MapScout: An Agentic Harness for Map Editing and Geospatial Data Labeling

Vishal Kumar
vishku@amazon.com
Amazon
Bellevue, WA, USA

Shaishav M. Maisuria
smaisur@amazon.com
Amazon
Bellevue, WA, USA

Emre Eftelioglu
efteli@amazon.com
Amazon
Bellevue, WA, USA

Abstract

Accurate map paths are crucial for routing vehicles, pedestrians, and autonomous systems, yet keeping that data fresh demands continual editing of features and geometry, and labeling of attributes such as surface type and lane count. At scale this is labor-intensive, and each change calls for spatial reasoning and verification that has resisted full automation, keeping a human in the loop. We present MapScout, an agentic-harness application that cuts the reviewer's workload while keeping the human as final authority. A supervisor agent decomposes a map tile into per-feature subtasks and dispatches subagents that reason over satellite imagery, OpenStreetMap, and GPS traces and invoke ML models (segmentation, geometry-stitching) as tools to propose edits and labels. Agents invoke deterministic, non-LLM validators, such as topology and route evaluators, as tools to self-correct their proposals and to flag low-confidence edits for the human. Our contribution is the harness, built on open components (Strands, Leaflet with Geoman) and public data. Demo session attendees interact with MapScout live, watching the subagents extract walking paths and sidewalk infrastructure that enrich the map for pedestrian and accessibility-constrained routing.

CCS Concepts

• **Information systems** → **Geographic information systems**; • **Computing methodologies** → **Multi-agent systems**; • **Human-centered computing** → *Interactive systems and tools.*

Keywords

agentic GeoAI, map editing, data labeling, human-in-the-loop, training data generation, OpenStreetMap, vision-language models, large language model agents

ACM Reference Format:
Vishal Kumar, Shaishav M. Maisuria, and Emre Eftelioglu. 2026. MapScout: An Agentic Harness for Map Editing and Geospatial Data Labeling. In *The 34th ACM International Conference on Advances in Geographic Information Systems (SIGSPATIAL '26), November 03–06, 2026, Riverside, CA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3841645.3843126

1 Introduction

Keeping a map fresh is a continuous operation: *editing* (adding and correcting features and geometry) and *labeling* (attaching and updating attributes). In community and commercial mapping alike, this work is done largely by hand and does not scale; the backlog of missing and stale features is large, especially for pedestrian and accessibility infrastructure, which is already sparse in open data [10, 15].

Automated extractors and vision-language models (VLMs) propose features from imagery at scale [10, 12], but their output still needs a human to curate and translate it into map elements and topology before commit. A model that confidently asserts a paved sidewalk where the ground is only a roadside strip of grass injects an error that downstream consumers will trust. Manual review and curation are the bottleneck, and recent interactive geospatial-AI systems pair a single model with a map interface [17] but still surface raw output to the editor, so the review burden is undiminished.

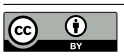
To address this, we present MapScout. Unlike prior interactive systems, where the editor still curates unscreened model output, MapScout has its agents invoke deterministic checks as tools to self-correct, so the human reviews a corrected set before the final commit. The reviewer's job shifts from authoring features to confirming a smaller, pre-checked set in the tool's interface, which also shows each agent's worklog artifacts (its reasoning log and the intermediate images it reasons over). The paper contributes:

- **A supervisor–subagent editing harness** that decomposes a map tile into per-feature subtasks and dispatches subagents in parallel; each reasons over imagery and invokes specialized ML models as tools to propose an edit or label.
- **Deterministic validators the agents invoke as tools** to self-correct edits, including a topology and routing impact analysis that catches connectivity-breaking changes before they reach a router. Low-confidence edits are flagged for the human.
- **An open, reproducible demonstration** on sidewalk extraction and surface labeling over public data, with a measured study of how much the validation layer self-corrects agent proposals and reduces human curation and handling time.

The remainder of the paper covers related work (Section 2), the frameworks and tools MapScout builds on (Section 3), the harness design (Section 4), the live demonstration (Section 5), and its operating characteristics (Section 6).

