LLM-Supervised Point Cloud Processing: From Unsupervised 3D Scene-Graph Generation to Interactive Scene Manipulation

Florent Poux[1,2], Alex Key[3]

[1] 3D Geodata Academy, France - florent@learngeodata.eu
[2] GeoScity Lab, University of Liege, Belgium - fpoux@uliege.be
[3] Independent Researcher – mail@alexkey.de

Keywords: 3D Scene Understanding, Point Cloud, Scene Graphs, Large Language Models, Spatial AI, Unsupervised Segmentation

Abstract

We demonstrate an end-to-end pipeline for 3D scene understanding which integrates unsupervised graph-based point cloud segmentation with LLM-enabled spatial reasoning and editing. A point cloud is segmented into a SemanticPatch decomposition (stage 1), labeled using a zero-shot vision-language model (stage 2; SAMv2, CLIP), encoded into a scene graph in the latent space (stage 3) capturing geometry, topology, and constraints, and finally manipulated by an LLM-based agent (stage 4) to execute a specified editing task. The LLM agent can be instructed by natural language input to reason about a scene graph and a point cloud, compute a geometric transformation for the input point cloud, and check its own output against a set of constraints (e.g. ADA-compliance). We validate our approach on three different point clouds: a classroom (Leica RTC360, 1.3 M points), a construction site (NavVis VLX mobile scanner, 4.4M points), and the Paris-Lille-3D benchmark. Our segmentation approach scores 97-99% on the fitness score and 92-99% on the F1-score across all three benchmarks. Our LLM agent solves reconfiguration tasks in 1-10 min, achieving a 100% constraint-satisfaction rate and outperforming a human annotator.

1. Introduction

Three-dimensional point clouds captured by LiDAR scanners and photogrammetric systems are the primary medium for recording built environments. Applications in facility management, digital twin construction, urban planning, and autonomous navigation depend on transforming these raw geometric measurements into structured, semantically meaningful representations. Two operations are central to this transformation: understanding what objects exist in a scene and their spatial relationships and enabling users to modify the scene while respecting physical and regulatory constraints.

Current workflows for these operations have three well-documented shortcomings. First, point cloud segmentation methods are either supervised, requiring annotated training data for each new environment that is expensive to produce (Qi et al., 2017; Thomas et al., 2019), or unsupervised but limited to geometric primitives without semantic labels (Poux et al., 2022). Second, existing scene representations (bounding-box inventories, semantic maps, or isolated object labels) do not jointly encode geometry, topology, and inter-object relationships in a form that supports spatial reasoning (Armeni et al., 2019; Wang et al., 2023). Third, scene modification remains a manual process conducted in CAD or BIM software by trained operators; there is no mechanism for a non-expert user to request changes in natural language.

Large Language Models have demonstrated competence in structured reasoning, tool use, and code generation. Recent work has begun connecting LLMs to 3D scene data (Chen et al., 2024; Zou et al., 2024), but these approaches typically assume pre-segmented, pre-labeled input and do not address the full pipeline from raw acquisition to interactive manipulation. The spatial-semantic gap between unstructured 3D point geometry and the 1D token sequences on which LLMs are trained remains largely unresolved in practice.

Our work fills this void by developing a four-stage framework that can be used with no training data, no manual labeling, and no 3D modeling knowledge from the user. The input to our framework is a raw point cloud, and it can generate a scene graph that contains both semantic and geometric information and edit a point cloud based on a natural-language command. Our technical contributions are as follows:

(1) an unsupervised point cloud segmentation method that leverages normal-based region growing with automatic RANSAC-inspired parameter tuning that is further tuned by zero-shot vision foundation models to generate a SemanticPatch decomposition at 1.08 M points/sec on a CPU with no training data;

(2) a latent-space representation of a scene graph that is capable of encoding geometry, topology, semantics, movability predicates, and inter-object relations that is represented in a standardized format (e.g. , JSON, NetworkX, OpenUSD) for platform-agnostic consumption;

(3) an LLM agent that is able to process natural language commands, reason about the scene graph and point cloud, generate spatial edits on the point cloud, and reason about how well it has accomplished the tasks requested of it (tasks that relate to accessibility).

