CITYGRASP: AGENT DRIVEN GEO-REASONING WITH ADAPTIVE SPATIAL PROGRAMMING FOR OPEN-VOCABULARY URBAN 3D GROUNDING

Anonymous authors
Paper under double-blind review

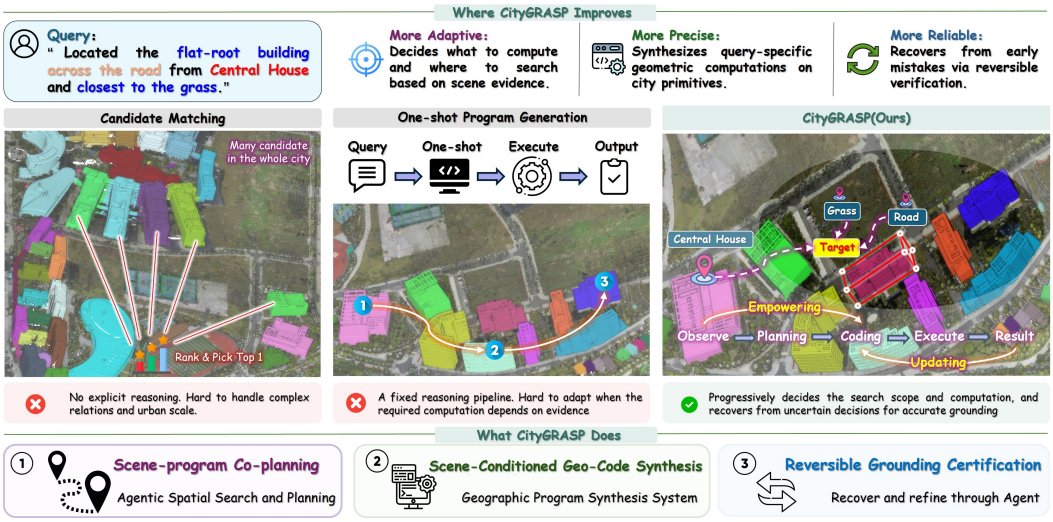

Figure 1: CityGRASP grounds city-scale 3D scenes through adaptive spatial programming, refining search and computation with execution evidence while keeping uncertain decisions recoverable.

ABSTRACT

Open-vocabulary city-scale 3D grounding localizes language-specified objects and regions in large urban scenes beyond predefined categories. However, the appropriate search scope and geometric computation often become clear only as contextual references and local geometry are resolved, while premature filtering can exclude the target. **We formulate city-scale grounding as adaptive spatial programming, in which spatial search and geometric computation evolve jointly with execution evidence and uncertain decisions remain recoverable.** We present **CityGRASP**, an agent-driven geo-reasoning framework that instantiates this formulation. Starting from query-guided open-vocabulary entity memory, **progressive scene–program co-planning** adapts the search scope and subproblem order as contextual references are resolved. **Scene-conditioned geo-code synthesis** composes low-level point-cloud and geographic primitives into computations tailored to local geometry, returning execution evidence to guide subsequent planning. To correct premature exclusions, **reversible grounding certification** checks query requirements and competing hypotheses, directs evidence revision, and tracks dependencies to restore affected alternatives while reusing valid preceding computations. Experiments on CityRefer, CityAnchor, and CitySTAR-3D demonstrate effective target disambiguation amid visually similar objects and complex spatial relations, with absolute Acc@0.50 gains of up to 17.73% over the state of the art.

1 INTRODUCTION

3D visual grounding localizes entities in a 3D scene from natural-language descriptions (Chen et al., 2020). Advances in 2D and indoor grounding (Deng et al., 2021; Zhan et al., 2023; Cao et al., 2024;

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Liu et al., 2026; Li et al., 2025c; Huang et al., 2025; Liu et al., 2025) have extended to urban point clouds (Miyanishi et al., 2023; Li et al., 2025a), where large candidate spaces, repeated structures, and spatially dispersed contextual entities complicate localization. Urban descriptions distinguish visually similar targets through landmarks and spatial relations spanning contextual entities. Consider *"the flat-roof building across the road from Central House and closest to the grass."* Resolving the landmark determines where to search, while inspecting the nearby road determines how to evaluate the spatial relation. Both decisions depend on contextual evidence that may initially be unavailable.

CityRefer (Miyanishi et al., 2023) incorporates geographic context into candidate matching, while CityAnchor (Li et al., 2025a) narrows search through coarse-to-fine localization. Programmatic approaches make spatial constraints executable (Yuan et al., 2024b;a); LaSP (Mi et al., 2025) generates relation encoders before grounding, and GeoProg3D (Yasuki et al., 2025) composes geographic APIs into city-scale reasoning programs. The concurrent CitySTAR framework (Zhang et al., 2026b) organizes urban grounding through tool-derived evidence, paired-hypergraph topology reasoning, and candidate-centered visual verification. These advances establish the value of executable spatial reasoning and cross-modal verification. A remaining challenge is to coordinate changes in search scope, geometric computation, and earlier decisions as new evidence becomes available. Scoring or verifying retained candidates alone cannot recover previously excluded target.

Our key insight is that spatial search and geometric computation should evolve together with execution evidence, while decisions based on uncertain evidence remain recoverable. In this example, an ambiguous landmark leaves the search region unresolved, and a road bend changes the local geometry needed to evaluate the across-road relation. Committing to either interpretation too early can propagate errors into later filtering. We formulate urban grounding as **adaptive spatial programming**: the system progressively constructs the grounding program by alternating among evidence acquisition, subproblem selection, and scene-conditioned computation. Dependency tracking identifies earlier decisions requiring revision and enables reuse of valid preceding computations during recovery.

We instantiate this formulation in **CityGRASP**, using query-guided open-vocabulary perception to initialize an updateable entity memory. First, **progressive scene–program co-planning** jointly selects the next scope and subproblem, adapting search and reasoning order as references are resolved. Second, **scene-conditioned geo-code synthesis** composes low-level point-cloud and geographic primitives into executable computations, adapting relation evaluation to the observed local geometry. Third, **reversible grounding certification** checks query requirements and competing hypotheses while retaining uncertain alternatives and evidence dependencies. When evidence is revised, CityGRASP restores affected hypotheses and reconstructs the corresponding reasoning sequence while reusing valid preceding computations. Experiments on CityRefer, CityAnchor, and CitySTAR-3D demonstrate consistent system-level accuracy gains when distinguishing similar objects under complex spatial relations. Controlled ablations on CityRefer support the proposed mechanisms; full-restart comparisons quantify selective recovery's computational savings.

Our contributions are summarized as follows:

- We introduce **progressive scene–program co-planning** to jointly select search scopes and grounding subproblems using intermediate execution evidence, adapting spatial search and reasoning order as contextual references are resolved.
- We propose **scene-conditioned geo-code synthesis** to compose low-level primitives into executable computations tailored to query constraints and local geometry, using execution results to guide subsequent planning and evidence acquisition.
- We develop **reversible grounding certification** to check query requirements and competing hypotheses, track evidence dependencies, and restore previously excluded alternatives after evidence revision while reusing valid preceding computations.

2 Related Works

Open-vocabulary 3D grounding and scene understanding. Semantic generalization and spatial search address distinct requirements in 3D grounding. Indoor benchmarks (Chen et al., 2020; Achlioptas et al., 2020) have driven contextual reasoning (Roh et al., 2022; Yang et al., 2023; Gu et al., 2024; Liu et al., 2026; 2025) and multi-view modeling (Huang et al., 2022; Yang et al., 2021; Zhang et al., 2024; Huang et al., 2025). Foundation models extend language-conditioned

3D understanding (Hong et al., 2023; Chen et al., 2024a; Fu et al., 2024; Yin et al., 2023; Tang et al., 2024; Li et al., 2024), while zero-shot approaches reduce task-specific supervision through open-vocabulary perception and visual reasoning (Yang et al., 2024; Xu et al., 2024; Li et al., 2025c; Zhang et al., 2026a; Drozdov et al., 2026; Tan et al., 2026). Broader semantic coverage, however, does not determine where to search when contextual landmarks remain unresolved. CityGRASP jointly resolves references and selects regions and contextual entities from intermediate evidence.

Programmatic and agentic 3D grounding. Executable methods differ in when and how they define spatial computations. ZSVG3D (Yuan et al., 2024b) composes predefined modules, CSVG (Yuan et al., 2024a) jointly solves target–anchor constraints; LaSP (Mi et al., 2025) generates and optimizes relation encoders before grounding. Agents gather tool and visual-inspection evidence (Yang et al., 2024; Wang et al., 2026; Huynh et al., 2026); SPAZER (Jin et al., 2026) combines anchor-guided screening and joint 3D–2D decisions; PruneGround (Cao Dinh et al., 2026) reduces irrelevant regions. These methods enable explicit reasoning and candidate reduction, yet reference resolution can change the required geometric computation. CityGRASP interleaves subproblem planning and scene-conditioned code synthesis from low-level primitives, both guided by observed geometry. SATURN (Kamali et al., 2026) composes soft spatial predicates via symbolic execution, preserving uncertainty in multi-hop reasoning. CityGRASP revises search decisions using empirical intervals for acceptance and recorded dependencies to revoke exclusions and branch closures as evidence changes.

