectory, we compute the group-relative advantage:

$$A_i = \frac{R(\tau_i) - \mu_R}{\sigma_R + \epsilon}, \tag{4}$$

where μ_R and σ_R are the mean and standard deviation of the group's rewards. The policy is then optimized by maximizing the clipped GRPO objective:

$$\mathcal{J}_{\mathrm{GRPO}}(\theta) = \frac{1}{G}\sum_{i=1}^{G}\frac{1}{T_i}\sum_{t=1}^{T_i}\left[\min(r_{i,t}(\theta)A_i,\ \mathrm{clip}(r_{i,t}(\theta), 1-\epsilon_c, 1+\epsilon_c)\,A_i) - \beta D_{\mathrm{KL}}^{i,t}\right]. \tag{5}$$

where $r_{i,t}(\theta) = \pi_\theta(a_{i,t} \mid s_{i,t})/\pi_{\theta_{\mathrm{old}}}(a_{i,t} \mid s_{i,t})$, ϵ_c is the clipping threshold, and β controls the token-level KL penalty to the reference policy π_{ref}. By relying solely on this RL objective without supervised trajectory demonstrations, the model explores the multi-view environment and learns a task-specific tool-use policy, including when to query landmarks, how large a zoom region should be, whether an oblique view justifies its step cost, and when the accumulated evidence is sufficient to commit a final bounding box.

4 Experiments

4.1 Experimental setup

We evaluate Urbex along three axes, namely in-domain performance, cross-scene generalization, and component sensitivity. CityRefer [7] is used as the main benchmark for comparing grounding accuracy and inference efficiency, while STPLS3D-Refer, our evaluation set constructed from STPLS3D [48], tests whether the learned policy transfers to scenes with different geometry and visual appearance. We convert both datasets into the same BEV-based interactive environment and use the same action protocol and evaluation code for all tool-using models. Detailed dataset statistics are provided in Appendix B, while training and inference protocols, hyperparameters, and other implementation details are provided in Appendix A.

Baselines. We compare three groups of methods that stress different aspects of the problem. General-purpose multimodal models, including GPT-5.4 and Qwen3.6-plus, provide strong visual reasoning priors. We connect them to the same Urbex tool environment to test whether a powerful off-the-shelf VLM can solve the task through prompting alone, without task-specific policy optimization. Indoor 3D grounding methods reveal two different failure modes when moved to city-scale scenes. EDA follows the common indoor practice of processing a downsampled point cloud, which discards much of the fine-grained geometry needed for small urban objects [9]. InstanceRefer instead operates in a candidate-ranking setting with provided instances, which is easier than direct full-scene localization and therefore performs more reasonably than dense point-cloud grounding methods [21]. City-scale methods such as CityRefer and CityAnchor are closer to our setting, but they still primarily rely on ranking precomputed candidates rather than actively searching the scene [7, 10]. Details on each baseline implementation are provided in Appendix A.3.

Evaluation settings and metrics. We report results on the novel description (ND) and novel object (NO) splits: ND uses unseen descriptions in scenes observed during training, while NO uses targets from held-out scenes that do not appear in training. In *With GT*, candidate-ranking baselines may use the ground-truth candidate instance set and are evaluated with Hit@1, Hit@3, and Hit@5. Urbex does not use candidate boxes during inference even in this setting; it always outputs a free-form BEV box, which is matched to candidates only after prediction: candidates are ranked by 2D IoU with the predicted box, and Hit@k records whether the target is among the top k, without a minimum overlap requirement. This measures target identification; thresholded Hit@1 and direct overlap metrics additionally assess box quality (Appendix D.5). In *Without GT*, methods cannot access ground-truth candidate boxes and must directly output a spatial prediction; we therefore report 2D IoU, 3D IoU, and point-level F1 against the target. This setting is closer to deployment, where complete and accurate instance boundaries are rarely available. Rows whose metrics would all be inapplicable are omitted from the corresponding subtable; otherwise inapplicable cells are marked as n/a. GPT-5.4 and Qwen3.6-plus do not expose comparable timing measurements, while InstanceRefer, CityRefer,



Figure 3: Qualitative examples on CityRefer. The visualization shows how Urbex moves from global BEV search to local zoom-in and final grounding in city-scale scenes with dense repeated objects and landmark-based references.

Table 1: Main results on CityRefer. Urbex supports candidate-free grounding and efficient inference. Five-run Hit@1 statistics are reported in Table 8.

Method	ND With GT			NO With GT			Time (s)
	Hit@1	Hit@3	Hit@5	Hit@1	Hit@3	Hit@5	
Closed-source VLMs w/ Urbex tools							
GPT-5.4	9.3	12.4	15.7	13.1	18.6	20.1	n/a
Qwen3.6-plus	6.4	9.6	11.9	11.5	16.2	16.8	n/a
Indoor 3D grounding methods							
InstanceRefer [21]	1.6	5.0	10.5	4.9	15.9	24.5	**0.6**
EDA [9]	0.1	0.1	0.2	0.2	0.3	0.4	3.0
Ranking-based city-scale methods							
CityRefer [7]	1.6	3.9	4.8	0.6	2.1	3.4	0.7
CityAnchor [10]	23.4	35.3	**45.4**	14.2	22.4	31.0	32.1
Urbex	**25.8**	**39.2**	42.3	**23.9**	**36.9**	**40.7**	3.2

Method	ND Without GT			NO Without GT		
	2D IoU	3D IoU	Point F1	2D IoU	3D IoU	Point F1
Closed-source VLMs w/ Urbex tools						
GPT-5.4	4.4	2.2	9.4	4.6	1.9	10.4
Qwen3.6-plus	3.0	1.8	9.5	4.3	2.1	12.1
Indoor 3D grounding methods						
EDA [9]	0.2	0.2	0.3	0.5	0.4	0.8
Urbex	**15.6**	**11.0**	**20.1**	**14.6**	**7.7**	**17.8**

and CityAnchor require candidate instances; we separately test CityAnchor with predicted proposals in Appendix D.4. Accuracy metrics are percentages. We report inference time per example on a single GPU, including model generation and online tool execution, while excluding offline scene preparation and cached rendering construction for all methods.

4.2 Main results on CityRefer

Table 1 reports the main CityRefer results. GPT-5.4 and Qwen3.6-plus obtain non-trivial accuracy after being connected to the Urbex tool environment, showing that strong VLMs can exploit the

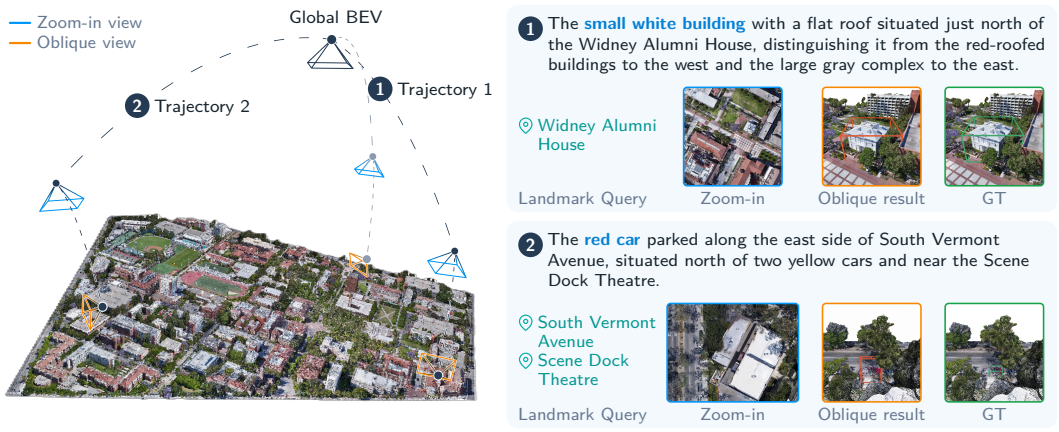


Figure 4: Qualitative examples on STPLS3D-Refer. The visualization highlights zero-shot transfer cases where the same active-search policy anchors, zooms, verifies, and grounds targets in scenes with different geometry and object distributions from CityRefer.

interactive views and tool feedback to perform partial active search. However, their performance remains far below Urbex, showing that these prompted tool-use baselines do not recover the accuracy of the trained policy. The indoor baselines illustrate why directly transferring indoor 3D grounding pipelines is also insufficient. EDA almost fails on both splits, consistent with the severe information loss caused by downsampling city-scale point clouds before grounding. InstanceRefer performs better because it is evaluated as a simpler candidate-ranking problem with provided instances, but it still remains far below city-scale methods and cannot directly solve candidate-free localization.