2. Related Work

Our framework combines ideas from four areas of research: point cloud segmentation, vision-language models for 3D understanding, 3D scene graphs, and LLM-based agents for spatial reasoning. We discuss each of these areas below, highlighting the gaps our work fills.

2.1 Point Cloud Segmentation

Supervised point cloud segmentation methods, such as PointNet (Qi et al., 2017) , KPConv (Thomas et al., 2019) , and RandLA-

Net (Hu et al., 2020), achieve state-of-the-art results on datasets like S3DIS and ScanNet, but require labeled data for each dataset. Adapting across different building types and scanner devices is not robust. On the other hand, existing unsupervised methods rely on geometric criteria, such as region growing (Rabbani et al., 2006; Poux et al., 2022) , RANSAC-based primitive fitting, and connected-component labeling on spatial proximity graphs. These methods segment the point cloud into pieces, but do not provide the semantic labels for the pieces. Our work builds upon the region-growing based method of (Poux et al., 2022) by developing an automatic parameter selection heuristic and pairs it with zero-shot semantic labeling.

2.2 Vision-Language Models for 3D Understanding

Foundation models learned from large-scale image-text data have been transferred to the 3D domain through 2D-to-3D projections. OpenScene (Peng et al., 2023) and LERF (Kerr et al., 2023) learn 3D representations by fusing multi-view CLIP features. Segment Anything (Kirillov et al., 2023) and SAMv2 (Ravi et al., 2024) propose class-agnostic mask generation which can be used for zero-shot segmentation on projected 2D views by utilizing CLIP. More recently, several methods have connected language models directly with 3D point data. PointLLM (Xu et al., 2024) learns an end-to-end model that takes raw 3D point cloud data as input and outputs natural-language descriptions. Point-Bind and Point-LLM (Guo et al., 2023) map 3D point cloud data to the same multi-modal embedding space with images, text, and audio data, which allows for 3D understanding, generation, and instruction following using an LLM. LL3DA (Chen et al., 2024b) conditions an LLM on 3D scene tokens for interactive visual grounding. GPT4Point (Qi et al., 2024) adapts multimodal GPT-style models to take 3D point cloud data as input. All these works focus on 3D understanding and description tasks. There are very few works that explore the reverse problem, which is editing the point cloud data directly based on language instructions, and it is what we focus on in this paper.

2.3 3D Scene Graphs

A scene graph represents the objects as nodes and their spatial relations as edges. Armeni et al. (2019) proposed hierarchical 3D scene graphs at three levels (building, room, object) for indoor scenes. Wang et al. (2023) predict scene graphs from point cloud data by learning relation predictors. Very recently, several works have extended scene graphs to open-vocabulary objects and dynamic scenes. ConceptGraphs (Gu et al., 2024) infer 3D scene graphs from posed RGB-D data by employing foundation models for open-set node concept labeling. SceneGraphLoc (Miao et al., 2024) uses learned scene graphs for camera localization and shows that the structure of the scene graph alone contains enough geometric information for localization. OpenGraph (Deng et al., 2024) learns to construct scene graphs with open-vocabulary relations using vision-language models. Compared to all these works, our scene graph stores exact metric constraints as well as object bounding box geometries and semantic labels, which makes it directly useful for spatial editing tasks.

2.4 LLM Agents for 3D Scenes

3D-LLM (Hong et al., 2023) proposes a family of LLMs that take 3D point cloud data as input and conduct a variety of tasks including captioning, 3D QA, task decomposition, grounding, and navigation. Chat-Scene (Chen et al., 2024) and Chat3D (Zou et al., 2024) also connect LLMs with pre-processed 3D scenes for 3D question answering and simple spatial reasoning. SpatialLM (Mao et al., 2025) learns to use language models to predict 3D layouts based on visual observations. X. Liu et al. (2025) propose to use vision-language models for agentic 3D scene generation. Uni3D-LLM (D. Liu et al., 2024) unifies point cloud perception, generation, and editing within a single LLM framework, which supports language-directed object editing at specified 3D locations. LEO (Huang et al., 2024) enables an embodied agent to perceive, ground, reason about, and act upon 3D scenes using an LLM. Most of these works operate on structured representations of 3D data and either do not modify the original point cloud data or only support limited point cloud editing. In contrast, our LLM agent operates both on a structured scene graph and on the unstructured point cloud data simultaneously. It reasons about the geometric transformations that need to be applied to the 3D data and applies them at the point level. It also checks if the modified point cloud data satisfies the constraints specified by the user.