City-scale 3D grounding and geographic reasoning. Urban grounding must combine spatial search with reasoning over extensive, heterogeneous scenes (Chen et al., 2019; Kolmet et al., 2022). CityRefer (Miyanishi et al., 2023) incorporates geographic context, CityAnchor (Li et al., 2025a) narrows search through coarse-to-fine localization, and 3EED (Li et al., 2025b) integrates heterogeneous RGB–LiDAR observations. Urban vision-language models further support scene understanding (Chen et al., 2024b; Sun et al., 2025; Chen et al., 2026a;b). GeoProg3D (Yasuki et al., 2025) generates Python programs composing provided geographic APIs for region selection and measurement over georeferenced 3D Gaussian language fields. CitySTAR (Zhang et al., 2026b) combines CodeLLM-driven evidence acquisition, paired-hypergraph topology reasoning, and reflective cross-modal verification. When early filtering excludes the target, however, verification restricted to retained candidates cannot recover it. CityGRASP preserves deferred hypotheses and evidence dependencies to restore alternatives and selectively recompute affected decisions.

3 METHOD

3.1 OVERVIEW

Problem definition. Given a city-scale scene $\mathcal{S}$, comprising a point cloud and available geographic metadata, and a natural-language query q, we predict the referred entity's axis-aligned 3D box:

$$\hat{B}_q = \mathcal{G}(\mathcal{S}, q), \qquad \hat{B}_q = (\hat{\mathbf{c}}, \hat{\mathbf{s}}) \in \mathbb{R}^3 \times \mathbb{R}^3_{>0}, \tag{1}$$

where $\hat{\mathbf{c}}$ and $\hat{\mathbf{s}}$ denote its center and side lengths. Geographic metadata supplies contextual information for named landmarks and roads; target ground-truth annotations are excluded from inference.

Framework. Figure 2 presents four interacting stages. Query-guided parsing maintains entity memory; co-planning selects search scopes and subproblems; geo-code synthesis produces scene-conditioned evidence; and certification checks hypotheses and directs revision. Execution evidence informs subsequent planning, while recorded dependencies identify which decisions require recovery when their supporting inputs change.

3.2 STAGE 0: QUERY-GUIDED OPEN-VOCABULARY SCENE PARSING

We parse q into $\mathcal{Q}_q = (c_q, \mathcal{T}_q, \mathcal{H}_q, \mathcal{R}_q)$: the target concept, entity-bound attributes, contextual references, and spatial requirements, preserving reference frames and selection domains. Query-guided masks from bird's-eye and oblique views are lifted through point-ID buffers and merged into entity memory $\mathcal{M}_0 = \{o_i\}$, storing geometry, boxes, semantic observations, and geographic context. Further observations add missing entities or revise existing representations. A hypothesis $h = (i, \mathbf{b})$ pairs target entity i with reference bindings $\mathbf{b}$. Alternative bindings retain separate evidence. Observations, geometry, and bindings are versioned to support dependency tracking and recovery.

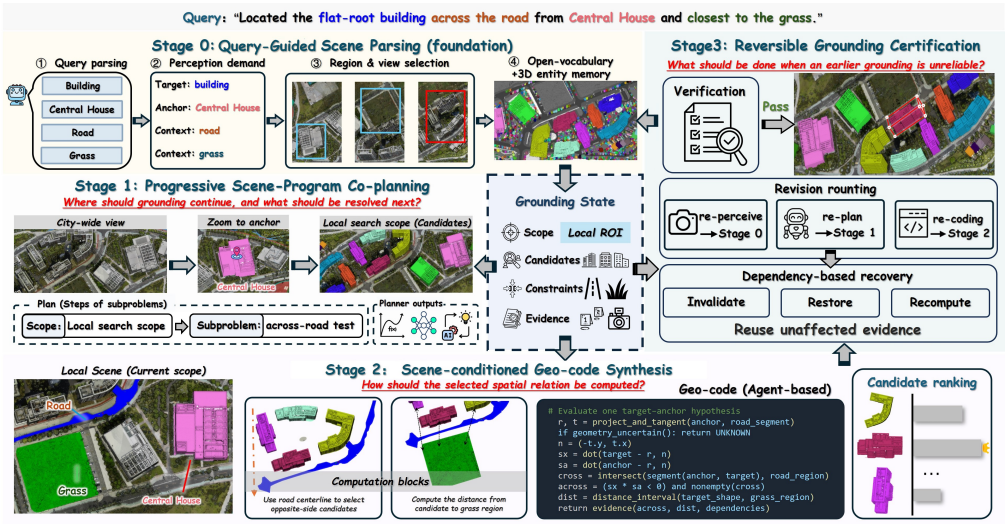

Figure 2: **Overview of CityGRASP.** The framework constructs query-relevant entities, co-plans scopes and subproblems, and synthesizes scene-conditioned geo-code for spatial reasoning. Reversible certification verifies grounding evidence and guides revision and recovery.

3.3 Stage 1: Progressive Scene–Program Co-Planning

State $\mathcal{Z}_t = (\mathcal{M}_t, \Omega_t, \mathcal{C}_t, \mathcal{D}_t, \mathcal{F}_t, \mathcal{U}_t, \mathcal{E}_t)$ stores entity memory, current scope, active/deferred hypotheses, unexplored branches, unresolved requirements, and valid evidence. The LLM selects

$$\mathbf{p}_t = \Phi(\mathcal{Q}_q, \mathcal{Z}_t) = (\Omega_t^{\text{next}}, g_t, \mathcal{H}_t^{\text{req}}, \sigma_t), \tag{2}$$

specifying next scope, subproblem, required references, and contract σ_t. The contract types inputs and measurement outputs, separating candidates from context. Across-road evaluation maps target/reference geometry and a road section to signed lateral offsets and road-intersection evidence. Unresolved requirements and execution evidence guide observations, reference resolution, or Stage 2 computation. Resolving Central House localizes search; missing road geometry prompts observation before across-road evaluation. Deterministic checks validate input availability and contracts; missing inputs remain unresolved, not negative evidence. Execution updates state before replanning. Scope restriction defers, without disproving, hypotheses to $\mathcal{D}_t$. Unprocessed scopes/bindings stay in $\mathcal{F}_t$; exploration adds hypotheses to represented sets. Closure requires recorded evidence of no admissible target; retained exclusion/closure dependencies enable hypothesis restoration and branch reopening upon evidence revision.

3.4 Stage 2: Scene-Conditioned Geo-Code Synthesis

The geo-coder composes typed geometry-retrieval, projection, direction-estimation, distance, intersection, and set primitives $\mathcal{B}$ into $P_t = \Psi(\mathbf{p}_t, \mathcal{Z}_t, \mathcal{B})$ under contracts, local geometry, bindings, and feedback, fetching dense geometry at runtime. Scene conditioning selects geometric inputs, local frames, and primitive compositions within the query contract; relation semantics/support scales stay fixed. Across-road code estimates road-section tangent $\mathbf{v}$ beside the reference. With planar normal $\mathbf{n} = (-v_y, v_x)$, road point $\mathbf{r}$, and target/reference positions $\mathbf{x}, \mathbf{a}$, require $[(\mathbf{x}-\mathbf{r})^\top \mathbf{n}][(\mathbf{a}-\mathbf{r})^\top \mathbf{n}] < 0$, and $[\mathbf{x}, \mathbf{a}]$ intersecting the road footprint. Curvature sets the local frame; select branches at junctions.

Execution checks syntax, permitted calls, geometry, and output types; errors prompt repair; missing/degenerate inputs stay unresolved. For necessary non-selection constraints $j \in \mathcal{J}_q$ (concepts, attributes, relations), record

$$\xi_{j,t}(h) = \big(\rho_{j,t}(h), [\ell_{j,t}(h), u_{j,t}(h)], e_{j,t}(h), \kappa_{j,t}(h)\big), \tag{3}$$

comprising normalized support, empirical bounds, measurements, and dependencies. Geometric support is $\rho_j = \text{clip}(1/2 + m_j/(2s_j), 0, 1)$ for signed satisfaction margin m_j (e.g., $r - d$ for $d \le r$) and fixed relation-specific scale $s_j > 0$. Repeated point subsampling without replacement recomputes geometry with fixed programs/bindings; $[\ell, u]$ spans support over subsamples plus nominal input.

Selection relations retain physical measurements and resampling intervals. Visual support uses sigmoid-calibrated Yes–No log-probability differences, fitted on labeled candidate–constraint pairs from test-disjoint development scenes. Prescribed-view means/extrema give $\rho[\ell, u]$; unavailable evidence retains $[0, 1]$. These empirical sensitivity intervals summarize support variation across prescribed geometric subsamples or visual views under the current computation; reference bindings are assessed separately. Calibration parameters, subsampling settings, and relation scales are fixed before testing without refitting. Dependencies record observations, geometry, bindings, program versions, and input hypothesis sets.

3.5 Stage 3: Reversible Grounding Certification

Stage 3 verifies execution evidence to accept hypotheses or direct revision. For $0 \leq \tau_{\text{rej}} < 0.5 < \tau_{\text{sup}} \leq 1$, constraint j is supported if $\ell_{j,t}(h) \geq \tau_{\text{sup}}$, contradicted if $u_{j,t}(h) \leq \tau_{\text{rej}}$, and unresolved otherwise. Qualification requires all $j \in \mathcal{J}_q$ supported under consistent bindings. Acceptance additionally requires $\mathcal{F}_t = \varnothing$: every represented branch is explored or closed by evidence of no admissible target. Scope restriction or missing evidence cannot justify closure. Competitors $\mathcal{P}_t(\hat{h}_t)$ include active/deferred hypotheses with other targets, excluding contradicted hypotheses and, for selection queries, those reliably outside the comparison domain. Unresolved alternatives remain eligible. Rejecting one reference assignment does not eliminate other assignments for the same target.