2 Related Work

Extracting pedestrian infrastructure from imagery. Tile2Net extracts sidewalk and crosswalk networks from aerial tiles with a segmentation model [10], CitySurfaces segments sidewalk surface



SIGSPATIAL '26, Riverside, CA, USA

ACM ISBN 979-8-4007-2950-8/2026/11
https://doi.org/10.1145/3841645.3843126

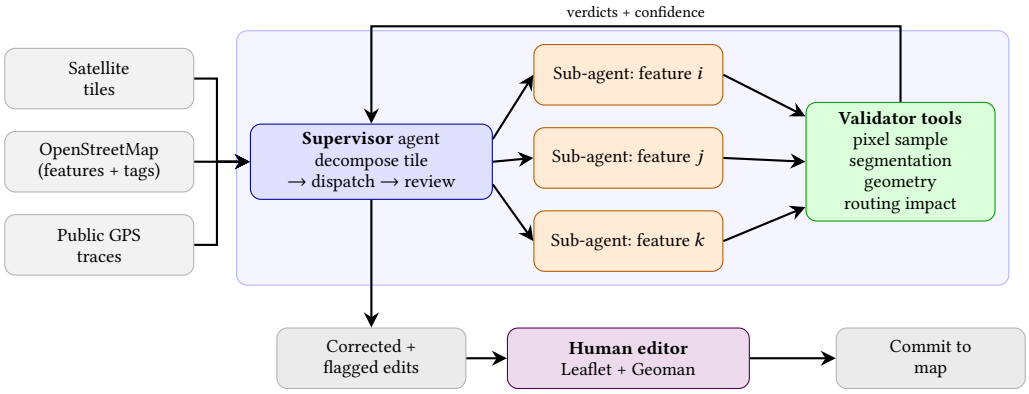


Figure 1: The MapScout harness. A supervisor decomposes a tile and dispatches subagents that reason over imagery, OpenStreetMap, and GPS traces to propose edits or labels. Agents self-correct using deterministic validators (topology, routing impact). A human editor reviews each edit with its artifacts and reasoning before commit.

materials at city scale [9], and Project Sidewalk collects accessibility attributes through guided human crowdsourcing [15]; PathwayBench then benchmarks how *routable* such inferred networks are [18]. These methods produce features in bulk or score a finished network, but the decision to commit any one of them to a live map remains a separate, manual step that MapScout targets.

LLM agents over geospatial data. A recent approach treats an LLM as the reasoning core of a GIS, where GIS Copilot realizes an agent that performs spatial analysis from natural language [2], and Text-to-OverpassQL maps language to executable OpenStreetMap queries [16]. These agents *query or analyze* geodata; MapScout instead has agents reason, validate, and then *propose edits*.

Grounding and verifying generative models. Vision-language models such as GeoChat [12] reason over imagery but, like language models generally, can produce plausible yet incorrect content [11]; a common remedy pairs the generator with a separate verifier that screens its outputs [13]. MapScout applies this to maps with *deterministic, non-LLM* checks and a human gate, not a second model.

3 Frameworks and Tools

MapScout is built largely on open-source tools and public data, described below. An *LLM agent* calls external functions (*tools*), observes results, and acts; we build the loop on *Strands*, an SDK in which one agent can invoke another as a tool [4]. A *vision-language model* (VLM) reads a satellite crop and reasons over its content. We build on *Segment Anything 2* (SAM 2) [14] for segmentation and an ML *geometry-stitching* model to assemble vector geometry; for deterministic, non-LLM checks and geometry authoring we use *Shapely* [7]. Our ML models run on *Amazon SageMaker* [3]. The substrate is *OpenStreetMap* [8]; the editor works in *Leaflet* [1] with *Geoman* [6] for hand-editing vector geometry. A *tile* is a fixed cell, split by the supervisor into per-feature subtasks.

4 The MapScout Harness

A map tile passes through four stages (Figure 1): a supervisor splits the work, subagents propose edits and invoke validators to self-correct them, and a human reviews and commits the approved edits. The harness coordinates these stages but is agnostic to the model and task. The demonstration covers sidewalk geometry, road-edge walkways, paved/unpaved surface, and obstruction presence; curb ramps, crossing nodes, steps, and grade are roadmap.

Decomposition. The supervisor reads the tile's OpenStreetMap context, splits it into per-feature subtasks (one per road or candidate feature), and dispatches a subagent for each in parallel.