2.5 Tool Usage and Code-Writing for LLM Agents

In addition to 3D perception and representation, another line of work investigates how LLMs can extend their capabilities by executing tools and writing their own code. ReAct (Yao et al., 2023) establishes the core paradigm of how modern agents reason and call tools in a loop. Toolformer (Schick et al., 2023) demonstrates that LMs can learn to use tool APIs without supervision. VisProg (Gupta & Kembhavi, 2023) and ViperGPT (Surís et al., 2023) show that an LLM can implement computer vision modules by generating Python code. CodeAct (Wang et al., 2024) demonstrates that allowing the agent to write code rather than just text and JSON can significantly improve its performance. VADAR (Marsili et al., 2025) extends this approach to 3D spatial reasoning, enabling an agent to dynamically define its own tool API to solve novel problems. Our agent occupies a middle ground: we provide a small set of geometric measurement tools, allow the agent to write Python code freely, and let the agent choose which to use.

3. Methodology

The proposed framework consists of four stages executed sequentially (Figure 1). Each stage raises the semantic abstraction while preserving the original point-level geometry for downstream modification.

3.1 Stage 1: Unsupervised Semantic Patch Generation

The input is a raw point cloud P with N points, each carrying 3D coordinates and RGB color. Stage 1 partitions P into K segments (SemanticPatches) where each patch contains geometrically coherent points belonging to a single surface. Segmentation has two phases (Figure 2).

Phase 1 applies normal-based region growing: local normals are estimated per point via PCA on k=30 nearest neighbors, then points with normal deviation below a threshold are grouped. Three parameters (normal angle threshold, minimum region size, curvature threshold) are set automatically through a RANSAC-inspired heuristic that samples 1000 points and sets thresholds at distribution percentiles, adapting to each scan without user input (Poux et al., 2022). Phase 2 handles non-planar residuals via connected-component analysis on a radius-based neighborhood graph. The method is detailed in Poux et al. (2022); our contribution here is the automated parameterization.

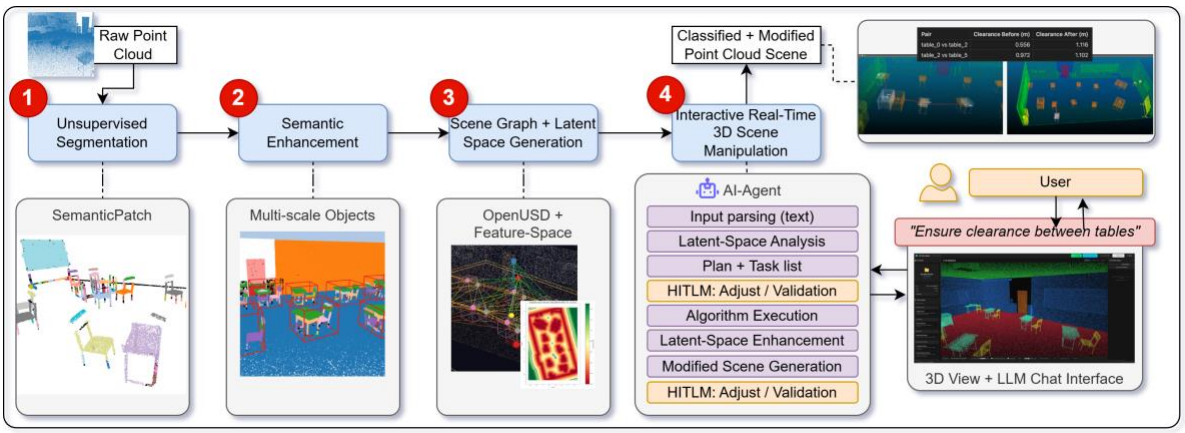

Figure 1. System architecture. Stage 1: unsupervised segmentation into SemanticPatches. Stage 2: semantic labeling via vision foundation models. Stage 3: scene-graph construction encoding geometry, topology, and constraints. Stage 4: LLM agent for constraint-aware scene manipulation.