For non-selection queries, uniform weights $w_j = 1/|\mathcal{J}_q|$ define aggregate bounds $\underline{\Gamma}_t(h) = \sum_j w_j \ell_{j,t}(h)$ and $\overline{\Gamma}_t(h) = \sum_j w_j u_{j,t}(h)$. After qualification and coverage checks, acceptance requires separation between these bounds. Closest-object queries instead compare physical distance intervals:

$$\underline{\Gamma}_t(\hat{h}_t) - \max_{h \in \mathcal{P}_t(\hat{h}_t)} \overline{\Gamma}_t(h) \geq \theta_{\text{cert}}, \tag{4}$$

$$\overline{d}_t(\hat{h}_t) < \min_{h \in \mathcal{P}_t(\hat{h}_t)} \underline{d}_t(h), \tag{5}$$

where $\theta_{\text{cert}} > 0$, and $\underline{d}_t, \overline{d}_t$ bound distance. Other selection operators use corresponding physical comparisons; empty maxima/minima equal $-\infty/+\infty$. Aggregate scores cannot compensate for unsupported constraints. Certification evaluates generated candidates and represented search branches using the available empirical evidence.

Failed checks route missing observations, unresolved references or coverage, and computational errors to Stages 0/1/2, respectively. Interval overlap prompts discriminating evidence without changing relation or measurement semantics. After input revision, dependencies locate the earliest invalid block. CityGRASP retains corrected inputs and the valid prefix, revokes suffix-supported exclusions and branch closures, restores affected hypotheses, and reconstructs the suffix. The retained prefix supplies reusable evidence, avoiding recomputation of valid preceding steps. Reopened branches return to the search frontier for further exploration or evidence-based closure. Revised landmark bindings thereby restore candidates excluded by invalidated spatial evidence. Restored alternatives undergo the same checks; acceptance returns the target entity's 3D box.

4 Experiments

Datasets. We evaluate on CityRefer (Miyanishi et al., 2023), CityAnchor (Li et al., 2025a), and CitySTAR-3D (Zhang et al., 2026b). CityRefer covers four categories in SensatUrban; CityAnchor covers nine in synthetic STPLS3D scenes. Their Novel Objects (NO) and Novel Descriptions (ND) settings evaluate unseen instances and new descriptions, respectively. CitySTAR-3D evaluates open-vocabulary grounding through 1,286 text–object pairs across 15 semantic groups, including specialized categories, extended regions, and multi-entity spatial descriptions.

Implementation details. CityGRASP uses four NVIDIA RTX 4090 GPUs with frozen Qwen3.5-9B for planning/code synthesis and Qwen3-VL-4B-Instruct for visual observations. Decoding temperature is 0. Bird's-eye and oblique SAM3 masks are lifted into 3D to initialize entity memory, updated by subsequent observations. We set $\tau_{\text{sup}} = 0.8$, $\tau_{\text{rej}} = 0.2$, and $\theta_{\text{cert}} = 0.05$, with uniform weights $w_j = 1/|\mathcal{J}_q|$ for necessary non-selection constraints. Further details appear in Appendix A.2.

Baselines and metrics. We rerun InstanceRefer, 3DVG-Transformer, EDA, CityRefer, CityAnchor, and CitySTAR on all evaluated datasets. CitySTAR (Zhang et al., 2026b) uses Qwen3.5-9B for reasoning and Qwen3-reranker-4B for visual verification. Both CitySTAR and CityGRASP use SAM3-based candidate construction, but their observation and reasoning pipelines differ. Table 1 therefore evaluates complete systems, with gains reflecting both perception and reasoning. Table 2 examines component contributions within CityGRASP using shared initial entities and fixed backbone models. Acc@0.25 and Acc@0.50 measure the percentages of predictions whose box IoU meets each threshold. Accuracy changes are absolute differences; cost reductions are relative. Section 3.5 governs prediction acceptance; accuracy is measured against ground truth over the full evaluation set.

4.1 Grounding Performance

Table 1 shows that CityGRASP achieves the highest accuracy across all five settings at both thresholds. Compared with CitySTAR, Acc@0.50 improves by $2.93\%/3.58\%$ on CityRefer NO/ND and $2.29\%/5.22\%$ on CityAnchor NO/ND. These gains demonstrate consistent improvements across unseen instances and new descriptions in real and synthetic urban scenes. Figures 3 and 4 illustrate how landmarks, connectivity, and neighboring entities distinguish targets with similar appearances.

Table 1: Quantitative comparison on CityRefer, CityAnchor, and CitySTAR-3D datasets. Accuracy is reported in %. NO and ND denote Novel Objects and Novel Descriptions. Best results are in bold.

Method	**CityRefer-NO**		**CityRefer-ND**		**CityAnchor-NO**		**CityAnchor-ND**		**CitySTAR-3D**	
	@0.25	@0.50	@0.25	@0.50	@0.25	@0.50	@0.25	@0.50	@0.25	@0.50
InstanceRefer	4.09	3.64	1.93	1.76	1.58	1.35	3.04	2.31	0.93	0.93
3DVG-Transformer	7.73	5.69	9.64	8.12	4.16	2.38	6.25	4.17	0.39	0.16
EDA	6.96	5.53	8.39	5.84	5.15	3.09	7.14	4.29	0.08	0.00
CityRefer	8.34	7.47	5.07	3.49	5.73	4.16	6.07	3.95	2.26	1.87
CityAnchor	50.69	46.86	53.17	50.37	41.23	35.11	47.81	43.40	22.40	20.61
CitySTAR	72.54	71.68	82.67	80.68	62.60	61.07	56.52	53.04	56.84	55.75
CityGRASP	**75.82**	**74.61**	**86.06**	**84.26**	**64.89**	**63.36**	**61.74**	**58.26**	**73.72**	**73.48**

CitySTAR-3D shows the largest improvements: 16.88% at Acc@0.25 and 17.73% at Acc@0.50, reaching 73.72% and 73.48%, respectively. The 0.24% gap indicates that nearly all predictions meeting the lower threshold also satisfy the stricter criterion; these metrics jointly reflect entity selection and box localization. Figure 5 illustrates complementary challenges. For the armored vehicle and box truck, CityGRASP resolves fine-grained appearance cues together with surrounding directions and distances. For the tree-centered region, it combines canopy, vehicle, and building references to identify the specified area. For the street-region query, its prediction covers the described street and flanking buildings, whereas competing predictions cover only a local fragment. These cases illustrate successful grounding of individual objects and extended regions under multiple contextual constraints. Additional comparisons and failure analyses appear in Appendices A.5 and A.6.

4.2 Ablation Study

Table 2 evaluates component contributions on CityRefer. All decreases below refer to Acc@0.50 on NO/ND, respectively. The complete-program variant generates an entire program before each execution attempt while retaining program checks, final verification, and recovery. Its accuracy drops by $28.32\%/34.06\%$. The accuracy drop supports conditioning program construction on intermediate execution evidence. Fixing subproblem order causes larger losses ($16.23\%/16.53\%$) than fixing scope schedules ($3.80\%/3.78\%$), showing greater sensitivity to reasoning order under these controls. Freezing entity memory reduces accuracy by $4.83\%/4.78\%$, supporting continued observation as contextual and geometric requirements emerge.

Replacing generated geo-code with predefined relation implementations reduces accuracy by $7.43\%/7.37\%$. Removing scene-specific evidence from generation and revision causes losses of $4.83\%/4.58\%$, despite retaining runtime geometry access. These controls support both generating computations and conditioning their construction on the scene. For example, across-road evalua-



Figure 3: Qualitative grounding comparison on CityRefer. Compared with CityRefer, CityAnchor, and CitySTAR, CityGRASP better identifies targets by appearance, landmarks, and spatial relations.



Figure 4: Qualitative grounding comparison on CityAnchor. CityGRASP better distinguishes visually similar objects using appearance and surrounding spatial context, reducing wrong-target predictions.

tion depends on the selected road section and reference frame; access to geometry alone does not determine the appropriate computation.

Replacing interval-aware verification with point-evidence decisions reduces accuracy by $3.28\%/3.18\%$, while retaining query constraints, selection rules, coverage checks, and recovery. Disabling verification-triggered revision and restoration causes losses of $4.14\%/3.98\%$, with program checks and verification retained. These controls support uncertainty-aware decisions and

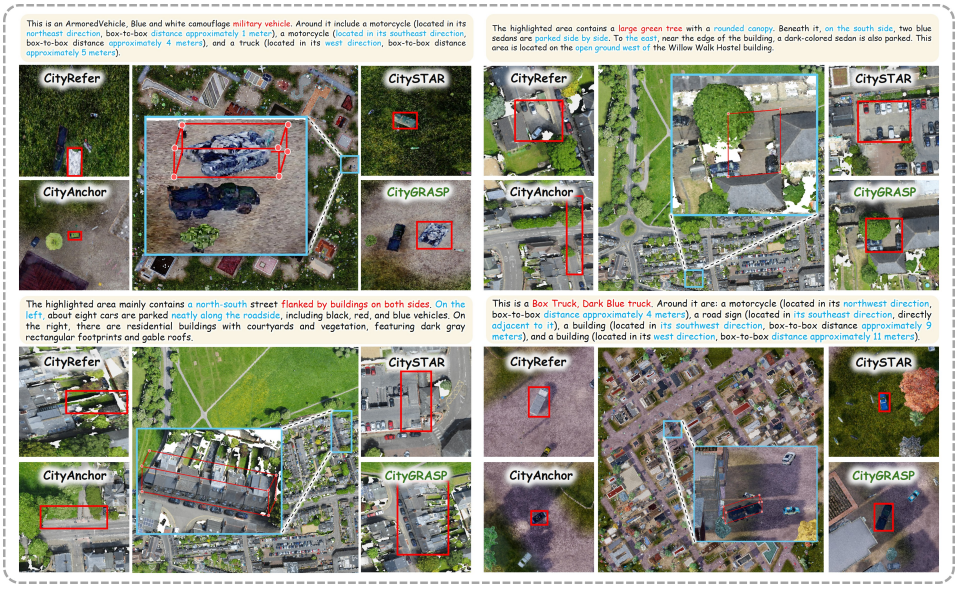

Figure 5: Qualitative grounding comparison on CitySTAR-3D. CityGRASP better localizes specialized objects and extended regions from detailed attributes and multi-entity spatial constraints.