Among city-scale baselines, CityRefer is limited when evaluated as a full ranking problem because its original setting assumes that the target appears in a small 10-instance candidate pool rather than requiring discrimination over all scene instances [7]. CityAnchor is much stronger because it scores candidates with multimodal large language models, but this design requires applying the scoring model to many instances and therefore incurs high inference cost. In the With-GT setting, Urbex achieves the best Hit@1 and Hit@3 on both test splits and the best Hit@5 on NO. CityAnchor retains the highest Hit@5 on ND: exhaustive ranking can retain several plausible GT candidates, including a target that Urbex misses during regional search or confuses with a local distractor. Thus, broader candidate recall and precise single-box search have different failure modes (Appendix D.6).

The advantage becomes clearer in the Without-GT setting, where candidate-ranking methods require a proposal-generation stage. General multimodal models can use the same prompted tool interface, but their 2D and 3D IoU remain substantially lower than Urbex. To test a complete proposal-based pipeline, we train SoftGroup [49] on 15 non-CityRefer SensatUrban scenes and supply its proposals to CityAnchor. On NO, this pipeline reaches 4.2/2.4/4.7 in BEV IoU/3D IoU/point F1, versus 14.6/7.7/17.8 for Urbex (Appendix D.4), showing the practical gap for this proposal-based pipeline.

Inference efficiency. Urbex is also substantially more efficient than the strongest ranking baseline under the online timing protocol above. CityAnchor, the second-best-performing method in the With-GT comparison, requires about 32.1 seconds per example because it scores many candidate instances, whereas Urbex takes about 3.2 seconds. This amounts to an order-of-magnitude inference-speed improvement, since Urbex inspects a small number of informative regions rather than exhaustively scoring all instances. The tools account for about 0.36 s/query. Building the optional oblique cache takes approximately 2.3 hours across 12 CityRefer scenes; rendering on demand instead gives an estimated 4.4 s/query (7.3× faster). These comparisons exclude offline scene preparation; Appendix D.7 separates the measured cache cost from other, unmeasured preparation components.

4.3 Zero-shot generalization on STPLS3D-Refer

We next evaluate zero-shot transfer to STPLS3D-Refer. The agent is trained only on CityRefer and is directly applied to STPLS3D-Refer after converting scenes into the same BEV-based interactive environment. This setting tests whether the model has learned a reusable spatial search strategy rather

Table 2: Zero-shot generalization on STPLS3D-Refer. Table 8 reports five-run Hit@1 statistics.

Method	With GT			Without GT			Time (s)
	Hit@1	Hit@3	Hit@5	2D IoU	3D IoU	Point F1	
Closed-source VLMs w/ Urbex tools							
GPT-5.4	16.3	17.8	19.1	8.0	5.3	12.1	n/a
Qwen3.6-plus	12.0	13.8	14.3	8.1	6.4	12.2	n/a
Indoor 3D grounding methods							
InstanceRefer [21]	3.8	7.8	12.3	n/a	n/a	n/a	**0.6**
EDA [9]	0.0	0.0	0.5	0.4	0.2	0.4	2.9
Candidate-dependent city-scale methods							
CityRefer [7]	1.4	2.5	3.8	n/a	n/a	n/a	1.4
CityAnchor [10]	2.1	4.2	6.0	n/a	n/a	n/a	47.2
Urbex	**27.2**	**32.1**	**33.4**	**13.0**	**10.0**	**16.2**	4.8

than a dataset-specific ranking function. As shown in Table 2, the new scenes are challenging for all compared methods: indoor grounding methods are sensitive to domain shift, and city-scale ranking models degrade when candidate distributions and visual statistics differ from CityRefer. GPT-5.4 and Qwen3.6-plus perform reasonably well after being connected to the Urbex tool environment, again showing that strong VLM reasoning can exploit interactive observations even without task-specific training. However, Urbex achieves the best score on every reported accuracy metric. These aggregate results suggest, but do not by themselves prove, that the RL-trained policy learns reusable search behaviors such as anchoring, zooming, verifying, and stopping. To examine transfer beyond named geographic anchors, we additionally evaluate 30 manually verified landmark-free re-annotations of STPLS3D-Refer. Without further training, Urbex achieves 23.3 Hit@1 on these expressions, providing complementary evidence that the learned policy transfers to queries expressed through visual attributes and local spatial relations. Appendix D.3 details this evaluation and the landmark-robustness experiments; Figure 4 illustrates the agent's spatial search and grounding trajectories in the new scenes.

4.4 Ablation studies

Tables 3 and 4 ablate the tool interface and the reward design. Tool variants add *Landmark Query* and *Oblique Rendering* on top of *Zoom-in*, while reward variants retain terminal IoU and progressively add distance and tool-level shaping terms.

Table 3: Tool and budget ablation for Urbex.

Zoom	Landmark	Oblique	Turns	Hit@3	2D IoU
–	–	–	0	10.5	2.1
✓	–	–	8	15.5	4.1
✓	✓	–	8	28.2	8.9
✓	✓	✓	7	34.8	11.0
✓	✓	✓	8	**36.9**	**14.6**
✓	✓	✓	9	36.8	13.2

Table 4: Reward ablation for Urbex.

Reward setting	Hit@3	2D IoU
IoU only	22.8	6.7
IoU + Distance	30.6	10.9
IoU + Tool-level	28.4	9.6
Full reward	**36.9**	**14.6**

The ablation results show that each component contributes a distinct part of the active-search behavior. Without tools, the agent sees the entire downsampled global map but reaches only 10.5 Hit@3 and 2.1 2D IoU, compared with 36.9 and 14.6 using all tools. Adding *Zoom-in* improves both metrics, confirming that high-resolution local evidence is important for small and densely arranged urban objects. The largest tool gain comes from adding *Landmark Query*: Hit@3 rises from 15.5 to 28.2, and 2D IoU more than doubles from 4.1 to 8.9. This supports our premise that named buildings, roads, and other geographic anchors are central to city-scale referring expressions. In contrast to indoor scenes where a local crop may already contain most relevant context, city-scale queries often require first locating the correct neighborhood before fine-grained recognition becomes meaningful.

Oblique Rendering further improves the result by helping the agent verify candidate regions with 3D-aware visual evidence. With all tools enabled, increasing the budget from 7 to 8 turns improves Hit@3 from 34.8 to 36.9 and 2D IoU from 11.0 to 14.6, suggesting that one additional interaction is often useful for confirmation before submission. Increasing the budget to 9 turns does not

improve Hit@3 and slightly lowers 2D IoU, indicating that excessive interaction can introduce unnecessary exploration or late-stage box adjustments. We therefore use the 8-turn setting as the main configuration, which provides the best accuracy-efficiency trade-off in this ablation. This also suggests that Urbex benefits from a bounded search process rather than unbounded deliberation.

The reward ablation confirms the same pattern. Terminal IoU alone reaches 22.8 Hit@3 and 6.7 2D IoU; adding distance shaping improves them to 30.6 and 10.9, tool-level shaping is also beneficial, and the full reward performs best. Distance shaping is especially useful when early predictions have no overlap with the target, while tool-level rewards make tool use more stable. Their combination shows that localization quality, distance progress, and efficient tool use are complementary.

4.5 Analysis

Tool-use policy. We further examine the learned policy using a fixed Landmark Query $\rightarrow$ Zoom-in $\rightarrow$ Oblique Rendering $\rightarrow$ Answer workflow (Appendix D.2). In the reported experiments, the base Qwen model reaches only 2.7 Hit@1, whereas RL-trained Urbex reaches 10.0 under the same tool order. The fixed workflow still requires the model to predict tool arguments and the final box. The improvement therefore suggests that training helps the model use the observed scene to make spatial decisions, such as selecting a crop and locating the target within it. The workflow is also useful for a stronger prompted VLM: GPT-5.4 maintains similar Hit@1 while reducing token use from 16,321 to 7,734. This suggests that the learned tool sequence provides an efficient way to organize search, while the trained model contributes to how the tools are used.

Spatial search. The landmark experiments further show the importance of geographic cues for city-scale search. On CityRefer-NO expressions with named landmarks, the 10 m coordinate-noise setting lowers Hit@1 by only 1.2 points, whereas missing or incorrect landmarks cause drops exceeding ten points (Appendix D.3). This suggests that a moderately inaccurate anchor can still guide the search, but losing the reference required by the expression is substantially more harmful. Figure 10 illustrates errors at different stages of this process. In (a), the first zoom excludes the target, and the agent selects another car within the crop. In (b), the zoom contains the target, but the agent selects vehicles in a different parking row. Both predictions remain unchanged after oblique inspection. The first case calls for revisiting the search region, while the second requires closer comparison of nearby instances using the query's spatial relations. These examples motivate improving both region selection and local verification (Appendix D.6).