Proposal. A subagent reasons over the satellite crop and calls ML tools (SAM 2 for segmentation, geometry-stitching) to propose an edit (a sidewalk centerline or road-edge walkway to add or correct) with labels such as surface type. Walking-classified GPS traces [5], where available, steer the path toward the building entrance it serves. The supervisor reviews each proposal and can reject it for rework, capped at two rounds; unconfirmed proposals are flagged low-confidence, not dropped. Each round records the revised proposal with its reasoning trace and tool-call artifacts, so the full run is inspectable.

Validation as agent tools. Rather than trust a confident proposal, the subagent invokes independent, non-LLM validators as tools and revises its proposal in response to their findings. *Per-proposal* checks come first: pixel sampling classifies a small RGB patch (paved versus vegetation) to test a surface claim, SAM 2 masks confirm surfaces and obstructions, and Shapely verifies crossings and offsets against the existing map. When a check fails the subagent self-corrects on its next revision. A claim no check can confirm is given a low confidence band and flagged for the human. A *topology and routing impact analysis* applies the edit to a working-copy network and partitions it into strongly-connected components (Tarjan's algorithm). It then finds any new connectivity issues introduced by the edit. An *island* connects only to itself, so no edge crosses its boundary. A *source* or *sink* can only be left or only entered. A multi-segment *jail* can be reached but never left. Such defects

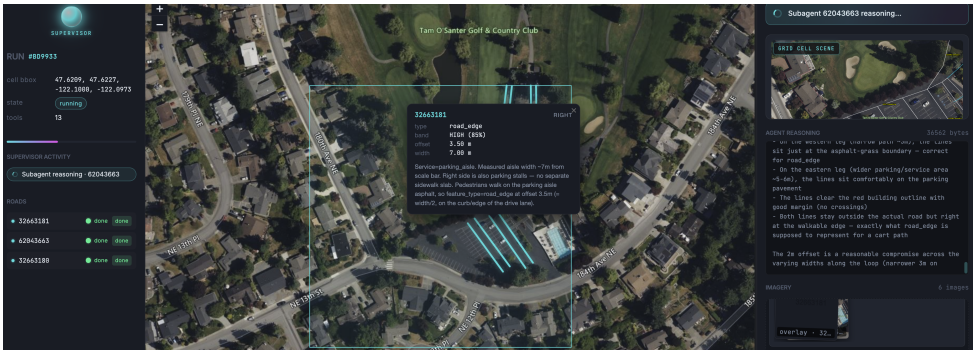
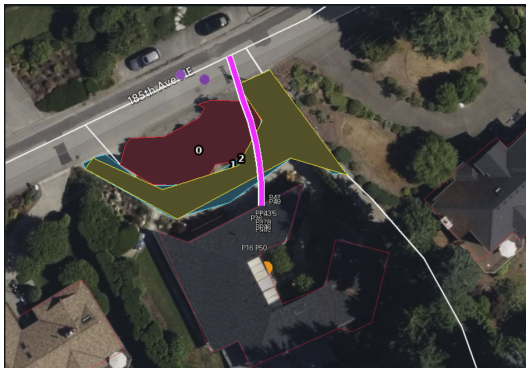


Figure 2: Supervisor–subagent interaction in the MapScout glass-box UI. The supervisor's scene decomposition and worklog (left), the live subagent reasoning and tool calls (right), and proposed features drawn live on the map.



(a) A subagent invokes a segmentation model (SAM 2) as a tool to detect a hazard zone (red) intersecting its proposed path.

(b) The agent self-corrects the proposed path (magenta) to avoid the hazard zones, biasing toward observed GPS trajectories (white).

Figure 3: Agents reason over imagery and invoke ML models as tools: hazard detection via segmentation (left) and GPS-biased path correction (right).

are corrected or flagged, so a locally plausible change that breaks reachability is caught before data is used by a routing engine [18].

Human commit. The self-corrected and flagged edits reach the editor, who sees each on the imagery with its labels, the validators' verdicts, and the agent's reasoning, then accepts, adjusts, or rejects it. This keeps the human as final authority and mirrors OpenStreetMap's norm of human review before acceptance, which the harness can enforce as a mandatory gate.

5 Demonstration

Attendees drive MapScout live over public data, one map tile at a time. The harness treats each tile as a single unit of work. The supervisor decomposes the tile and dispatches subagents that propose edits across it. The walkthrough has five steps.