Table 2: Ablation of CityGRASP components on CityRefer. Grounding accuracy (%) is reported for NO and ND. Parentheses show absolute accuracy drops from full CityGRASP.

Variant	CityRefer-NO		CityRefer-ND	
	Acc@0.25	Acc@0.50	Acc@0.25	Acc@0.50
Stage 0: Query-guided scene parsing				
w/o Entity Memory Updates	71.33 (↓4.49)	69.78 (↓4.83)	81.87 (↓4.19)	79.48 (↓4.78)
Stages 1–2: Progressive program construction				
w/o Progressive Synthesis	46.98 (↓28.84)	46.29 (↓28.32)	52.79 (↓33.27)	50.20 (↓34.06)
Stage 1: Scene–program co-planning				
w/o Adaptive Scope Selection	72.37 (↓3.45)	70.81 (↓3.80)	82.87 (↓3.19)	80.48 (↓3.78)
w/o Adaptive Subproblem Ordering	59.93 (↓15.89)	58.38 (↓16.23)	69.92 (↓16.14)	67.73 (↓16.53)
Stage 2: Scene-conditioned geo-code synthesis				
w/o Geo-Code Synthesis	68.74 (↓7.08)	67.18 (↓7.43)	79.28 (↓6.78)	76.89 (↓7.37)
w/o Scene Conditioning	71.50 (↓4.32)	69.78 (↓4.83)	82.07 (↓3.99)	79.68 (↓4.58)
Stage 3: Reversible grounding certification				
w/o Uncertainty-Aware Verification	72.88 (↓2.94)	71.33 (↓3.28)	83.47 (↓2.59)	81.08 (↓3.18)
w/o Reversible Recovery	72.19 (↓3.63)	70.47 (↓4.14)	82.67 (↓3.39)	80.28 (↓3.98)
Full CityGRASP	**75.82**	**74.61**	**86.06**	**84.26**

verification-triggered revision, respectively. Figure 6 shows how revised reference evidence restores a previously excluded target for renewed verification.

4.3 Efficiency and Recovery

Table 3 reports accuracy and mean per-query inference cost over 1,081 pooled CityRefer NO/ND queries. Costs include failed attempts but exclude reusable scene preprocessing; token counts include both inputs and outputs. The CityGRASP variants use identical hardware and execution settings. Full Restart retains revised inputs but recomputes grounding; Selective Recovery retains the valid prefix and reconstructs the affected suffix.

Selective Recovery reduces runtime from 43.5 to 24.1 seconds, model calls from 28.6 to 15.3, and tokens from 121k to 63k, corresponding to reductions of 44.6%, 46.5%, and 47.9%. It correctly grounds 855 queries versus 832 for Full Restart, a net gain of 23. CityGRASP gains 3.23% accuracy over CitySTAR at higher runtime, model-call, and token costs, demonstrating an accuracy–cost trade-off.

Table 3: Accuracy (%) and per-query inference cost on CityRefer. Parentheses indicate absolute gains over CitySTAR. Costs include failed attempts but exclude scene preprocessing.

Method	Acc@0.50	Time (s)	Model Calls	Tokens (k)
CitySTAR	75.86	11.6	7.4	24
CityGRASP (Full Restart)	76.97 (↑1.11)	43.5	28.6	121
CityGRASP (Selective Recovery)	**79.09** (↑3.23)	24.1	15.3	63

Figure 6 shows a real execution case involving road-end, along-street, and opposite-building constraints. The initial opposite-building computation relies on an incorrect Primark binding and excludes candidate B. Verification prompts reconsideration of this reference, and the revised binding invalidates the evidence supporting the exclusion. CityGRASP identifies the earliest affected computation, restores B to the comparison set, and reconstructs the affected suffix while preserving the valid execution prefix. The recomputed opposite-building evidence supports B, which is then verified against the target concept, the remaining spatial requirements, and competing hypotheses. Candidate B passes these checks and becomes the final target under the revised reference binding, with valid preceding computations reused throughout recovery.

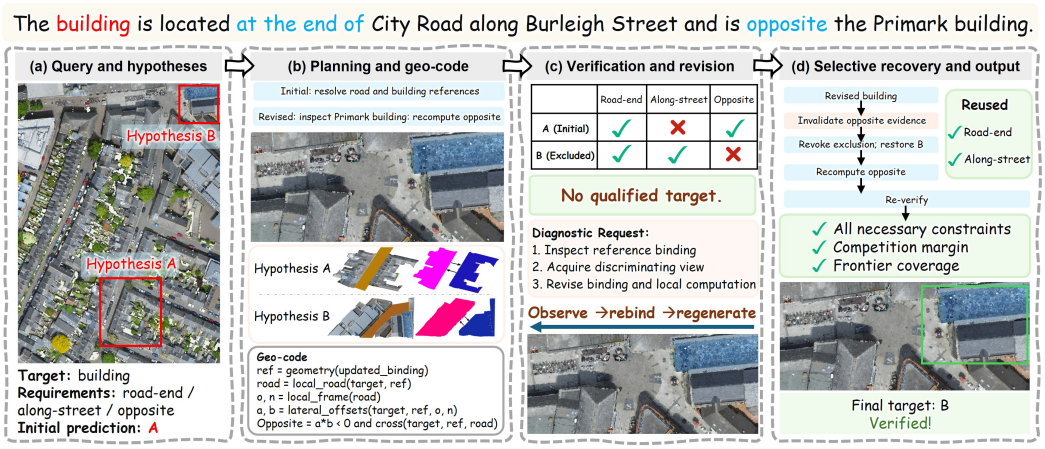

Figure 6: Selective recovery in CityGRASP. Revising the reference binding invalidates the spatial evidence used to exclude candidate B. CityGRASP restores B and reconstructs the affected suffix while preserving the valid execution prefix, yielding a verified grounding result.

5 Limitation and Conclusion

Limitation. CityGRASP relies on external perception for entity coverage and geometric quality, making occluded and sparsely observed regions challenging. Its empirical intervals characterize sampling and view sensitivity rather than semantic correctness. Experiments establish system-level gains on three benchmarks and component contributions on CityRefer; broader evaluation would clarify cross-dataset generalization and sensitivity to perception and verification settings. Selective recovery reuses valid computations, but complex queries or repeated verification failures require additional planning, evidence acquisition, and program revision, increasing token consumption.

Conclusion. We presented CityGRASP, an agent-driven framework for open-vocabulary city-scale 3D grounding through adaptive spatial programming. Progressive scene–program co-planning adapts search scopes and subproblems to execution evidence, while scene-conditioned geo-code synthesis composes geometric primitives into computations tailored to local geometry. Reversible grounding certification checks query requirements and competing hypotheses, guiding revision and restoring affected alternatives while preserving valid evidence. Together, these mechanisms make the grounding procedure itself revisable as evidence changes. This unifies adaptive search, geometric program synthesis, and selective recovery within a single executable inference loop. Experiments on CityRefer, CityAnchor, and CitySTAR-3D demonstrate consistent accuracy gains and effective target disambiguation in scenes with visually similar objects and complex spatial relations.

AI USE STATEMENT

We used generative AI tools to assist with manuscript drafting and revision, language editing, LaTeX preparation, figure-prompt development, and cross-section consistency checks. AI tools also supported discussions of methodological descriptions and interpretations of experimental results. AI-generated suggestions were treated as draft material requiring author verification, not as experimental evidence or independent validation of mathematical or algorithmic claims. The foundation models used within CityGRASP are components of the evaluated method, and their roles are specified in Section 4. The authors take full responsibility for the research design, implementation, experimental results, references, and conclusions, including the accuracy of all AI-assisted content retained in the final manuscript.

ETHICS STATEMENT

This work studies language-guided localization of objects and regions in urban 3D scenes using existing benchmarks. The task does not involve identifying individuals or inferring personal attributes. Nevertheless, urban imagery and point clouds may contain privacy-sensitive information, and their use and redistribution should respect the source datasets' permissions and applicable privacy requirements. Grounding errors and uneven performance across scene conditions may affect downstream applications. CityGRASP's verification mechanism checks consistency with available evidence and does not guarantee correctness; it should not be interpreted as a safety assurance for deployment in consequential settings.

REPRODUCIBILITY STATEMENT

Section 3 defines the grounding stages, evidence representations, acceptance criteria, and selective recovery procedure. Section 4 reports the datasets, baseline methods, model configuration, evaluation metrics, and experimental results. Appendix A.2 supplements the implementation settings, ablation definitions, and computational cost accounting. Appendix A.3 explains the planning state, geometric and visual evidence construction, and dependency-based recovery. Appendix A.4 provides illustrative geometric code and an explanation of the recovery case; the illustrative code is distinguished from an executable implementation or a verbatim execution trace. Additional qualitative comparisons and failure analyses appear in Appendices A.5 and A.6.

REFERENCES

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*, pp. 422–440. Springer, 2020.

Shengcao Cao, Liang-Yan Gui, and Yu-Xiong Wang. Emergent visual grounding in large multimodal models without grounding supervision. *arXiv preprint arXiv:2410.08209*, 2024.

Duc Cao Dinh, Khai Le-Duc, Florent Draye, Chris Ngo, Terry Jingchen Zhang, Bernhard Schölkopf, and Zhijing Jin. Pruneground: Plug-and-play spatial pruning for 3d visual grounding. *arXiv e-prints*, pp. arXiv–2606, 2026.