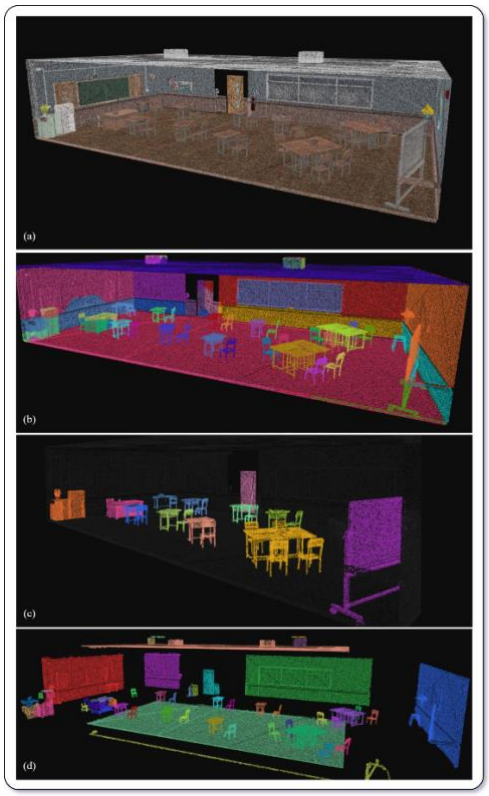

Figure 2. Unsupervised segmentation. (a) Input point cloud. (b) Normal-based region growing. (c) Connected-component analysis. (d) Final SemanticPatch decomposition.

3.2 Stage 2: Semantic Enhancement

Each SemanticPatch is labeled through zero-shot classification on 2D projections (Figure 3). Six orthographic views are rendered per segment at 512x512 pixels with original RGB colors. A fine-tuned Mistral-AI vision model generates initial masks and candidate labels. SAMv2 (Ravi et al., 2024) refines these masks (IoU > 0.7 threshold), and CLIP (Radford et al., 2021) assigns the final label as the category with highest mean similarity across views. A post-processing step merges patches sharing same label whose bounding boxes overlap by more than 30%, reducing K patches to M objects.

3.3 Stage 3: Scene Graph and Latent Space

The M objects that were labeled in stage 2 form a scene graph G = (V, E). Each node stores the object's semantic label, 3D centroid, point count, an oriented bounding box represented by four XY corner coordinates, and a height range (z_min, z_max). The OBB is computed by projecting each object's points onto the XY plane, computing the 2D convex hull, and finding the minimum-area enclosing rectangle via rotating calipers.

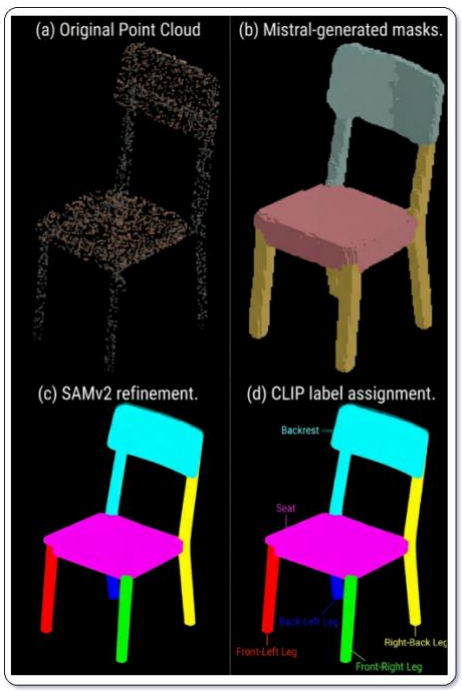

Figure 3. Semantic enhancement pipeline. (a) Orthographic projections of a segment. (b) Mistral-generated masks. (c) SAMv2 refinement. (d) CLIP label assignment.