Localization quality. We also examine the quality of the free-form boxes produced by Urbex (Appendix D.5). On CityRefer-NO, replacing the original IoU matching rule with positive-overlap filtering or CIoU changes Hit@1 by at most 0.29 points, indicating that target identification is stable under these matching rules. However, additionally requiring the first-ranked target to have BEV IoU ≥ 0.25 reduces Hit@1 from 23.53 to 18.71. A prediction can therefore identify the correct candidate while still including substantial surrounding context. The area-ratio statistics show the same issue: the predicted-to-target area ratio has median 1.40 and P90 9.92, indicating that some boxes are considerably larger than the target. These results suggest that refining the box after target selection could further improve localization. We report direct IoU and point F1 alongside Hit@k to assess both target identification and the accuracy of the predicted spatial extent.

5 Conclusion

We presented Urbex, an agentic framework for city-scale 3D grounding. Rather than ranking a large set of precomputed candidates, Urbex formulates grounding as active spatial search in an interactive multi-view environment. We further optimize this tool-use policy with reinforcement learning under a limited interaction budget, encouraging accurate and efficient exploration. Experiments on CityRefer show that Urbex achieves strong grounding accuracy, supports candidate-free localization, and substantially improves online inference efficiency over ranking-based baselines. Zero-shot results on STPLS3D-Refer further suggest that active spatial search provides a robust alternative to static candidate retrieval for large-scale 3D scene understanding.

Acknowledgments and Disclosure of Funding

This work was partly supported by the Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China under Grant JYB2025XDXM111, the National Natural Science Foundation of China under Grant 42371321, and the Key Research and Development Program of Hubei Province under Grant 2025BAB024.

References

[1] Daizong Liu, Yang Liu, Wencan Huang, and Wei Hu. A survey on text-guided 3-d visual grounding: Elements, recent advances, and future directions. *IEEE Transactions on Neural Networks and Learning Systems*, 2025.

[2] Wenhao Cheng, Junbo Yin, Wei Li, Ruigang Yang, and Jianbing Shen. Language-guided 3d object detection in point cloud for autonomous driving. *arXiv preprint arXiv:2305.15765*, 2023.

[3] Junha Roh, Karthik Desingh, Ali Farhadi, and Dieter Fox. Languagerefer: Spatial-language model for 3d visual grounding. In *Conference on Robot Learning*, pages 1046–1056. PMLR, 2022.

[4] Rong Li, Yuhao Dong, Tianshuai Hu, Ao Liang, Youquan Liu, Dongyue Lu, Liang Pan, Lingdong Kong, Junwei Liang, and Ziwei Liu. 3eed: Ground everything everywhere in 3d. *arXiv preprint arXiv:2511.01755*, 2025.

[5] Dave Zhenyu Chen, Angel X Chang, and Matthias Nießner. Scanrefer: 3d object localization in rgb-d scans using natural language. In *European conference on computer vision*, pages 202–221. Springer, 2020.

[6] Panos Achlioptas, Ahmed Abdelreheem, Fei Xia, Mohamed Elhoseiny, and Leonidas Guibas. Referit3d: Neural listeners for fine-grained 3d object identification in real-world scenes. In *European conference on computer vision*, pages 422–440. Springer, 2020.

[7] Taiki Miyanishi, Fumiya Kitamori, Shuhei Kurita, Jungdae Lee, Motoaki Kawanabe, and Nakamasa Inoue. Cityrefer: Geography-aware 3d visual grounding dataset on city-scale point cloud data. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

[8] Qingyong Hu, Bo Yang, Sheikh Khalid, Wen Xiao, Niki Trigoni, and Andrew Markham. Sensaturban: Learning semantics from urban-scale photogrammetric point clouds. *International Journal of Computer Vision*, 130(2):316–343, 2022.

[9] Yanmin Wu, Xinhua Cheng, Renrui Zhang, Zesen Cheng, and Jian Zhang. Eda: Explicit text-decoupling and dense alignment for 3d visual grounding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19231–19242, 2023.

[10] Jinpeng Li, Haiping Wang, Jiabin Chen, Yuan Liu, Zhiyang Dou, Yuexin Ma, Sibei Yang, Yuan Li, Wenping Wang, Zhen Dong, and Bisheng Yang. Cityanchor: City-scale 3d visual grounding with multi-modality LLMs. In *The Thirteenth International Conference on Learning Representations*, 2025.

[11] Li Jiang, Hengshuang Zhao, Shaoshuai Shi, Shu Liu, Chi-Wing Fu, and Jiaya Jia. Pointgroup: Dual-set point grouping for 3d instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and Pattern recognition*, pages 4867–4876, 2020.

[12] Junyu Luo, Jiahui Fu, Xianghao Kong, Chen Gao, Haibing Ren, Hao Shen, Huaxia Xia, and Si Liu. 3d-sps: Single-stage 3d visual grounding via referred point progressive selection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16454–16463, 2022.

[13] Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Russ R Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. *Advances in Neural Information Processing Systems*, 33:4247–4258, 2020.

[14] Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12538–12547, 2019.

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

[16] 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.

[17] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

[18] Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

[19] Yuxiang Ji, Yong Wang, Ziyu Ma, Yiming Hu, Hailang Huang, Xuecai Hu, Guanhua Chen, Liaoni Wu, and Xiangxiang Chu. Thinking with map: Reinforced parallel map-augmented agent for geolocalization. *arXiv preprint arXiv:2601.05432*, 2026.

[20] Shijia Huang, Yilun Chen, Jiaya Jia, and Liwei Wang. Multi-view transformer for 3d visual grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15524–15533, 2022.

[21] Zhihao Yuan, Xu Yan, Yinghong Liao, Ruimao Zhang, Sheng Wang, Zhen Li, and Shuguang Cui. Instancerefer: Cooperative holistic understanding for visual grounding on point clouds through instance multi-level contextual referring. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1791–1800, 2021.

[22] Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. Language conditioned spatial relation reasoning for 3d object grounding. *Advances in neural information processing systems*, 35:20522–20535, 2022.

[23] Wenyuan Huang, Zhao Wang, Zhou Wei, Ting Huang, Fang Zhao, Jian Yang, and Zhenyu Zhang. Openground: Active cognition-based reasoning for open-world 3d visual grounding. *arXiv preprint arXiv:2512.23020*, 2025.

[24] Yuanyuan Liu, Haiyang Mei, Dongyang Zhan, Jiayue Zhao, Dongsheng Zhou, Bo Dong, and Xin Yang. View-on-graph: Zero-shot 3d visual grounding via vision-language reasoning on scene graphs. *arXiv preprint arXiv:2512.09215*, 2025.

[25] Jiaxi Zhang, Yunheng Wang, Wei Lu, Taowen Wang, Weisheng Xu, Shuning Zhang, Yixiao Feng, Yuetong Fang, and Renjing Xu. Uniground: Universal 3d visual grounding via training-free scene parsing. *arXiv preprint arXiv:2603.08131*, 2026.

[26] Qiao Gu, Ali Kuwajerwala, Sacha Morin, Krishna Murthy Jatavallabhula, Bipasha Sen, Aditya Agarwal, Corban Rivera, William Paul, Kirsty Ellis, Rama Chellappa, Chuang Gan, Celso Miguel De Melo, Joshua B. Tenenbaum, Antonio Torralba, Florian Shkurti, and Liam Paull. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5021–5028, 2024.

[27] Yuhao Zheng, Li'an Zhong, Yi Wang, Rui Dai, Kaikui Liu, Xiangxiang Chu, Linyuan Lv, Philip Torr, and Kevin Qinghong Lin. Code2world: A gui world model via renderable code generation. *arXiv preprint arXiv:2602.09856*, 2026.

[28] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

[29] Xiangxiang Chu, Limeng Qiao, Xinyu Zhang, Shuang Xu, Fei Wei, Yang Yang, Xiaofei Sun, Yiming Hu, Xinyang Lin, and Bo Zhang. Mobilevlm v2: Faster and stronger baseline for vision language model. *arXiv preprint arXiv:2402.03766*, 2024.

[30] Xiangxiang Chu, Hailang Huang, Xiao Zhang, Fei Wei, and Yong Wang. GPG: A simple and strong reinforcement learning baseline for model reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026.

[31] Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11888–11898, 2023.

[32] Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14953–14962, 2023.

[33] Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh J Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *Advances in Neural Information Processing Systems*, 37:52040–52094, 2024.

[34] Ziyu Ma, Shidong Yang, Yuxiang Ji, Xucong Wang, Yong Wang, Yiming Hu, Tongwen Huang, and Xiangxiang Chu. Skillclaw: Let skills evolve collectively with agentic evolver. *arXiv preprint arXiv:2604.08377*, 2026.

[35] Zhenlong Yuan, Xiangyan Qu, Chengxuan Qian, Rui Chen, Jing Tang, Lei Sun, Xiangxiang Chu, Dapeng Zhang, Yiwei Wang, Yujun Cai, and Shuo Li. Video-STAR: Reinforcing open-vocabulary action recognition with tools. In *The Fourteenth International Conference on Learning Representations*, 2026.