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*, pp. 202–221. Springer, 2020.

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*, pp. 12538–12547, 2019.

Jiabin Chen, Haiping Wang, Jinpeng Li, Yuan Liu, Zhen Dong, and Bisheng Yang. Spatialllm: From multi-modality data to urban spatial intelligence. *International Journal of Applied Earth Observation and Geoinformation*, 147:105177, 2026a.

Sijin Chen, Xin Chen, Chi Zhang, Mingsheng Li, Gang Yu, Hao Fei, Hongyuan Zhu, Jiayuan Fan, and Tao Chen. Ll3da: Visual interactive instruction tuning for omni-3d understanding reasoning and planning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 26428–26438, 2024a.

Yiping Chen, Shuai Zhang, Ting Han, Yumeng Du, Wuming Zhang, and Jonathan Li. Chat3d: Interactive understanding 3d scene-level point clouds by chatting with foundation model for urban ecological construction. *ISPRS Journal of Photogrammetry and Remote Sensing*, 212:181–192, 2024b.

Yiping Chen, Jinpeng Li, Wenyu Ke, Yang Luo, Jie Ouyang, Zhongjie He, Li Liu, Hongchao Fan, and Hao Wu. 3dcity-llm: Empowering multi-modality large language models for 3d city-scale perception and understanding. *arXiv preprint arXiv:2603.23447*, 2026b.

Jiajun Deng, Zhengyuan Yang, Tianlang Chen, Wengang Zhou, and Houqiang Li. Transvg: End-to-end visual grounding with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1769–1779, 2021.

Nikita Drozdov, Andrey Lemeshko, Nikita Gavrilov, Anton Konushin, Danila Rukhovich, and Maksim Kolodiazhnyi. Z3d: Zero-shot 3d visual grounding from images. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 147–154, 2026.

Rao Fu, Jingyu Liu, Xilun Chen, Yixin Nie, and Wenhan Xiong. Scene-llm: Extending language model for 3d visual understanding and reasoning. *arXiv preprint arXiv:2403.11401*, 2024.

Qiao Gu, Ali Kuwajerwala, Sacha Morin, Krishna Murthy Jatavallabhula, Bipasha Sen, Aditya Agarwal, Corban Rivera, William Paul, Kirsty Ellis, Rama Chellappa, et al. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5021–5028. IEEE, 2024.

Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 3d-llm: Injecting the 3d world into large language models. *Advances in Neural Information Processing Systems*, 36:20482–20494, 2023.

Ronggang Huang, Haoxin Yang, Yan Cai, Xuemiao Xu, Huaidong Zhang, and Shengfeng He. Viewsrd: 3d visual grounding via structured multi-view decomposition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9726–9736, 2025.

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*, pp. 15524–15533, 2022.

Cuong Huynh, Maxim Popov, Denis Gridusov, and Sergey Kolyubin. Agentgrounder: Zero-shot 3d visual pointcloud grounding using multimodal language models. *arXiv preprint arXiv:2605.25901*, 2026.

Zhao Jin, Rong-Cheng Tu, Jingyi Liao, Wenhao Sun, Xiao Luo, Shunyu Liu, and Dacheng Tao. Spazer: Spatial-semantic progressive reasoning agent for zero-shot 3d visual grounding. *Advances in Neural Information Processing Systems*, 38:165549–165576, 2026.

Danial Kamali, Tanawan Premsri, Shreya Rajpal, Amir Zadeh, Chuan Li, and Parisa Kordjamshidi. Saturn: Symbolic spatial reasoning for multi-perspective grounding. *arXiv preprint arXiv:2606.22694*, 2026.

Manuel Kolmet, Qunjie Zhou, Aljoša Ošep, and Laura Leal-Taixé. Text2pos: Text-to-point-cloud cross-modal localization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6687–6696, 2022.

Jinpeng Li, Haiping Wang, Jiabin Chen, Yuan Liu, Zhiyang Dou, Yuexin Ma, Sibei Yang, Yuan Li, Wenping Wang, Zhen Dong, et al. Cityanchor: City-scale 3d visual grounding with multi-modality llms. In *ICLR*, 2025a.

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*, 2025b.

Rong Li, Shijie Li, Lingdong Kong, Xulei Yang, and Junwei Liang. Seeground: See and ground for zero-shot open-vocabulary 3d visual grounding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 3707–3717, 2025c.

Zeju Li, Chao Zhang, Xiaoyan Wang, Ruilong Ren, Yifan Xu, Ruifei Ma, Xiangde Liu, and Rong Wei. 3dmit: 3d multi-modal instruction tuning for scene understanding. In *2024 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, pp. 1–5. IEEE, 2024.

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. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 7386–7394, 2026.

Zhenyang Liu, Yikai Wang, Sixiao Zheng, Tongying Pan, Longfei Liang, Yanwei Fu, and Xiangyang Xue. Reasongrounder: Lvlm-guided hierarchical feature splatting for open-vocabulary 3d visual grounding and reasoning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 3718–3727, 2025.

Boyu Mi, Hanqing Wang, Tai Wang, Yilun Chen, and Jiangmiao Pang. Language-to-space programming for training-free 3d visual grounding. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 3844–3864, 2025.

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. *arXiv preprint arXiv:2310.18773*, 2023.

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

Penglei Sun, Yaoxian Song, Xiangru Zhu, Xiang Liu, Qiang Wang, Yue Liu, Changqun Xia, Tiefeng Li, Yang Yang, and Xiaowen Chu. City-vlm: Towards multidomain perception scene understanding via multimodal incomplete learning. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 3448–3457, 2025.

Wenbin Tan, Jiawen Lin, Yuan Xie, Yachao Zhang, and Yanyun Qu. Uz3dvg: Unaided zero-shot 3d visual grounding with generated language conditions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9547–9557, 2026.

Yuan Tang, Xu Han, Xianzhi Li, Qiao Yu, Yixue Hao, Long Hu, and Min Chen. Minigpt-3d: Efficiently aligning 3d point clouds with large language models using 2d priors. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 6617–6626, 2024.

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.

Runsen Xu, Zhiwei Huang, Tai Wang, Yilun Chen, Jiangmiao Pang, and Dahua Lin. Vlm-grounder: A vlm agent for zero-shot 3d visual grounding. *arXiv preprint arXiv:2410.13860*, 2024.

Jianing Yang, Xuweiyi Chen, Shengyi Qian, Nikhil Madaan, Madhavan Iyengar, David F Fouhey, and Joyce Chai. Llm-grounder: Open-vocabulary 3d visual grounding with large language model as an agent. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7694–7701. IEEE, 2024.

Li Yang, Ziqi Zhang, Zhongang Qi, Yan Xu, Wei Liu, Ying Shan, Bing Li, Weiping Yang, Peng Li, Yan Wang, et al. Exploiting contextual objects and relations for 3d visual grounding. *Advances in Neural Information Processing Systems*, 36:49542–49554, 2023.

Zhengyuan Yang, Songyang Zhang, Liwei Wang, and Jiebo Luo. Sat: 2d semantics assisted training for 3d visual grounding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1856–1866, 2021.

Shunsuke Yasuki, Taiki Miyanishi, Nakamasa Inoue, Shuhei Kurita, Koya Sakamoto, Daichi Azuma, Masato Taki, and Yutaka Matsuo. Geoprog3d: Compositional visual reasoning for city-scale 3d language fields. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9737–9748. IEEE, 2025.

Zhenfei Yin, Jiong Wang, Jianjian Cao, Zhelun Shi, Dingning Liu, Mukai Li, Xiaoshui Huang, Zhiyong Wang, Lu Sheng, Lei Bai, et al. Lamm: Language-assisted multi-modal instruction-tuning dataset, framework, and benchmark. *Advances in Neural Information Processing Systems*, 36: 26650–26685, 2023.

Qihao Yuan, Kailai Li, and Jiaming Zhang. Solving zero-shot 3d visual grounding as constraint satisfaction problems. *arXiv preprint arXiv:2411.14594*, 2024a.

Zhihao Yuan, Jinke Ren, Chun-Mei Feng, Hengshuang Zhao, Shuguang Cui, and Zhen Li. Visual programming for zero-shot open-vocabulary 3d visual grounding. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20623–20633. IEEE, 2024b.

Yang Zhan, Zhitong Xiong, and Yuan Yuan. Rsvg: Exploring data and models for visual grounding on remote sensing data. *IEEE transactions on geoscience and remote sensing*, 61:1–13, 2023.

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*, 2026a.

Sha Zhang, Di Huang, Jiajun Deng, Shixiang Tang, Wanli Ouyang, Tong He, and Yanyong Zhang. Agent3d-zero: An agent for zero-shot 3d understanding. In *European Conference on Computer Vision*, pp. 186–202. Springer, 2024.

Shuai Zhang, Hongye Hou, Qinghe Liu, Zhuoxiao Li, Dongli Wu, Jing Ou, Yuan Liu, and Wufan Zhao. Citystar: Structured and topology-aware reasoning for open-vocabulary urban 3d grounding. *arXiv preprint arXiv:2609.19911*, 2026b.

A SUPPLEMENTARY MATERIAL

This supplement specifies evaluation protocols, implementation details, evidence construction, and recovery operations, followed by qualitative comparisons and limitations.

A.1 DATASETS AND EVALUATION PROTOCOL

A.1.1 BENCHMARK DESCRIPTIONS

Table 4 summarizes the benchmarks. Published dataset totals are distinct from evaluation-split sizes.

Table 4: Overview of the urban grounding benchmarks. Annotation counts describe the published datasets. NO and ND denote Novel Objects and Novel Descriptions.