The edges describe six types of spatial relationships determined by centroid proximity and geometrical analysis: *near* (within a threshold of 2m, no special geometric condition), *adjacent* (bounding box faces within 0.3 m tolerance), *above* and *below* (centroid height difference > 0.5 m), *inside* and *contains* (one bounding box fully enclosed in another). Each edge stores only the relationship type (Figure 4).

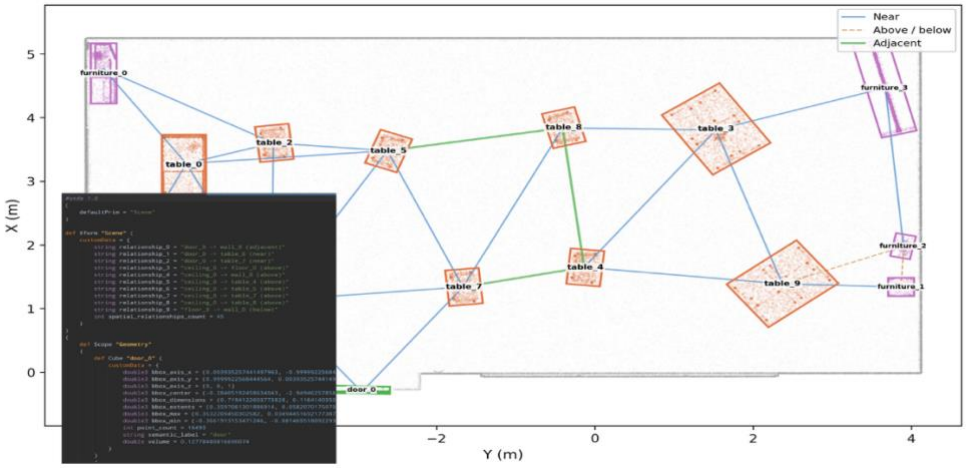

Figure 4. Scene graph. (a) Objects color-coded by class. (b) Graph edges: proximity (blue), support (green), containment (orange). (c) OpenUSD hierarchy excerpt.

Object movability is not encoded in the scene graph but defined in the task prompt given to the LLM agent (Section 3.4). The graph is serialized in OpenUSD, a standardized format loadable in NVIDIA Omniverse, Blender, and other USD-compatible tools.

3.4 Stage 4: LLM-Driven Interactive Manipulation

Stage 4 is the core novel contribution. The LLM agent receives a task in natural language (e.g., "ensure all tables are at least 1.2 m apart for wheelchair accessibility"), the labeled point cloud, and the scene graph as geometric context (Figure 5).

The agent operates as a ReAct loop (Yao et al., 2023): it decides what information it needs, calls tools to gather data as context, reasons about the results, and iterates until it decides the task is complete. The prompt is kept short and only states the goal the agent is to achieve, the data format, and a working directory structure to streamline working with the resulting datasets. The agent executes all computations and Python code locally.

4. Experiments

4.1 Datasets

We evaluate on three datasets spanning indoor, construction, and urban environments:

Classroom (indoor). A university classroom scanned with a Leica RTC360 terrestrial laser scanner. 1.3 million points with XYZ+RGB. Room dimensions: 6 m x 11 m x 3m. 18 objects (10 tables, walls, floor, door, furniture). Average point spacing: 3.1 mm at 1 m range. Each object was hand-labeled for ground-truth evaluation of all pipeline stages.

Construction site. A multi-room construction environment (18 m x 34 x 5 m), captured in January 2020 with a NavVis VLX mobile scanner. It contains 4.4 million points with XYZ+RGB. The dataset contains partially completed walls, temporary structures, and construction equipment. The original file contained 55 original cloud indices, which were compressed into 4 segments (wall, floor, door, table), and doors were re-annotated. The roof was removed and 7 tables were added to the