(1) **Choose a tile** on a Leaflet map backed by satellite imagery and OpenStreetMap (all data public).

(2) **Dispatch agents and inspect their work.** The interface exposes both the supervisor and its subagents (the supervisor's scene decomposition and each subagent's reasoning and tool calls) and draws detections live as proposals are made (Figure 2), so attendees watch the scene being built.

(3) **Watch agents self-correct with validator tools.** Attendees see an agent revise after a check fails, such as a proposed sidewalk whose pixels are vegetation, caught by the segmentation tool (Figure 3), or an edit that would create a routing island, caught by the topology check.

(4) **Review and edit in the loop.** The corrected and flagged edits appear as editable Geoman features; the attendee accepts, adjusts, relabels, or rejects each, with a running tally displayed.

(5) **Commit and inspect** the approved edits as a sortable list with labels, confidence, verdicts, and the agent's reasoning behind each.

6 Operating Characteristics

MapScout's purpose is to lower the human editing cost, so we ask two questions: what is the token consumption by agents per tile, and how much cleaner is the set the editor finally reviews? We measure both over agentic runs on more than 1,000 residential tiles spanning several ZIP codes around Austin, Texas, across urban and

Table 1: Average token consumption per tile by density, for sidewalk and walk-path detection to building entrances.

Tile density	Turns	Images	In (K)	Out (K)
Urban	22	54	213	12
Suburban	28	68	310	18

Table 2: Proposal quality before and after validation, vs. auditor labels.

Condition	Precision	Recall
Raw agent proposals	40%	60%
After validate-and-revise loop	70%	74%

suburban blocks. Ground truth was annotated by a separate team of map auditors, independent of this work, who judge each feature from satellite imagery and sensor-derived evidence such as GPS traces. In all, the harness has generated walk-path geometry over roughly 1,640 such tiles, about 1,040 km^2 of the same region.

Human time per tile. On matched residential tiles of roughly 0.6 km^2 (~920 × 690 m), editing a tile's sidewalk and walk-path features (OpenStreetMap nodes, ways, and relations) by hand averaged 110 minutes per tile across multiple editors. With MapScout, the same editors reviewed the screened proposals and applied follow-up edits in 35 minutes, a 68% reduction (3.1× faster). The savings come from confirming a pre-checked set rather than authoring each feature.

Token consumption per tile (Table 1). We report mean per-tile token use, all rounds included, for runs detecting sidewalks and walk-paths to building entrances. Counterintuitively, suburban tiles require more agent turns than urban ones. A suburban tile contains one short walk-path per single-family home, and the agent reasons about each separately, whereas a dense block consolidates entrances onto a shared frontage.

Effect of validation (Table 2). Raw agent proposals match the auditor labels at only 40% precision and 60% recall; the validate-and-revise loop raises these to 70% and 74% (screening lifts precision, GPS-biased revisions recover recall). We choose this operating point deliberately: a committed sidewalk that does not exist can route a wheelchair user or pedestrian into a hazard, so residual error is asymmetric in cost and we prefer a 70% set behind a human gate to a stricter filter that also discards true positives.

7 Conclusion

MapScout shows that agents can propose candidate edits and labels, then use deterministic checks as tools to self-correct, leaving the human editor the final decision and shifting their workload from authoring to confirmation. The pattern applies wherever spatial data is edited or labeled under human authority. Only the OpenStreetMap path is demonstrated here, but portability is confined to two adapters—a context reader and a node/way/relation writer—since decomposition, the ML tools, the validators, and the review UI operate on plain geometry plus an attribute dictionary. Future work extends the roadmap tasks above (curb ramps, steps, and grade) and adds street-view imagery for features overhead views capture poorly; verified edits and harness logs can also serve as training data to improve the agents.

References

[1] Volodymyr Agafonkin. 2024. Leaflet: an open-source JavaScript library for interactive maps. https://leafletjs.com. Accessed 2026-06-27.

[2] Temitope Akinboyewa, Zhenlong Li, Huan Ning, and M. Naser Lessani. 2025. GIS Copilot: towards an autonomous GIS agent for spatial analysis. *International Journal of Digital Earth* 18 (2025). doi:10.1080/17538947.2025.2497489

[3] Amazon Web Services. 2025. Amazon SageMaker AI. https://aws.amazon.com/sagemaker/. Accessed 2026-06-27.