Benchmark	Scene source	Coverage	Published scale	Evaluation
CityRefer	SensatUrban	4 categories	41 scenes; approximately 35,000 descriptions	NO / ND
CityAnchor	Synthetic STPLS3D	9 categories	25 scenes; 1,448 text–object pairs	NO / ND
CitySTAR-3D	SensatUrban and STPLS3D	15 semantic groups	1,286 pairs: 686 from SensatUrban and 600 from STPLS3D	Open-vocabulary grounding

CityRefer (Miyanishi et al., 2023) provides geography-aware descriptions of buildings, cars, ground, and parking targets, with approximately 5,000 landmark annotations associated with OpenStreetMap. The scene count follows CityAnchor (Li et al., 2025a). CityAnchor covers nine categories: building, vegetation, aircraft, truck, vehicle, light pole, fence, street sign, and bike; its original protocol assigns 80% of text–object pairs to training and 20% to testing. CitySTAR-3D (Zhang et al., 2026b) evaluates open-vocabulary grounding of specialized objects and extended regions. For its STPLS3D-derived samples, it adopts CityAnchor instance boxes and supplies richer contextual descriptions. CityGRASP outputs axis-aligned 3D boxes, preserving the annotated extent for parking regions and building clusters.

A.1.2 EVALUATION SETTINGS

We follow the benchmark-provided partitions and evaluation implementations. Novel Objects (NO) evaluates instances absent from the corresponding training split; Novel Descriptions (ND) evaluates previously represented objects with new descriptions. NO does not necessarily imply unseen semantic categories. CitySTAR-3D provides the separate open-vocabulary evaluation. For N evaluation queries,

$$\mathrm{Acc}\,@\tau = \frac{100}{N}\sum_{n=1}^{N}\mathbf{1}\Big[\mathrm{IoU}(\hat{B}_n, B_n^{\mathrm{gt}}) \geq \tau\Big], \qquad \tau \in \{0.25, 0.50\}, \tag{6}$$

where $\hat{B}_n$ and B_n^{gt} are predicted and annotated boxes. The denominator includes the complete evaluation set.

Certification in Section 3.5 evaluates query constraints, represented competitors, and search coverage. Benchmark correctness is evaluated separately using annotated boxes.

A.1.3 BASELINE IMPLEMENTATIONS AND COMPARISON PROTOCOL

InstanceRefer, 3DVG-Transformer, EDA, CityRefer, CityAnchor, and CitySTAR follow their official evaluation implementations. The CitySTAR rerun uses Qwen3.5-9B for reasoning and Qwen3-reranker-4B for verification. CityGRASP uses Qwen3.5-9B for planning and code synthesis and Qwen3-VL-4B-Instruct for visual observations. Both systems use SAM3-based candidate construction, but their subsequent observations and reasoning differ. Table 1 compares complete systems; Table 2 controls initial entities and backbone models within CityGRASP.

A.2 Implementation Details

A.2.1 Models and Inference Configuration

Table 5 reports the inference settings. Foundation-model parameters remain frozen. Stage 0 lifts SAM3 masks through point-ID buffers and merges them into entity memory; subsequent observations update geometry and semantic evidence. The scalar visual calibration is fitted separately from the frozen foundation models.

Table 5: Implementation and inference configuration of CityGRASP.

Setting	**Configuration**
Models and execution	
Hardware	Four NVIDIA RTX 4090 GPUs.
Planner and geo-coder	Frozen Qwen3.5-9B.
Visual observation	Frozen Qwen3-VL-4B-Instruct.
Decoding temperature	0.
Scene representation	
Open-vocabulary perception	SAM3 on bird's-eye and oblique views.
Stage 0 view configuration	$2 + 24$ views, totaling 26.
Image-to-point association	Point-ID buffers.
Output representation	Axis-aligned 3D bounding box.
Evidence and verification	
Geometric resampling	Nominal geometry and 16 point subsamples drawn without replacement.
Visual evidence	Sigmoid-calibrated Yes–No log-probability differences; mean support and extrema across prescribed views.
Constraint weights	$w_j = 1/\lvert\mathcal{J}_q\rvert$ for necessary non-selection constraints.
Support and rejection thresholds	$\tau_{\mathrm{sup}} = 0.8$ and $\tau_{\mathrm{rej}} = 0.2$.
Non-selection certification margin	$\theta_{\mathrm{cert}} = 0.05$.

A.2.2 Ablation Definitions

Table 6 summarizes selected component controls from Table 2. All variants share initial entities and backbone models.

The complete-program control retains program checking and revision, isolating the use of intermediate results during progressive construction. The scene-conditioning control retains runtime geometry access but excludes scene-specific evidence from code generation and revision. The recovery ablation removes both verification-triggered revision and restoration; Full Restart instead retains revision and tests the benefit of reusing the valid execution prefix.

A.2.3 Efficiency and Cost Accounting

CitySTAR, Full Restart, and Selective Recovery are evaluated on the same 1,081 pooled CityRefer NO/ND queries. Pooled accuracy is

$$\mathrm{Acc}_{\mathrm{pooled}}\,@\tau = 100\frac{C_{\mathrm{NO}}(\tau) + C_{\mathrm{ND}}(\tau)}{N_{\mathrm{NO}} + N_{\mathrm{ND}}}, \quad N_{\mathrm{NO}} + N_{\mathrm{ND}} = 1081, \tag{7}$$

where C counts correct predictions at threshold τ. Full Restart retains revised inputs and recomputes the process; Selective Recovery retains the valid prefix and reconstructs the affected suffix. Both use identical hardware and execution settings. Mean costs cover all evaluated queries and include online execution and failed attempts, excluding reusable scene preprocessing; tokens include input and output tokens. Selective Recovery yields 855 correct predictions at IoU 0.50, versus 832 for Full Restart (79.09% versus 76.97%), while reducing runtime, model calls, and tokens by 44.6%, 46.5%, and 47.9%. The net increase of 23 correct predictions is not a count of recovery events. Table 3 also reports the remaining cost difference relative to CitySTAR.

Table 6: Selected CityGRASP component-ablation controls. Each variant changes the stated mechanism.

Variant	Intervention
Stage 0: Query-guided scene parsing	
w/o Entity Memory Updates	Freeze entity memory after initialization; retain computations over existing entities and execution evidence.
Stages 1–2: Progressive program construction	
w/o Progressive Synthesis	Generate a complete program before each execution attempt; retain program checks, final verification, and recovery.
Stage 2: Scene-conditioned geo-code synthesis	
w/o Geo-Code Synthesis	Replace generated geo-code with predefined relation implementations.
w/o Scene Conditioning	Remove scene-specific evidence from code generation and revision; retain runtime geometry access.
Stage 3: Reversible grounding certification	
w/o Uncertainty-Aware Verification	Use point-evidence decisions; retain query constraints, selection rules, coverage checks, and recovery.
w/o Reversible Recovery	Disable verification-triggered revision and restoration; retain program checks and verification.
Full CityGRASP	Retain observation updates, adaptive co-planning, progressive scene-conditioned synthesis, and reversible certification.

A.3 Additional Method Details

A.3.1 Query Parsing and Entity Memory

Stage 0 constructs $\mathcal{Q}_q = (c_q, \mathcal{T}_q, \mathcal{H}_q, \mathcal{R}_q)$, preserving the target concept, entity-bound attributes, contextual references, and spatial requirements. For a car near a red building, *red* constrains the building. A hypothesis $h = (i, \mathbf{b})$ combines a target identity with a reference assignment; evidence from different assignments cannot be combined to satisfy a single reference. Entity memory stores geometry, boxes, semantic observations, and geographic context. Versioned inputs and bindings identify which execution evidence becomes invalid after revision.

A.3.2 Planning State and Module Contracts

The state $\mathcal{Z}_t = (\mathcal{M}_t, \Omega_t, \mathcal{C}_t, \mathcal{D}_t, \mathcal{F}_t, \mathcal{U}_t, \mathcal{E}_t)$ separates entity memory, scope, active hypotheses, deferred hypotheses, unexplored branches, unresolved requirements, and valid evidence. Equation 2 selects the next scope and subproblem. Table 7 summarizes representative instruction excerpts and structured outputs.

Table 7: Module instructions and output contracts. Quoted text gives representative prompt excerpts.

Module	Instruction and output
Query parsing	"Keep names verbatim." Returns `target_concept`, `attributes`, `contextual_entities`, `spatial_constraints`, and `selection`.
Planning	"Describe WHAT must be computed, NEVER how." Returns `subproblem`, `scope`, `required_context`, `program_spec`, `needs_perception`, and `perception_request`.
Code generation	"Never hard-code entity IDs or consult ground truth." Returns `program_id`, `target_constraint`, `code`, `required_primitives`, `expected_output_type`, and `depends_on`.
Visual judgment	"Answer exactly Yes or No. Do not infer an unseen attribute from context." Returns the answer and the first-token log probabilities of `Yes` and `No`.
Revision	"Dependency edges specify invalidation, not proof of root cause. Interval overlap is not a programming error." Returns a repair `index` and an evidence-based `reason`.

The planning contract distinguishes target candidates from contextual inputs. Required road geometry remains accessible beyond the current target scope. The geo-coder receives the plan, bindings, candidate domain, primitive interfaces, and feedback. Non-selection constraints return support; selection operators return physical measurements with units.

Structured parsing example. Listing 1 shows the supplied parser response for *"The building along New Square Road, next to the New Square Dental & Chiropody building."* The controller separately adds the necessary `concept:building` check.