[36] Yu Wang, Yi Wang, Rui Dai, Yujie Wang, Kaikui Liu, Xiangxiang Chu, and Yansheng Li. Urban socio-semantic segmentation with vision-language reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026.

[37] Yi Wang, Haopeng Zhang, Chengxiang Huang, Rui Dai, Kaikui Liu, Piotr Koniusz, and Xiangxiang Chu. LoopArena: Benchmarking models as runtime controllers for loop engineering. *arXiv preprint arXiv:2608.28281*, 2026.

[38] Xiaojian Ma, Silong Yong, Zilong Zheng, Qing Li, Yitao Liang, Song-Chun Zhu, and Siyuan Huang. Sqa3d: Situated question answering in 3d scenes. *The Eleventh International Conference on Learning Representations*, 2023.

[39] Jiangyong Huang, Silong Yong, Xiaojian Ma, Xiongkun Linghu, Puhao Li, Yan Wang, Qing Li, Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. An embodied generalist agent in 3d world. In *The Twelfth International Conference on Learning Representations*, 2024.

[40] Haibo Wang, Zihao Lin, Zhiyang Xu, and Lifu Huang. Think, act, build: An agentic framework with vision language models for zero-shot 3d visual grounding. *arXiv preprint arXiv:2604.00528*, 2026.

[41] Zhuofan Zhang, Ziyu Zhu, Pengxiang Li, Tengyu Liu, Xiaojian Ma, Yixin Chen, Baoxiong Jia, Siyuan Huang, and Qing Li. Task-oriented sequential grounding in 3d scenes. *arXiv preprint arXiv:2408.04034*, 2024.

[42] OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org. https://www.openstreetmap.org, 2026.

[43] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2 edition, 2018.

[44] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9339–9347, 2019.

[45] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The twelfth international conference on learning representations*, 2023.

[46] Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pages 278–287. Citeseer, 1999.

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

[48] Meida Chen, Qingyong Hu, Zifan Yu, Hugues Thomas, Andrew Feng, Yu Hou, Kyle McCullough, Fengbo Ren, and Lucio Soibelman. Stpls3d: A large-scale synthetic and real aerial photogrammetry 3d point cloud dataset. In *33rd British Machine Vision Conference*, 2022.

[49] Thang Vu, Kookhoi Kim, Tung M. Luu, Thanh Nguyen, and Chang D. Yoo. SoftGroup for 3D instance segmentation on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2708–2717, 2022.

A Implementation Details

A.1 Hyperparameters and Hardware

Urbex is initialized from Qwen3-VL-4B-Instruct [15]. Training and evaluation use the same city-tool interface, action schema, coordinate convention, and execution code for both CityRefer and STPLS3D-Refer. Unless otherwise specified, each trajectory allows at most eight assistant turns: up to seven tool calls and one final answer turn.

Training configuration. Urbex is trained on the CityRefer training split with GRPO [47]: rollout batch size 256, group size 8, 300 optimization steps, and one PPO epoch per rollout batch. The actor learning rate is 1×10^{-6}, the KL coefficient is 0.001, and bf16 training freezes the vision encoder. The reward combines final IoU, center-distance shaping, a step cost, and tool-level shaping rewards, with $\lambda_{\text{iou}} = 1.0$, $\lambda_{\text{dist}} = 0.3$, $\tau = 80$m, and $\lambda_{\text{step}} = 0.05$ in Equation 2.

Hardware and timing protocol. Training uses 16 NVIDIA H20 GPUs. Local-model inference and timing use a single NVIDIA H20 GPU; closed-source VLMs are accessed through their APIs. We report average online inference time per query, including model generation and online tool execution, while excluding offline scene preprocessing such as cached renderings and candidate-pool preparation for all methods.

A.2 Tool Interface

Table 5 summarizes the tool interface shared by prompted VLM baselines and Urbex. Before execution, the environment validates tool names, argument types, box bounds, and view identifiers.

A.3 Baseline Settings

Closed-source VLMs use the same Urbex tool environment and maximum-turn budget but no task-specific policy optimization. GPT-5.4 and Qwen3.6-plus receive the same action format, coordinate convention, tool descriptions, final-answer format, and tool-output history, then submit a final BEV-aligned box. These baselines test whether strong off-the-shelf VLMs can solve city-scale grounding with prompting and tool access alone.

InstanceRefer [21] and EDA [9] are adapted from indoor 3D grounding pipelines by following their released preprocessing and downsampling each city-scale point cloud to 40,000 points. InstanceRefer is evaluated with provided instances in its native candidate-ranking setting and is therefore omitted from candidate-free Without-GT evaluation; EDA uses the same point budget. This preserves the indoor pipelines while exposing the loss of fine-grained geometry needed for small or repetitive urban objects.

Table 5: Definitions, inputs, and outputs of the city-aware tools.

Tool	Definition	Inputs	Outputs
Landmark Query	Resolves a named urban landmark, such as a building, road, or facility, and uses it as a global spatial anchor.	`landmark_name`: text span or proper noun; `source_view_id`: view in which the returned projection should be expressed; optional `landmark_type` when the name is ambiguous.	Match status; matched landmark name and type; world-space center or footprint when available; projection of the landmark into the requested view.
Zoom-in	Maps a region from an existing BEV view to the original high-resolution image and crops it, exposing detail absent from the global observation.	`source_view_id`: global or local BEV view; `crop_box`: normalized 2D box; optional `scale` or requested output resolution.	Local BEV image; new `view_id`; updated view-to-world transform; crop extent in the source and world coordinate frames.
Oblique Rendering	Renders an agent-selected candidate region from a cardinal direction to provide 3D-aware verification of shape, height, facade appearance, and spatial placement.	`source_view_id`: BEV view containing the candidate; `candidate_box`: normalized 2D region; `direction`: one of north, south, east, or west; optional `margin` around the region.	Oblique image observation; rendering metadata including direction and covered world region; supporting evidence for verification. The oblique view is not used as the coordinate frame for final box submission.

CityRefer and CityAnchor [7, 10] are retrained on our CityRefer split with the original hyperparameters, optimizer, schedule, and loss weights unless noted. For CityAnchor, we use the released ROI-localization and candidate-scoring pipeline, lower the ROI threshold to 0.1 to retain candidates under our larger pool, use landmark features from the retrained CityRefer language module as officially recommended, and include online candidate scoring in its timing.

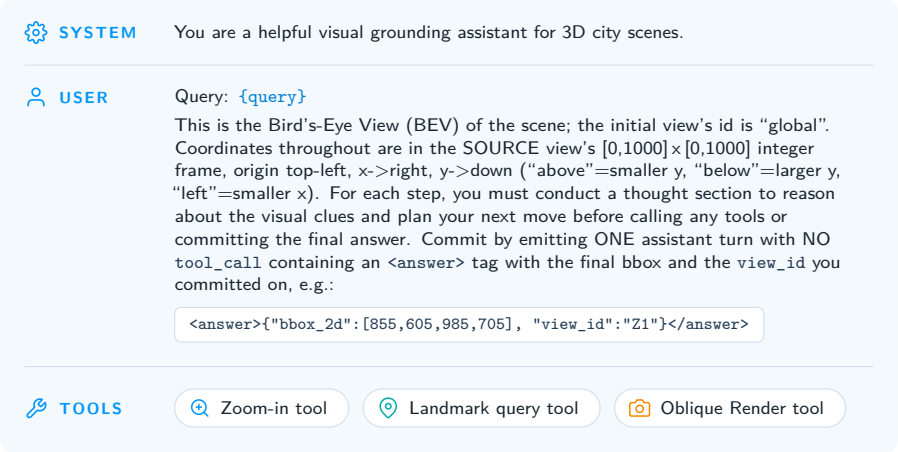


Figure 5: Prompt template for tool-using VLM agents, specifying the task, tools, coordinates, action format, and final box format shared by prompted VLM baselines and Urbex.

B Dataset Details

B.1 CityRefer Data Preparation

CityRefer [7], built on SensatUrban [8], is our in-domain benchmark. We remove duplicate `(scene, query)` rows and low-quality annotations with visibly incorrect instance labels or boxes, then create train, validation, ND, and NO splits. ND contains unseen descriptions in training scenes; NO tests held-out-scene generalization. Seven scenes are reserved for NO, and the remaining five are split across train, validation, and ND with a fixed random seed, yielding 2,744 training, 549 validation, 1,098 ND, and 1,396 NO queries.

Table 6: Scale of the 12 CityRefer scenes used in our experiments. Test queries combine ND and NO.

Statistic	Total across 12 scenes	Per-scene range
Area (km^2)	1.594	0.029–0.165
Points (millions)	877.4	12.1–116.3
GT candidate objects	2,234	31–367
OSM landmarks	299	9–51
Test queries	2,494	14–556

Small objects are difficult to distinguish in a downsampled global BEV. We therefore retain a high-resolution BEV for each scene and use *Zoom-in* to retrieve detailed local crops during search.