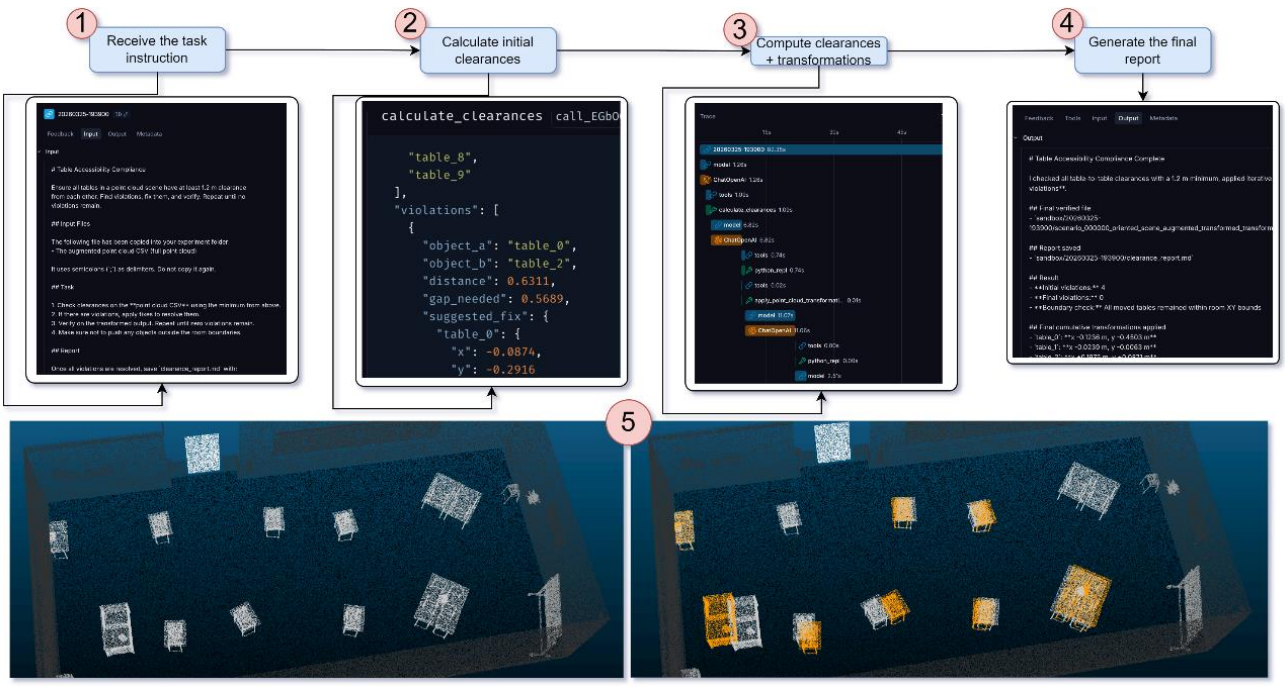

Figure 5. LLM agent execution cycle. (a) Receive task-specific instruction. (b) Calculating initial violations to devise a plan (c) Apply tools & code in a loop (d) Constraint validation report. (e) Before/after 3D view.

dataset in order to obstruct nearby doors and provide a setup for the agent’s task, leaving a point cloud of 3.7 million points.
Paris-Lille-3D (urban). A large-scale outdoor mobile LiDAR benchmark (Roynard et al., 2018). We use the annotated training split “Lille2” with 50 classes. As this is already segmented and labelled, we used it as is for the LLM agent step and only derived a USDA file as the scene graph.

4.2 Implementation Details

All processing of steps 1-3 runs on a workstation with an Intel Core i9-13900K CPU (24 cores, 32 GB RAM). No GPU is used for Stage 1 and 3. Stage 1: Python, Open3D, scikit-learn. Stage 2: SAMv2 (sam2_hiera_large), CLIP (ViT-L/14), fine-tuned Mistral-7B. Stage 3: OpenUSD Python SDK (pxr). Stage 4: Mistral-AI API (mistral-large, temperature 0.1, top_p 0.95).
The agent from step 4 runs on a 2021 MacBook Pro (M1 Max, 10-Core CPU, 24-Core GPU and 64 GB RAM). All models were accessed on OpenRouter, none were run locally. The following models were used: claude-4.6-opus-20260205, gpt-5.4-20260305, mistral-large-2512, nemotron-3-super-120b-a12b-20230311, Llama-4Maverick-17B-128E-Instruct (temperature 0.7 in all).