Listing 1: Structured query-parsing response.

```
{
  "target_concept": "building",
  "attributes": [],
  "contextual_entities": [
    {"name": "New Square Road", "kind": "road"},
    {"name": "New Square Dental & Chiropody building",
     "kind": "entity"}
  ],
  "spatial_constraints": [
    {"relation": "along",
     "arguments": ["New Square Road"],
     "raw_text": "along New Square Road"},
    {"relation": "adjacent_to",
     "arguments": ["New Square Dental & Chiropody building"],
     "raw_text":
       "next to the New Square Dental & Chiropody building"}
  ],
  "selection": ""
}
```

Scope and coverage. Scope restriction defers hypotheses to $\mathcal{D}_t$ without disproving them. Unprocessed scopes and reference assignments remain in $\mathcal{F}_t$ until explored or closed by recorded evidence of no admissible target. Missing observations cannot justify closure. Exclusions and branch closures retain dependencies so that revision can reverse them.

A.3.3 Geo-Code Synthesis and Evidence Construction

Relation semantics and support scales remain fixed. Scene conditioning determines geometric inputs, reference frames, and primitive composition within the query contract. For a road relation, these choices concern the relevant road section and its local orientation, rather than redefining the relation to favor a candidate. Each computation records its geometry type, distance definition, units, and input dependencies; these remain fixed during resampling. Syntax, permitted-call, geometry, and output-type checks guard execution.

Primitive interfaces. Table 8 lists selected interfaces. Geometry getters return result objects with a `value` field; projection returns an array. Road and entity identifiers belong to separate registries. Missing or degenerate geometry leaves a requirement unresolved.

Geometric sampling. The implementation caches at most 2,048 observed points per entity and evaluates nominal geometry and $K = 16$ independently seeded subsamples. Sampling without replacement retains

$$n_{\text{keep}} = \min\left(N, \max(8, \lceil 0.8N \rceil)\right), \tag{8}$$

where N is the cached point count; fewer than eight valid points leave the entity unresolved. The PCG64 seed is the unsigned big-endian integer represented by the first eight bytes of the SHA256 digest of `20260924:{program_id}:{entity_id}:{k}`. Each sample recomputes the relevant center, box, planar convex hull, and area. Programs and bindings remain fixed, as does map-derived road geometry without corresponding point observations.

Support and empirical intervals. For signed satisfaction margin m_j and relation scale s_j,

$$\rho_j = \text{clip}\left(\frac{1}{2} + \frac{m_j}{2s_j}, 0, 1\right), \qquad s_j > 0. \tag{9}$$

Table 8: Selected primitive interfaces used by generated spatial programs.

Primitive	Input and returned value
`get_footprint`	Entity identifier → planar footprint.
`get_named_road`	Road name → collection of road geometries.
`project_to_xy`	Point or point array → planar array.
`xy_path_distance`	Points or polylines → continuous planar distance.
`boundary_distance`	Two geometries → boundary distance.
`segment_polygon_margin`	Segment endpoints and polygon → signed crossing measurement.
`support_from_margin`	Entity, margin, and relation name → normalized support.
`vlm_verify`	Candidate and constraint → visual evidence with availability and score.

With $k = 0$ denoting nominal geometry,

$$[\ell_j, u_j] = \left[\min_{0 \le k \le K} \rho_j^{(k)}, \max_{0 \le k \le K} \rho_j^{(k)} \right]. \tag{10}$$

Invalid inputs, degenerate geometry, exceptions, or any failed required sample leave the interval at $[0, 1]$. The executor does not compute an envelope from successful samples alone. Table 9 gives the margin conventions.

Table 9: Geometric margin conventions. Distances and scales are in meters unless stated otherwise. Missing required geometry leaves the relation unresolved.

Relation	Signed margin m_j	s_j
Within distance	$r - d$, using the numerical radius stated in the query.	1
Near	$20 - d$.	5
Adjacent to	$5 - d$, using boundary distance between distinct objects.	2
Along road	$10 - d$, with d the planar distance to the bound road geometry, whose representation is fixed by the contract.	2
At road end	$10 - d$ to a genuine endpoint after connected road segments are joined.	2
At junction	$\min(10 - d_1, 10 - d_2)$ for the specified intersecting roads.	2
At a road corner	$10 - d$ to an observed bend of at least 30° when only one road is specified.	2
Directional	Signed displacement in the specified geographic or observation frame.	1
Above / below	Signed vertical separation.	1
Inside / contains	Minimum signed boundary clearance with boundary-crossing checks.	1
Between	Minimum longitudinal and lateral clearance within the anchor-defined corridor.	1
Across road	Minimum normalized opposite-side and road-crossing margin; dimensionless.	1

Road-end evaluation excludes shared internal endpoints of joined segments. Directional relations require an available geographic or observation frame. Adjacency requires distinct objects; containment with zero distance does not establish adjacency.

For *between*, let anchor centroids be $\mathbf{a}, \mathbf{b}$, $L = \|\mathbf{b} - \mathbf{a}\|$, $\mathbf{v} = (\mathbf{b} - \mathbf{a})/L$, and $\mathbf{n} = (-v_y, v_x)$. For target position $\mathbf{x}$, let $z = (\mathbf{x} - \mathbf{a})^\top \mathbf{v}$ and $y = (\mathbf{x} - \mathbf{a})^\top \mathbf{n}$. Facing anchor-footprint extents $A_{\text{front}}, B_{\text{front}}$ are expressed in this same longitudinal coordinate system. Then

$$m_{\text{between}} = \min\left(z - A_{\text{front}}, B_{\text{front}} - z, w(z) - |y|\right), \tag{11}$$

where $w(z) = (1 - z/L)w_A + (z/L)w_B$ interpolates the perpendicular anchor half-widths. Missing footprints, coincident centroids, or overlapping longitudinal extents leave the relation unresolved. Closest selection retains physical distance intervals and requires the selected distance upper bound to be below every eligible competitor's lower bound, as in Equation 5.

Visual calibration. Following the main paper, the sigmoid mapping is fitted on development scenes disjoint from testing. The development set contains 168 candidate–constraint pairs from 71 objects:

104 category pairs and 64 vehicle-color pairs, balanced between 84 positives and 84 negatives. Category labels come from released object annotations; color labels use explicit descriptions of red, blue, white, and black cars. Negative categories pair buildings with car and other categories with building; negative colors use red–blue and white–black swaps, treating the paired descriptions as incompatible. For the first-token log-probability difference $z = \log P(\texttt{Yes}) - \log P(\texttt{No})$,

$$\begin{aligned} &p = \sigma(az + b), \\ &a = 0.2109169343221665, \qquad b = 0.36632797825141816. \end{aligned} \tag{12}$$

L-BFGS-B minimizes binary negative log likelihood with $10^{-6}(a^2 + b^2)/2$ regularization and $a \geq 0$. Each pair has equal total weight and one development view. At inference, mean calibrated support defines ρ and extrema over prescribed valid views define $[\ell, u]$. A missing required view or answer-token probability retains $[0, 1]$; nominal support uses available views, or 0.5 if none is valid. Candidate-judgment views are distinct from the 26 Stage 0 scene views. Both geometric and visual intervals follow Equation 3; they measure sampling and view sensitivity, not statistical confidence.

A.3.4 Verification and Selective Recovery

A necessary constraint is supported if $\ell_j \geq \tau_{\text{sup}}$, contradicted if $u_j \leq \tau_{\text{rej}}$, and unresolved otherwise. Every necessary concept, attribute, and non-selection relation must be supported under a consistent binding. Aggregate support cannot compensate for an unsupported necessary constraint.

Competition and revision. Competitors include active and deferred hypotheses with different target identities, excluding contradicted hypotheses and those reliably outside a selection domain. Unresolved alternatives remain competitors; rejecting one binding does not reject other bindings for the same target. Acceptance applies Equation 4 or Equation 5, together with $\mathcal{F}_t = \varnothing$. Empty maxima and minima are $-\infty$ and $+\infty$, respectively. Missing observations return to Stage 0, unresolved bindings or coverage to Stage 1, and computational errors to Stage 2. Interval overlap calls for discriminating evidence without changing the relation semantics.

Recovery boundary. Revision locates the earliest invalid execution block, preserves the corrected inputs and valid prefix, and reconstructs the suffix. Exclusions and branch closures supported by the discarded suffix are revoked. The valid prefix supplies retained evidence for subsequent verification. Listing 2 gives algorithmic state operations, rather than a verbatim software API.

Listing 2: Pseudocode for dependency-based recovery.

```
def recover(state, trace, revised_inputs):
    state.apply_revised_inputs(revised_inputs)
    r = earliest_invalid_block(trace, state.versions)
    if r is None:
        return state

    suffix = trace[r:]
    state.revoke_exclusions_supported_by(suffix)
    state.reopen_branches_closed_by(suffix)
    state.remove_evidence_from(suffix)
    state.retain_valid_prefix(trace[:r])

    # Revised inputs remain in the current state.
    # Restored hypotheses undergo normal acceptance checks.
    return reconstruct_suffix(state, valid_prefix=trace[:r])
```

Restoration revokes an invalidated exclusion and returns a hypothesis to comparison; it does not itself certify that hypothesis. Activating a deferred hypothesis changes its processing status, while discovering an entity absent from memory requires additional perception.

A.4 Program and Recovery Examples

A.4.1 Geometric Computation for Across-Road Reasoning

Across-road reasoning requires an opposite-side configuration and intersection with the relevant finite road footprint. The generated computation must bind the road section and obtain its local frame

before evaluating these conditions. At a bend or junction, a tangent from an unrelated section does not resolve the requested relation. An ambiguous section or unavailable footprint leaves the measurement unresolved. The following example expands the geometric evaluation after these inputs have been bound; it is not a full code-generation transcript.