B.2 STPLS3D-Refer Construction

STPLS3D-Refer is a cross-domain zero-shot benchmark for testing whether a CityRefer-trained policy transfers to new urban geometry, appearance, and geography. It is derived from public STPLS3D RealWorld photogrammetry point clouds [48] and used only for evaluation. Because STPLS3D lacks CityRefer-style expressions and object instances, we combine geometric instance extraction, geographic registration, OpenStreetMap landmarks [42], and VLM-assisted expression generation to build a diagnostic, semi-automatic benchmark.

Source scenes and target categories. From four STPLS3D drone-photogrammetry scenes, we retain the two with reliable public geographic context: University of Southern California Park Campus (USC, 94.7M points, 1083×807m) and Wrigley Marine Science Center (WMSC, 9.98M points, 490×293m). We exclude one scene dominated by a single mega-building and one without verifiable public geographic context. Targets are limited to *Building* and *Vehicle*.

Instance extraction. Since STPLS3D provides semantic labels but not instances, we cluster semantic masks with DBSCAN: buildings use planar xy coordinates with eps$=2.0$m, min_samples$=80$, and footprint filter $[4, 5000]\text{m}^2$; vehicles use xyz with eps$=0.6$m, min_samples$=30$, and footprint filter $[1.5, 30]\text{m}^2$. From more than 1,000 pseudo-instances, we remove fragmented clusters, merged objects, ground/noise artifacts, and projection-ambiguous candidates through geometric filters and manual inspection. The verified pool contains 59 buildings/62 vehicles in USC and 11 buildings/17 vehicles in WMSC, with axis-aligned 3D boxes and point masks computed for each retained cluster.

Geo-registration and landmarks. Each scene is registered to WGS-84 by fitting a first-order affine transform from 5–7 ground-control points selected against recent satellite imagery, with RMS residuals of 0.72m for USC and 2.19m for WMSC. We query OpenStreetMap features using tags including `building`, `amenity`, `highway`, `leisure`, `tourism`, `office`, and `natural`, project them back to local coordinates, and match landmarks to building instances when applicable. The two scenes contain 278 real OSM landmarks.

Scene conversion and baseline compatibility. Each scene is converted to the CityRefer interface: global BEV image, view-to-world mapping, Zoom-in crops, optional oblique renderings, and annotations for candidate-based and candidate-free evaluation. We also export candidate metadata and per-instance features required by CityAnchor-style pipelines [10]; ranking methods use explicit candidates, while Urbex receives only the visual tool environment and submits a free-form BEV box.

Referring-expression generation and filtering. For each target, we render a centered BEV crop together with oblique side views from the four cardinal directions, mark the target and nearby same-class distractors in all rendered views, and give the target color, size, distractor relations, and nearest OSM landmarks to Qwen3.6-plus to generate three concise CityRefer-style expressions. The oblique views supply 3D-aware cues such as building height, facade appearance, and relative layout that are difficult to disambiguate from the BEV alone. Automatic filters check length, meta-language, landmark-name validity, projection validity, and token-level duplication. The pipeline first yields more than 1,000 preliminary target-expression samples; manual inspection then removes unclear targets, incorrect landmarks, severe instance artifacts, and ambiguous descriptions, leaving 445 high-confidence landmark-grounded samples.

Table 7: Statistics of STPLS3D-Refer. "Inst." counts unique candidate instances, and "Lndmk." counts named OSM landmarks.

Scene	Inst. (Bldg./Veh.)	Building Expr.	Vehicle Expr.	Total Expr.	Lndmk.
USC	59/62	177	184	361	272
WMSC	11/17	33	51	84	6
Total	70/79	210	235	445	278

C Additional Qualitative Examples

Figure 6 provides additional CityRefer examples, and Figure 8 shows full agent trajectories from query and global BEV observation to Landmark Query, Local Zoom-in, Oblique Rendering, and final box prediction.

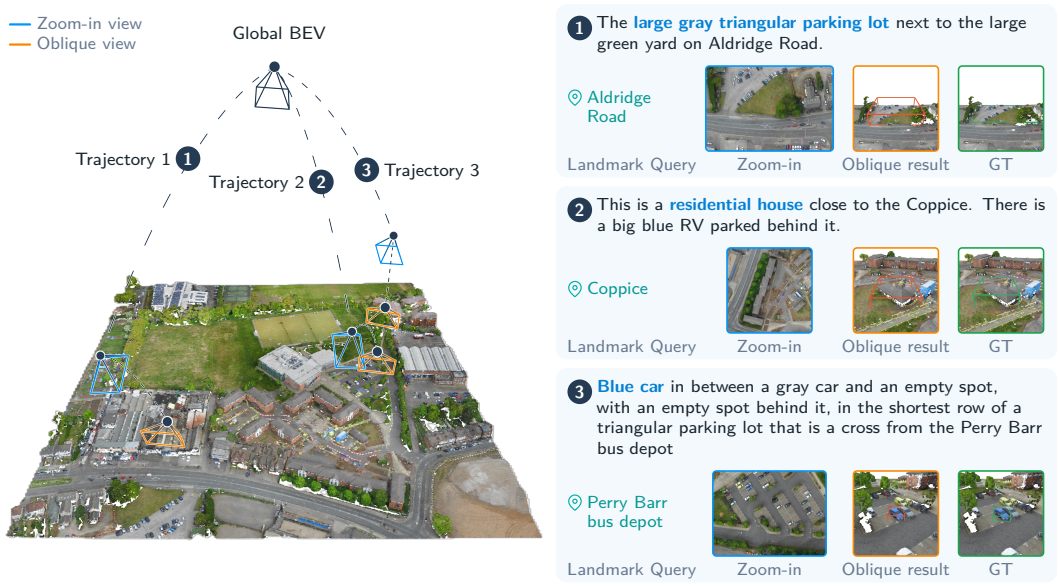


Figure 6: Additional qualitative examples on the CityRefer NO split.

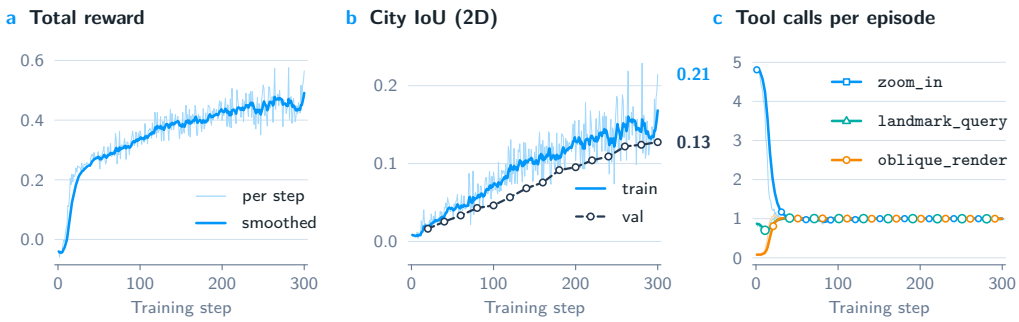


Figure 7: Training dynamics on CityRefer. (a) Total task reward. (b) Train/validation 2D grounding IoU with periodic validation markers and final-value annotations. (c) Per-episode tool-call frequency, showing convergence from over-calling *zoom_in* and under-using *oblique_render* to one call per tool.

Figure 7 summarizes the first 300 CityRefer optimization steps. Total reward rises from slightly negative values to a stable positive regime, while train and validation 2D IoU improve together with a small, stable gap, suggesting generalization rather than memorization. Tool-use statistics show the policy moving from inefficient exploration to a disciplined sequence: early training overuses *zoom_in* (peak $\approx$ 4.7 calls, first-30-step mean 2.6) and underuses *oblique_render* (mean $\approx$ 0.6), but later

The blue car between an empty parking space and a red car in the parking lot at Buckingham Room.



1 Landmark Query

look up a named landmark on the global BEV

THOUGHT

To locate the blue car between an empty parking space and a red car, I should first identify the Buckingham Room, as it is likely to be near the parking lot mentioned. Using the landmark lookup tool, I can find the coordinates of the Buckingham Room to narrow down the area where the blue car might be.

ACTION

```
landmark_query(names=[Buckingham Room]
  view_id="global")
```

RESULT

```
get_landmark_xy results:
 - 'Buckingham Room' → world (360.0,1528.3);
   on view global at (897,179)
```

2 Zoom In

agent crops a region of interest

THOUGHT

Now that I have the coordinates of the Buckingham Room at (897, 179) on the global view, I can focus on this area to locate the blue car between an empty parking space and a red car. The next step is to zoom in on this region to better visualize the parking lot and identify the specific car mentioned.