[4] Amazon Web Services. 2025. Strands Agents: an open-source SDK for building AI agents. https://strandsagents.com. Accessed 2026-06-27.

[5] Emre Eftelioglu, Gil Wolff, Sai Krishna Tejaswi Nimmagadda, Vishal Kumar, and Amber Roy Chowdhury. 2022. Deep classification of frequently-changing activities from GPS trajectories. In *Proceedings of the 15th ACM SIGSPATIAL International Workshop on Computational Transportation Science* (Seattle, Washington) *(IWCTS '22)*. Association for Computing Machinery, New York, NY, USA, Article 6, 10 pages. Dataset: https://github.com/amazon-science/goal-gps-ordered-activity-labels. doi:10.1145/3557991.3567784

[6] Geoman.io. 2024. Geoman: Leaflet plugin for drawing and editing geometry. https://geoman.io. Accessed 2026-06-27.

[7] Sean Gillies et al. 2024. Shapely: Manipulation and Analysis of Geometric Objects. https://github.com/shapely/shapely. Accessed 2026-06-27.

[8] Mordechai Haklay and Patrick Weber. 2008. OpenStreetMap: User-Generated Street Maps. *IEEE Pervasive Computing* 7, 4 (2008), 12–18. doi:10.1109/MPRV.2008.80

[9] Maryam Hosseini, Fabio Miranda, Jianzhe Lin, and Claudio T. Silva. 2022. CitySurfaces: City-scale semantic segmentation of sidewalk materials. *Sustainable Cities and Society* 79 (2022), 103630. doi:10.1016/j.scs.2021.103630

[10] Maryam Hosseini, Andres Sevtsuk, Fabio Miranda, Roberto M. Cesar Jr., and Claudio T. Silva. 2023. Mapping the walk: A scalable computer vision approach for generating sidewalk network datasets from aerial imagery. *Computers, Environment and Urban Systems* 101 (2023), 101950. doi:10.1016/j.compenvurbsys.2023.101950

[11] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of Hallucination in Natural Language Generation. *Comput. Surveys* 55, 12 (2023), 1–38. doi:10.1145/3571730

[12] Kartik Kuckreja, Muhammad Sohail Danish, Muzammal Naseer, Abhijit Das, Salman Khan, and Fahad Shahbaz Khan. 2024. GeoChat: Grounded Large Vision-Language Model for Remote Sensing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 27831–27840. arXiv:2311.15826.

[13] Potsawee Manakul, Adian Liusie, and Mark J. F. Gales. 2023. SelfCheckGPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 9004–9017. doi:10.18653/v1/2023.emnlp-main.557

[14] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. 2024. SAM 2: Segment Anything in Images and Videos. *arXiv preprint arXiv:2408.00714* (2024).

[15] Manaswi Saha, Michael Saugstad, Hanuma Teja Maddali, Aileen Zeng, Ryan Holland, Steven Bower, Aditya Dash, Sage Chen, Anthony Li, Kotaro Hara, and Jon E. Froehlich. 2019. Project Sidewalk: A Web-based Crowdsourcing Tool for Collecting Sidewalk Accessibility Data At Scale. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. doi:10.1145/3290605.3300292

[16] Michael Staniek, Raphael Schumann, Maike Züfle, and Stefan Riezler. 2024. Text-to-OverpassQL: A Natural Language Interface for Complex Geodata Querying of OpenStreetMap. *Transactions of the Association for Computational Linguistics* 12 (2024). doi:10.1162/tacl_a_00654

[17] Yanan Xiao, Rui Song, Yinan Xiao, Lu Jiang, Minghao Yin, and Pengyang Wang. 2025. UrbanXplain: A Language-Driven Urban Planning System with Explainable Reasoning and Real-Time 3D Rendering. In *Proceedings of the 33rd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL '25)*. doi:10.1145/3748636.3762792

[18] Yuxiang Zhang, Bill Howe, Sachin Mehta, Nicholas Bolten, and Anat Caspi. 2024. PathwayBench: Assessing Routability of Pedestrian Pathway Networks Inferred from Multi-City Imagery. *arXiv preprint arXiv:2407.16875* (2024).