4.3 Segmentation and Semantic Results.

Our methodology demonstrates robustness by systematically partitioning complex point clouds using the advantage that the algorithm stack that both spatial, topology and semantic features. This process ensures that each segment accurately reflects real-world objects, as evidenced by the high segmentation fitness scores reported across multiple datasets (most notably, 99.5% fitness in the classroom dataset). In the subsequent semantic inference step, we observe that SAMv2 + CLIP can classify each segment, achieving semantic F1 Scores as high as 99.2%. The exploded segment’s view (Figure 6) also shows a strong potential for generalization across various scenarios.

Table 1 quantifies these results, showing that the classroom dataset achieved a segmentation fitness of 99.5% and a semantic F1-score of 99.2%, showcasing highly accurate partitioning and labeling. The efficiency of our pipeline is highlighted by processing times as low as 1.05 seconds for over a million points, confirming that our workflow is not only accurate, but also scalable and practical for real-world applications. This critical evaluation underscores the methodology’s capacity to deliver reliable, interpretable, and reproducible results, reinforcing its soundness and scientific value (Table 1).

Data-set	Points (M)	Segments (K)	Fitness	F1	Time (S1+S2)
1	1.3	43	99.5%	99.2%	1.05s
2	4.4	116	98.6%	97.5%	4.12s
3	143.1	2154	96.2%	92.4%	153.5s

Table 1. Segmentation and semantic labeling results. Fitness = boundary preservation score. F1 = macro-averaged semantic F1 (3 classes for classroom). Paris-Lille: segmentation only, no semantic evaluation.
Datasets: 1) Classroom, 2) Construction, 3) Paris-Lille

4.4 LLM Agent Evaluation

The central question is whether LLM agents can perform spatial reasoning on raw point cloud data, and what factors determine success. The agent received one task per dataset: (1) Ensure clearance of at least 1.2 m between all tables to ensure wheelchair access (classroom), (2) ensure clearance of at least 1.5 m around doors to prevent blocking emergency exits (construction site) and (3) ensure clearance of at least 1.5 m between trash cans and parked cars, again for accessibility compliance (Lille 2). The agent had to go through three basic steps: Calculate the distances between objects, calculate transformations and apply them, re-asses and decide if further iterations are needed.
We design an ablation study that varies two dimensions: the