ACTION

```
zoom_in(view_id="global"
  bbox_2d=[800, 100, 1000, 300])
```

RESULT

```
zoom_in ok → new view 'Z1'
```

3 Oblique Render

verifies a candidate bbox in 3D · prediction (red)

THOUGHT

Now that I've zoomed in to view Z1, I can clearly see the parking lot near the Buckingham Room. The description mentions a blue car between an empty parking space and a red car. In the zoomed-in view, I can identify a row of parked cars, including red and blue vehicles. The blue car in question is likely to be adjacent to a red car and an empty space. I will now use the render_oblique tool to visually confirm the 3D structure of the blue car and its surroundings, ensuring it matches the description.

ACTION

```
oblique_render(view_id="Z1"
  bbox_2d=[515, 890, 565, 935] direction="east")
```

RESULT

```
render_oblique ok
```

4 Submit Answer

agent's final answer (no tool call)

THOUGHT

The 3D oblique view has successfully highlighted the blue car, which is positioned between an empty parking space and a red car, matching the query's description. The red 3D wireframe in the oblique view confirms the correct location of the blue car. Now, I will commit to the final answer using the specified format, as the visual confirmation has been achieved.

ACTION

```
answer = {"bbox_2d": [515, 890, 565, 935],
  "view_id": "Z1"}
```

RESULT

```
2D IoU = 0.678 → ✓ success
```

Figure 8: Example agent reasoning processes from the initial query and global scene observation to landmark anchoring, local zoom-in, oblique verification, and final box commitment.

converges to using each tool roughly once per episode. This matches the Landmark Query → Local Zoom-in → Oblique Rendering pattern shown in Figure 8.

D Controlled Experiments and Diagnostic Analysis

We report additional experiments on policy execution, landmark robustness, proposal dependence, matching rules, localization quality, and scene preparation. Unless stated otherwise, accuracy values are percentages. Tables 1 and 2 retain the original evaluation run; repeated-run summaries below are reported separately rather than mixing means with single-run metrics.

D.1 Repeated Decoding Runs

Table 8 summarizes five stochastic decoding runs of the same trained checkpoint on fixed test sets. The reported uncertainty is the sample standard deviation across runs, in percentage points. It measures inference-time sampling variation, not variability across independently trained GRPO policies. An approximate 95% Student-t interval for the ND mean is $25.62 \pm t_{4,0.975}(0.33/\sqrt{5}) = [25.21, 26.03]$, using $t_{4,0.975} \simeq 2.776$. This interval lies above CityAnchor's reported 23.4, supporting a margin relative to the observed decoding variation. It is not a paired significance test: it neither captures test-sample uncertainty nor accounts for training or evaluation variation in the baseline. Independent training seeds and paired per-query comparisons would be needed to assess those sources of uncertainty.

Table 8: Hit@1 over five decoding runs of one checkpoint (mean ± sample standard deviation). Baselines are the strongest comparison results in the original main tables.

Dataset	Urbex Hit@1	Comparison Hit@1	Comparison method
CityRefer-ND	25.62 ± 0.33	23.4	CityAnchor
CityRefer-NO	23.53 ± 0.73	14.2	CityAnchor
STPLS3D-Refer	26.34 ± 0.58	16.3	GPT-5.4

D.2 Fixed Tool Workflows and Learned Execution

After RL training, a common trajectory follows Landmark Query → Zoom-in → Oblique Rendering → Answer (L-Z-O). This order emerges during learning rather than being prescribed in the training protocol. We extract it and enforce it at inference to form a scripted baseline with the same tool interface. The model still predicts tool arguments and the final box; "fixed" constrains the tool sequence, not the spatial coordinates. Table 9 compares the reported free and fixed-policy results.

Table 9: Free and fixed L-Z-O tool policies. The fixed workflow is extracted from the learned policy; tokens are reported per query.

Model	Policy	Hit@1	Hit@3	Hit@5	Tokens/query
Qwen3-VL-4B base	Fixed	2.7	5.0	6.0	6,576
Urbex RL	Free	23.5	37.0	40.9	7,187
Urbex RL	Fixed	10.0	16.0	22.0	6,786
GPT-5.4	Free	13.1	18.6	20.1	16,321
GPT-5.4	Fixed	13.5	23.0	26.5	7,734

Under the fixed workflow, RL-trained Urbex reaches 10.0 Hit@1 versus 2.7 for the base model, indicating that the trained model contributes beyond access to a useful tool sequence. Free Urbex reaches 23.5, but this gap should not be attributed solely to choosing a different order: most free trajectories already follow L-Z-O. Externally forcing the sequence also changes the interaction context in which arguments and answers are generated. The comparison shows that replaying a tool order does not recover the full learned execution policy; it does not isolate tool ordering from argument generation or prompting effects.

For GPT-5.4, the fixed workflow maintains similar Hit@1 (13.5 versus 13.1), improves Hit@3/5 from 18.6/20.1 to 23.0/26.5, and reduces token use from 16,321 to 7,734. Thus, the discovered workflow is also a useful efficiency prior for an untuned VLM, even though it does not reproduce Urbex's full accuracy.

D.3 Landmark Availability and Construction Bias

We separate CityRefer-NO into 1,302 expressions with named landmarks and 94 without them. For landmark-guided expressions, *Missing* returns no match, *Noisy 10 m* perturbs the correct coordinate by 10 m, and *Wrong* returns a same-scene landmark at least 50 m away. Table 10 reports the resulting Hit@1. Landmark-free rows use their own expressions and are not perturbations of the same queries.

Table 10: Landmark robustness (Hit@1, %). Dashes denote inapplicable landmark perturbations. The 30-example landmark-free STPLS3D set is a separate diagnostic.

Model / expression subset	N	Clean	Missing	Noisy 10 m	Wrong
CityRefer-NO					
Urbex, with landmarks	1,302	23.5	12.7	22.3	11.3
Urbex, without landmarks	94	31.2	–	–	–
STPLS3D-Refer					
GPT-5.4, with landmarks	445	16.3	5.7	15.3	7.3
Urbex, with landmarks	445	26.3	10.7	25.8	13.9
Urbex, landmark-free diagnostic	30	23.3	–	–	–

On CityRefer-NO, 10 m coordinate noise changes landmark-guided Hit@1 by 1.2 points, while missing or wrong matches cause much larger reductions of 10.8 and 12.2 points. Accurate retrieval remains important when a landmark is needed to distinguish similar targets across a scene. The 31.2 Hit@1 on expressions without named landmarks shows that some targets can instead be identified through visual attributes and local relations; it does not imply that removing necessary landmarks from the other expressions is harmless.

All original STPLS3D-Refer expressions are landmark-grounded because both generation and filtering use OSM cues. To probe this construction bias, we re-annotate 30 targets without named landmarks and manually verify that each expression uniquely identifies its target among same-class instances. Urbex reaches 23.3 Hit@1 on this separate diagnostic. On the original 445 expressions, it exceeds GPT-5.4 under each matched perturbation, although missing and wrong landmarks substantially degrade both models. These results provide limited evidence of landmark-free transfer; a 30-example targeted set does not establish general robustness on an independently constructed, landmark-independent benchmark.

D.4 Grounding with Predicted Instance Proposals

A practical ranking pipeline can replace ground-truth candidates with predicted instances. We train SoftGroup [49] on 15 non-CityRefer SensatUrban scenes and use its proposals as CityAnchor's candidate pool. The reported CityRefer-NO proposal recall is 35.6% at IoU ≥ 0.25. Table 11 reports the resulting pipeline alongside the GT-candidate reference and proposal-free Urbex.

Table 11: CityRefer-NO proposal comparison. All metrics, including overlap and point F1, are percentages. Hit@k uses method-dependent candidate matching; direct IoU/F1 assess localization without a GT candidate pool.

Method	Proposals	Hit@1	Hit@3	Hit@5	BEV IoU	3D IoU	Point F1
CityAnchor	Ground truth	14.2	22.4	31.0	–	–	–
CityAnchor	SoftGroup	6.5	13.0	15.0	4.2	2.4	4.7
Urbex	None	23.5	37.0	40.9	14.6	7.7	17.8

The proposal pipeline has substantially lower direct localization quality than Urbex: BEV IoU is 4.2 versus 14.6, 3D IoU is 2.4 versus 7.7, and point F1 is 4.7 versus 17.8. Its Hit@1/3/5 are 6.5/13.0/15.0, compared with 14.2/22.4/31.0 for CityAnchor supplied with GT candidates. Predicted candidates lack GT instance identities, so their success requires geometric matching; these Hit@k values should not be conflated with the GT-identity protocol. The direct IoU/F1 comparison provides complementary evidence that Urbex avoids the proposal-quality bottleneck. It does not imply that all proposal generators would have the same recall.