For a reference-adjacent road section with unit tangent $\mathbf{v}$, normal $\mathbf{n} = (-v_y, v_x)$, and point $\mathbf{r}$, define the target and reference offsets

$$d_x = (\mathbf{x} - \mathbf{r})^\top \mathbf{n}, \qquad d_a = (\mathbf{a} - \mathbf{r})^\top \mathbf{n}. \tag{13}$$

The opposite-side margin is

$$m_{\text{side}} = -\frac{d_x d_a}{|d_x| + |d_a|}. \tag{14}$$

A zero denominator leaves the measurement unresolved. The crossing margin m_{cross} is the connecting segment's length inside the road footprint when crossing occurs, negative separation when disjoint, and zero at contact. Define

$$m_{\text{across}} = \min\left(\frac{m_{\text{side}}}{1\,\text{m}}, \frac{m_{\text{cross}}}{1\,\text{m}}\right). \tag{15}$$

Equation 9 uses $s_{\text{across}} = 1$. Thus, nominal support reaches 0.8 only if both normalized margins reach 0.6; qualification additionally checks the lower bound over prescribed samples. Listing 3 implements this mapping using the supplied geometry primitives.

Listing 3: Across-road margin and support computation. Positions and road geometry are expressed in meters.

```
import numpy as np
def across_road_evidence(x, a, r, v, road_footprint):
    inputs = (x, a, r, v, road_footprint)
    if any(value is None for value in inputs):
        return {"status": "unresolved"}
    x, a, r, v = [
        np.asarray(value, dtype=float)[:2]
        for value in (x, a, r, v)
    ]
    if not all(np.all(np.isfinite(p)) for p in (x, a, r, v)):
        return {"status": "unresolved"}
    norm = np.linalg.norm(v)
    if norm == 0:
        return {"status": "unresolved"}
    v = v / norm
    n = np.array([-v[1], v[0]])
    dx = float(np.dot(x - r, n))
    da = float(np.dot(a - r, n))
    denominator = abs(dx) + abs(da)
    if denominator == 0:
        return {"status": "unresolved"}
    side_margin = -dx * da / denominator
    cross_margin = segment_polygon_margin(
        x, a, road_footprint
    ).value
    if cross_margin is None or not np.isfinite(cross_margin):
        return {"status": "unresolved"}
    margin = min(side_margin / 1.0, cross_margin / 1.0)
    support = float(np.clip(0.5 + margin / 2.0, 0.0, 1.0))
    return {
        "status": "measured",
        "side_margin_m": side_margin,
        "cross_margin_m": float(cross_margin),
        "support": support
    }
```

Opposite sides of an extended road axis do not establish intersection with the finite road footprint. The crossing condition excludes this configuration. The program, road frame, and binding remain fixed during resampling; revising the binding invalidates its dependent computation.

A.4.2 Selective Recovery of an Excluded Target

Figure 6 illustrates the successful case at CityRefer ND row 163 in `cambridge_block_20`: *"The building is located at the end of City Road along Burleigh Street and is opposite the Primark building."* The three spatial constraints refer to the same target, while the opposite-building constraint additionally depends on the Primark binding.

Exclusion and reference revision. The initial opposite-building computation uses an incorrect Primark binding and excludes candidate B. Verification prompts reference reconsideration. Once the binding changes, evidence computed with its previous version can no longer justify the exclusion. The revised binding changes the contextual input; it does not change the meaning of the query or lower the acceptance thresholds.

Restoration and recomputation. The controller identifies the earliest invalid block, retains the valid prefix, and revokes exclusions and branch closures supported by the discarded suffix. Candidate B returns to the represented comparison set. The controller reconstructs the suffix under the revised Primark binding and evaluates the opposite-building relation again. Restoring B revokes the previous exclusion; it does not supply positive evidence for any query constraint.

Final decision. The revised opposite-building computation supports B. The final decision also checks its target concept, road-end and along-street requirements, represented search coverage, and separation from eligible competing targets. The successful outcome identifies B as the referred building. Restoration is therefore an intermediate state change, followed by renewed verification rather than automatic acceptance. This account describes the recovery mechanism; the component ablations and Full Restart comparison provide its quantitative assessment.

A.5 Additional Qualitative Comparisons

Figures 7–9 show five successful examples per benchmark alongside CityRefer, CityAnchor, and CitySTAR. Red boxes indicate predictions.

CityRefer. Similar vehicles are distinguished through neighboring objects and parking context, while building queries use named landmarks and roads. These cases illustrate instance disambiguation beyond category and appearance matching.

CityAnchor. The examples cover buildings, vegetation, a vehicle, a light pole, and a bicycle. Dimensional attributes and neighboring entities distinguish targets across object scales.

CitySTAR-3D. The examples include an aircraft, a building cluster, vegetation, a parking region, and a truck. Cluster and parking descriptions require the intended region extent; individual objects require attributes and spatial constraints involving contextual entities.

A.6 Failure Analysis and Limitations

Figure 10 shows two failures per benchmark, with predictions and annotated targets marked in red and green in the rightmost panels. The failures highlight three challenging conditions: nearby instances with similar attributes in CityRefer, identical object models with insufficiently precise contextual descriptions in CityAnchor, and complex descriptions combined with sparse point clouds or occlusion in CitySTAR-3D. These examples do not quantify failure frequencies.

CityRefer: similar vehicles in a shared parking context. The first query identifies a red car by its parking location and the black and white cars to its left, right, and rear. The prediction and annotation select different red cars within the parking area. The two candidates have similar visible attributes and are close to one another, so both inherit much of the same parking context. This combination makes the neighboring-car configuration essential for disambiguation. The left, right, and behind relations must distinguish the local configurations around those candidates, using a consistent interpretation of the query's viewing frame.

CityRefer: selecting the intended grass region. The second query refers to grassy ground on the right side of Northampton Street and north of Westminster College. The predicted box and annotated box cover different grass patches separated by paved paths. Both patches have similar

surface appearance and lie in the same local area. Recognizing the grass category and general geographic neighborhood therefore leaves the intended patch unresolved. The street and college references must distinguish these nearby regions. This error concerns region identity as well as localization: tightening the predicted box around the selected patch would still leave a different patch from the annotation.

CityAnchor: identical models and contextual ambiguity. The car query specifies a red car in the middle of the road beside a blue car. The local crops show red vehicles, but the overview places the prediction and annotated target at different locations. CityAnchor's virtual scenes can reuse exactly the same object model at multiple locations. Such instances share intrinsic appearance cues, making the surrounding context central to identification. Errors arise when descriptions of that context are insufficiently precise to separate the repeated instances. For this query, the road position and neighboring blue car carry the distinguishing information. Matching the red-car model alone cannot identify the intended instance.

CityAnchor: a truck constrained by a fence and building. The truck query combines a black vehicle, containment within a fence, and an opposite irregular building with a white roof and an approximate height of 12 meters. The predicted and annotated trucks occupy different parts of the scene. With repeated models, vehicle appearance offers limited discrimination, so the fence and opposite building must identify the intended occurrence. An imprecise account of the surrounding environment can leave several locations plausible despite the number of attributes in the query. The fence, roof appearance, building height, and opposite relation must be interpreted together for the same truck. Contextual detail helps only when it distinguishes the target's surroundings from those of competing instances.

CitySTAR-3D: complex descriptions and sparse geometry. The building query specifies a gray roof, approximate dimensions of $9 \times 6 \times 9$ meters, and named buildings to the northeast and southwest. The prediction selects a different building from the annotation despite a broadly similar roof appearance. The description combines appearance, metric extent, and several named spatial references. In sparsely sampled parts of the point cloud, incomplete geometric observations make these constraints harder to evaluate reliably. Missing observations may affect the estimated building extent or the geometry used to relate it to the Sinclair Building, Swinhoe House, and Petersfield Mansions. Roof appearance alone cannot compensate for uncertain evidence about the remaining conditions in this complex description.

CitySTAR-3D: occlusion and neighborhood constraints. The vegetation query combines a tall green target of specified dimensions with a building to the northwest, four tall vegetation entities, and medium vegetation to the east, including metric proximity constraints. The prediction and annotation select different vegetation locations in the overview. Complex occlusion further complicates grounding by limiting observations of the target and its contextual entities. With sparse point clouds, partially observed vegetation and neighboring structures can make extent, direction, and box-to-box distance evidence uncertain. The long description then requires several constraints to be resolved from incomplete observations. Shared green appearance provides little help in deciding which vegetation instance satisfies that combination.

Implications for verification and recovery. Verification must retain candidate-specific bindings when combining appearance, contextual, and geometric evidence. Selective Recovery can reconsider a represented candidate after an invalid decision is identified, but it cannot by itself supply missing observations or make an imprecise description more discriminative. Resampling fixed geometry probes sensitivity to the available points; it does not reconstruct unobserved surfaces or establish semantic correctness. Benchmark results and ablations provide the quantitative assessment.

Methodological Limitations. CityGRASP depends on entity coverage and the geometric inputs provided by perception. Completing represented search branches does not establish complete real-world entity coverage. The evidence intervals describe sensitivity to sampled geometry and prescribed views rather than semantic correctness. Visual calibration covers the specified categories and vehicle colors. Selective Recovery reduces inference cost relative to Full Restart but remains more expensive than CitySTAR in Table 3. The experiments report system-level comparisons on three benchmarks and component ablations on CityRefer.



Figure 7: Qualitative comparisons on CityRefer. CityGRASP resolves appearance and geographic context to identify the referred targets.



Figure 8: Qualitative comparisons on CityAnchor. CityGRASP grounds objects at different scales using appearance, dimensions, and relations.



Figure 9: Qualitative comparisons on CitySTAR-3D. CityGRASP resolves specialized objects and extended regions under spatial constraints.



Figure 10: Failure cases on CityRefer, CityAnchor, and CitySTAR-3D, from top to bottom. In the rightmost panels, red and green boxes indicate predictions and annotated targets, respectively.