D.5 Candidate Matching and Box-Extent Diagnostics

For a free-form prediction $\hat{b}$ and GT candidate boxes $\{b_i\}_{i=1}^{N}$, standard With-GT evaluation ranks candidates in descending order of $\mathrm{IoU}_{\mathrm{BEV}}(\hat{b}, b_i)$. A hit occurs when the target's identity is among the top k. GT boxes are used only for this post-inference scoring; Urbex neither selects from them nor receives them as input. Ranking alone imposes no minimum target overlap, so it primarily measures identification rather than precise extent.

Sensitivity to the ranking rule. Table 12 re-scores predictions from the five decoding runs on CityRefer-NO using the original IoU rule, a variant that excludes zero-overlap candidates, and CIoU, which adds center-distance and aspect-ratio penalties. Hit@1 varies by at most 0.29 points and Hit@3 by at most 0.22. This supports stability of identification under the tested ranking rules. Positive-overlap filtering removes zero-IoU ties, but it does not directly diagnose oversized boxes; stable ranking alone cannot establish tight localization.

Table 12: Matching-rule sensitivity on CityRefer-NO (five-run mean, %).

Matching rule	Hit@1	Hit@3	Hit@5
Original IoU	23.53	36.97	40.86
IoU with positive-overlap filtering	23.53	36.87	40.49
CIoU	23.24	36.75	43.18

Target-IoU-thresholded Hit@1. We additionally require the target to rank first and the free-form box to satisfy $\mathrm{IoU}_{\mathrm{BEV}}(\hat{b}, b^\star) \geq \eta$, for $\eta \in \{0.25, 0.5\}$:

$$\mathrm{Hit@1}_\eta = \frac{100}{Q} \sum_{j=1}^{Q} \mathbf{1}\left[\mathrm{rank}_j(o_j^\star) = 1\right] \mathbf{1}\left[\mathrm{IoU}_{\mathrm{BEV}}(\hat{b}_j, b_j^\star) \geq \eta\right]. \tag{6}$$

Here Q is the number of test queries. Table 13 shows that imposing 0.25 target IoU reduces Hit@1 by 4.82 points on NO and 5.24 points on ND. Loose extents therefore have a measurable effect. These are thresholded Hit@1 diagnostics, not thresholded Hit@3/5 results.

Table 13: Target-IoU-thresholded Hit@1 (%). The target must rank first under the original IoU matching rule.

Split	No IoU threshold	Target IoU ≥ 0.25	Target IoU ≥ 0.5
CityRefer-NO	23.53	18.71	11.46
CityRefer-ND	25.62	20.38	12.61

Candidate overlap and area ratio. We count candidates overlapped by each free-form prediction, first at any positive IoU and then at IoU ≥ 0.25 (Table 14). The mean number of positive overlaps is 6.5 on NO and 4.6 on ND, but the median number reaching 0.25 IoU is one on both splits. Thus, multiple weak overlaps are common, whereas a typical prediction does not strongly overlap multiple candidates. The predicted-to-target area ratio $\mathrm{area}(\hat{b})/\mathrm{area}(b^\star)$ has medians of 1.40 and 1.56, with P90 values of 9.92 and 7.21 (Table 15). This reveals a tail of oversized predictions, particularly consequential for small targets. These diagnostics support separating target identification from box quality; they do not establish that oversizing is absent.

Table 14: Number of candidate boxes overlapped per prediction.

	IoU > 0		IoU ≥ 0.25	
Split	Mean	Median	Mean	Median
CityRefer-NO	6.5	4	0.93	1
CityRefer-ND	4.6	4	0.82	1

Table 15: Predicted-to-target BEV area ratio (unitless).

Split	Median	P75	P90
CityRefer-NO	1.40	3.39	9.92
CityRefer-ND	1.56	3.48	7.21

D.6 City-Scale Search and Disambiguation Failures

We analyze two factors that complicate city-scale grounding: the distance between the target and its landmark, and the number of same-class distractors within 50 m. Table 16 reports the trajectory analysis, visualized in Figure 9. Regional-search success measures whether the selected Zoom-in region reaches the target area before local disambiguation. Local-disambiguation success measures correct identification among nearby instances after the correct area is reached.

Table 16: Reported trajectory diagnostics (success, %). The two panels measure different stages of grounding.

Target–landmark distance *Regional-search success*	ND	NO	STPLS3D
$<$ 50 m	40.0	34.3	42.2
50–100 m	31.1	32.6	26.6
100–200 m	27.3	29.1	21.7
$\geq$ 200 m	14.3	22.5	10.0
Same-class distractors within 50 m *Local-disambiguation success*	**ND**	**NO**	**STPLS3D**
0	64.7	56.7	29.5
1–3	43.1	39.1	31.2
4–9	41.8	36.5	2.4
$\geq$ 10	25.1	14.4	0.0

Regional-search success decreases with landmark distance on all three datasets. A distant landmark gives a less precise cue when an expression describes qualitative spatial relations rather than a metric displacement. Local-disambiguation success also falls markedly in dense same-class neighborhoods: on ND it drops from 64.7% without nearby same-class distractors to 25.1% with at least ten. The STPLS3D trend is not monotonic in the first two bins (29.5% versus 31.2%), but the dense bins are substantially harder. These are associations across query groups; they do not isolate a causal effect of distance or density, and no bin-level uncertainty is reported.

These factors help explain the ND Hit@5 gap. CityAnchor can keep several plausible GT candidates in its top five, including a target that Urbex misses during regional search. Even after reaching the correct region, Urbex can confuse nearby repeated instances. This is consistent with search and disambiguation errors rather than evidence of a fixed visual-recognition ceiling. Adaptive initial crops, distractor-aware verification, and boundary refinement target different parts of this failure process.

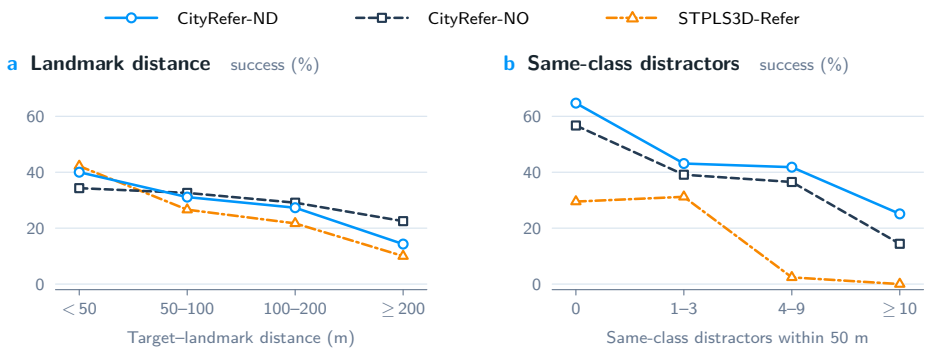


Figure 9: Aggregate trajectory diagnostics from Table 16. Left: success versus target–landmark distance. Right: success versus same-class distractor count within 50 m. These curves visualize the reported bin rates, not individual failure trajectories; per-bin sample counts and error bars are unavailable.

Representative failure trajectories. Figure 10 shows two saved CityRefer-NO trajectories: a zoom that excludes the target and a wrong-row prediction despite a zoom containing the target. In both cases, the agent commits the same box after one oblique inspection. These selected zero-IoU examples illustrate distinct failure stages; they do not estimate failure frequencies or establish the causes of the aggregate trends.

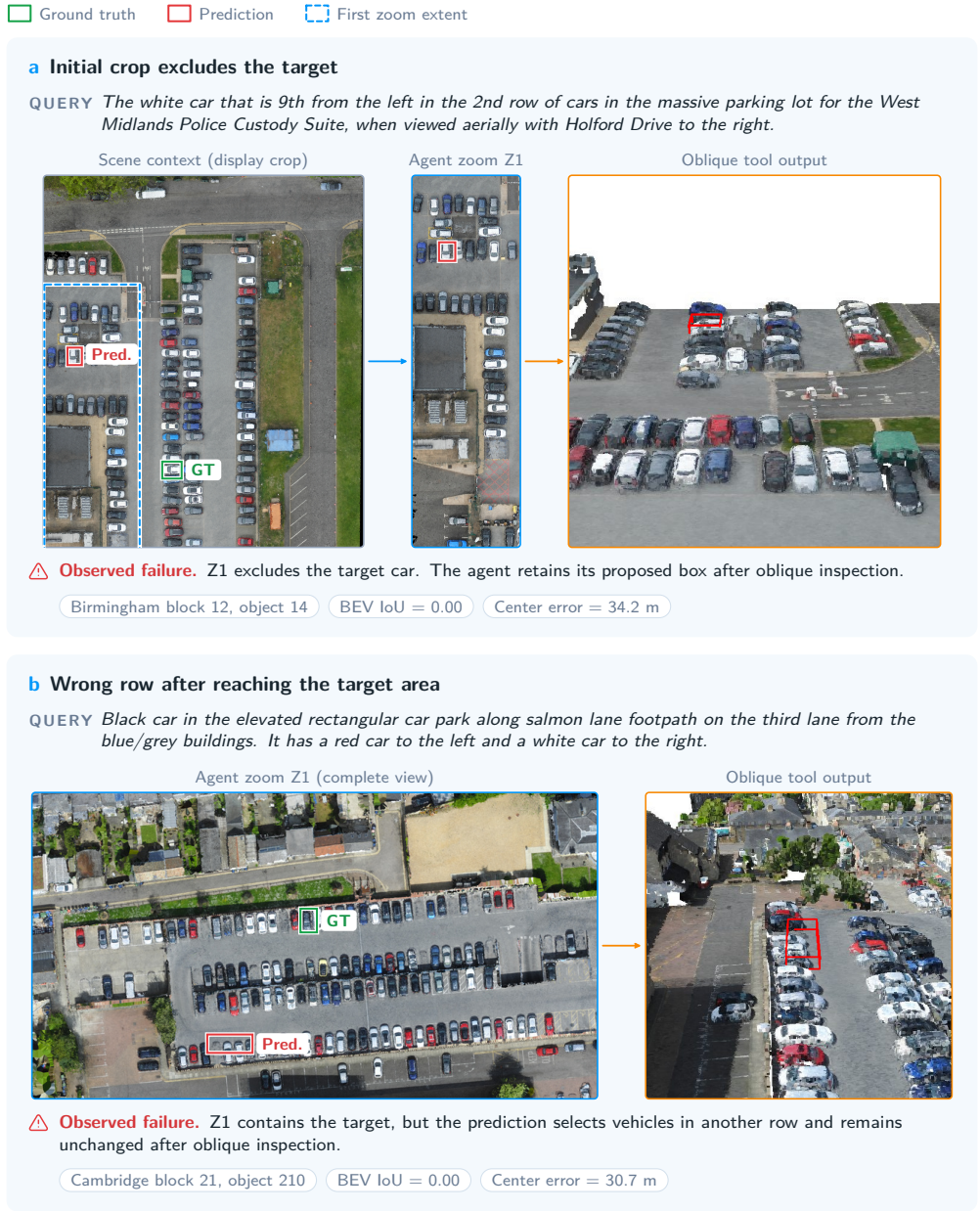


Figure 10: Qualitative failure trajectories on CityRefer-NO. (a) The first zoom excludes the annotated target. (b) The zoom includes the target, but the prediction selects vehicles in a different parking row. Scene context in (a) is an offline display crop of the high-resolution BEV. Zoom and oblique panels preserve the complete recorded tool images; GT and prediction overlays on BEV views are added for diagnosis. Green denotes GT, red denotes the prediction, and dashed blue denotes the first zoom extent. Both runs retain their proposed box after oblique inspection and obtain zero BEV IoU.

D.7 Inference efficiency

Table 17 separates online tool costs, full-query latency, and the one-time oblique-cache construction cost. Cached Landmark Query, Zoom-in, and Oblique Rendering together take about 0.36 s, included in Urbex's 3.2 s/query. CityAnchor's 32.1 s/query also excludes offline preparation and uses pre-extracted point-cloud and landmark features. The approximately $10\times$ ratio therefore compares warm online inference under the same accounting scope.

Table 17: CityRefer latency and optional scene-cache preparation. The on-demand total is an estimate; cache construction covers all 12 scenes.

Component	Latency / preparation time
Landmark Query	< 1 ms
Zoom-in from high-resolution BEV	70 ms
Oblique Rendering from cache	292 ms
All three tools	≈ 0.36 s
Urbex, full model and cached tools	3.2 s/query
CityAnchor, warm online inference	32.1 s/query
Optional oblique cache, all 12 scenes	≈ 2.3 hours (one time)
Additional latency with on-demand oblique views	≈ 1.2 s/query
Urbex, estimated on-demand total	≈ 4.4 s/query

The cache primarily makes RL rollouts efficient and is optional at inference. Rendering oblique views on demand increases the estimated total to 4.4 s/query, still about $7.3\times$ faster than CityAnchor's reported online latency. The 2.3-hour figure covers oblique-cache construction only, not the complete deployment pipeline: BEV generation, view-to-world calibration, and landmark registration are additional scene-preparation steps whose individual construction times are not reported. These costs depend on the scene and are amortized over queries; neither online ratio should be interpreted as an end-to-end deployment speedup.

E Limitations

Urbex depends on a task-specific interactive environment: reliable BEV rendering and view-to-world calibration. Errors in calibration or rendering can project a plausible 2D box to the wrong world location, while absent, ambiguous, outdated, or missing landmarks force the agent to rely more on visual search and local context, reducing efficiency.

The current policy submits a BEV-aligned bounding box, which is efficient and compatible with city-scale search but still leaves room for improving fine-grained 3D segmentation accuracy. A natural extension is to expose multi-view 3D instance segmentation models as additional tools, allowing the agent to refine the final prediction from a localized 2D region into a more accurate 3D instance mask or box.

Broader impacts and mitigation. Beyond beneficial applications such as navigation, urban planning, and accessibility, accurate language-driven localization at city scale risks misuse for surveillance, persistent target tracking, sensitive-site identification, or automated monitoring. We mitigate these risks by training only on public benchmarks (CityRefer, STPLS3D) without personal identifiers, releasing derived data under the terms of the source datasets, and recommending privacy-preserving preprocessing (e.g., blurring faces and license plates) for downstream adaptations.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes].

   Justification: The abstract and introduction state the proposed agentic formulation, interactive tools, RL training, and the main empirical claims on CityRefer and STPLS3D.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes].

   Justification: Appendix E discusses dependence on the task-specific interactive environment (BEV rendering, view-to-world calibration, cached oblique renderings, and landmark name availability), the limitations of the current 2D-box submission relative to fine-grained 3D segmentation, and broader impacts including surveillance, tracking, and sensitive-site risks together with mitigation measures.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A].

Justification: The paper is empirical and does not present formal theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes].

Justification: The main paper describes the method, reward, evaluation protocol, and baselines, while the appendix provides implementation details including the base model, GRPO hyperparameters, interaction budget, decoding settings, and hardware.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes].

   Justification: The code, trained checkpoint, and processed data are publicly available at https://github.com/AMAP-ML/Urbex, whose README gives the environment setup and the commands for downloading the data and model, training, and evaluation. The original SensatUrban point clouds are not redistributed and should be obtained from the official SensatUrban release.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes].

   Justification: The main algorithm and reward are specified, and the appendix reports the base checkpoint, batch sizes, GRPO group size, learning rate, KL coefficient, training steps, decoding settings, interaction budget, and hardware.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes].

   Justification: Appendix D.1 reports Hit@1 means and sample standard deviations across five decoding runs of one checkpoint, with an approximate 95% t-interval for ND. These quantify inference-time sampling variation on fixed test sets, not independent training-seed variability or a paired significance test against baselines.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes].

   Justification: Appendix A.1 specifies the compute resources, namely 16 NVIDIA H20 GPUs for training and a single NVIDIA H20 GPU for inference and timing, along with the optimization steps, batch sizes, and online-inference timing protocol used to reproduce the reported numbers.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes].

   Justification: The work uses publicly released urban 3D scene datasets (CityRefer, STPLS3D) for localization research and does not involve personal identification or human-subject experimentation, in line with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes].

    Justification: The introduction motivates the work by positive applications such as embodied navigation, autonomous driving, and aerial inspection, and Appendix E contains a dedicated broader-impacts paragraph that discusses negative societal risks including surveillance, persistent target tracking, sensitive-site identification, and automated city-scale monitoring, together with mitigation measures.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes].

    Justification: The paper does not release a high-risk generative model or scraped dataset, but to address misuse concerns at city scale, Appendix E describes the measures taken: training only on public benchmarks without personal identifiers, releasing derived data under the terms of the source datasets, and recommending privacy-preserving preprocessing for downstream users (e.g., blurring license plates and faces).

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes].

Justification: All external assets used in this work, including CityRefer [7] and STPLS3D [48], are properly cited and used in accordance with their original licenses; baselines and pretrained models are likewise credited to their authors and used under their respective release terms.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes].

Justification: The released code, checkpoint, and data are documented in the repository README and the Hugging Face model and dataset cards, which cover installation, data sources, training, evaluation, and licenses; the appendix documents the training configuration, evaluation protocol, and dataset construction.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A].

Justification: The experiments do not involve crowdsourcing or human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A].

    Justification: The research does not involve human subjects and therefore does not require IRB approval.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes].

    Justification: The method explicitly uses VLMs as the agent policy and discusses their role in the abstract, introduction, and method sections.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.