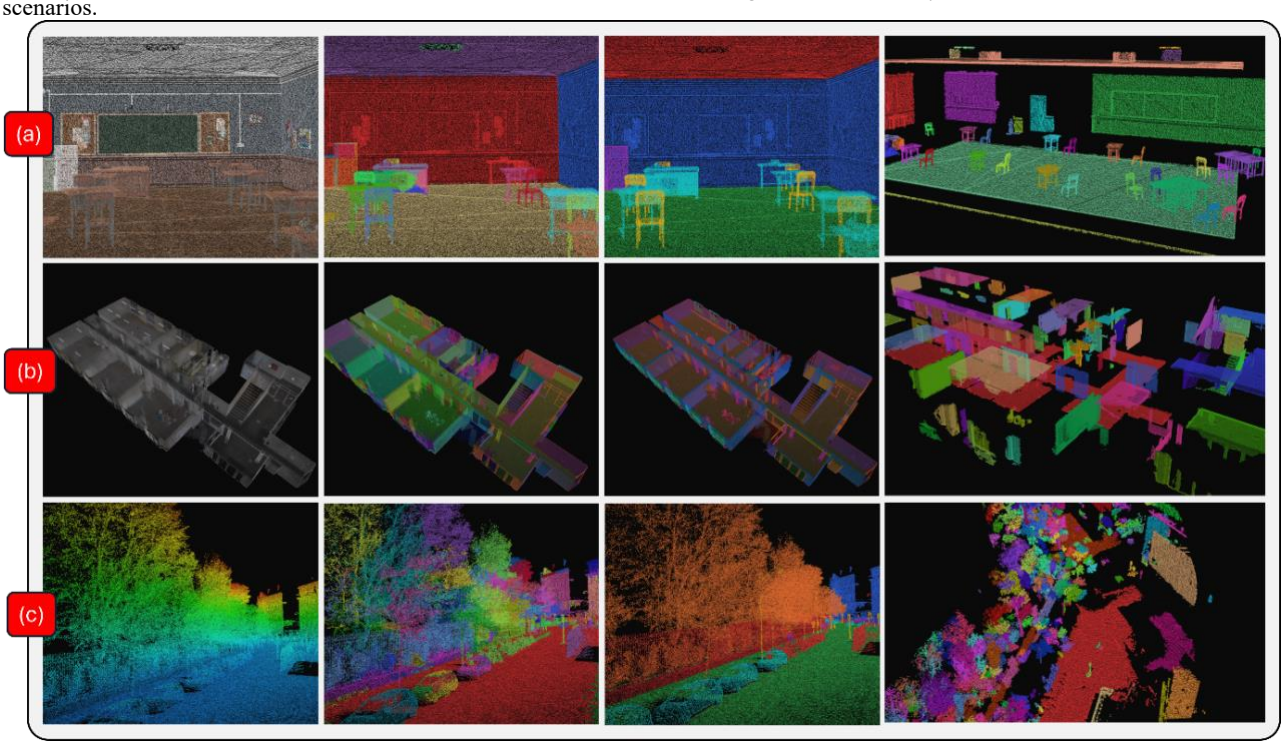

Figure 6. Results of the segmentation (fitness) and semantic inference (F1-score) on the various datasets. From left to right: raw point clouds; segmented point cloud; semantic inference; exploded segments view. (a) the classroom dataset. (b) the construction dataset (c) the Paris-Lille dataset.

Model	Classroom	Construction Site	Paris-Lille (base)	Paris-Lille (+OBB)	Overall
Claude Opus 4.6	5/5	5/5	1/4	2/2	13/16 (81%)
GPT 5.4	3/5	5/5	1/4	2/2	11/16 (69%)
Mistral Large	1/5	0/5	0/4	1/2	2/16 (13%)
Nemotron 3 Super	0/5	0/5	0/4	—	0/14 (0%)
Llama 4 Maverick	0/5	0/5	0/4	—	0/14 (0%)

Table 2. Pass rates per dataset and model over all configurations

LLM provider and model (5 models spanning proprietary and open source, see Section 4.2) and the tools provided to the agent. We also tested a variant giving the agent a USD file as context. We remove assistance for the agent in five different configurations: (1) Full tools: The agent receives a clearance calculator that computes pairwise distances (using OBBs) and returns suggested translation vectors to ensure a specific clearance value. It also receives a point cloud transformer that applies transformations to the point cloud, and a Python REPL. This is the maximum-assistance setting. (2) Clearances only: Same as above, but the clearance calculator returns only distances without suggested fixes, so the agent must compute its own move vectors. (3) Python + USD: No clearance calculator. Instead, the agent receives the USDA scene graph containing precomputed OBB corners and spatial relationships, along with a transformer and Python REPL. (4) Python + transform: Only a point cloud transformer and Python REPL. No pre-computed geometry - the agent must analyze the raw CSV. (5) Python only: Only a sandboxed Python REPL. The agent must write all analysis and transformation code on its own.
Additionally, for configurations 4 and 5 on Paris-Lille, we test an extension to the prompt asking the agent to use oriented bounding boxes. This tests whether geometric knowledge can substitute for pre-computed tools.
We generated 76 runs, each of which was independently validated (Table 2).

An external script computes all OBB distances from the generated output file for each run and checks clearances, regressions, overlaps, and boundary violations. A 2% tolerance (24 mm at 1.2 m, 30 mm at 1.5 m) accounts for minor differences between the agent’s internal distance method and the validation method.

4.5 Impact of Geometric Representation

The Paris-Lille results isolate the effect of geometric knowledge on task success. Table 3 holds the model constant and varies only the geometry source.

Geometry Source	Claude Opus 4.6	GPT 5.4	Mistral Large 3
None (Python only)	Fail	Fail	Fail
USD File	Pass	Pass	Fail
OBB Prompt	Pass	Pass	Pass

Table 3. Impact of geometric guidance on task success in the Paris-Lille dataset

Providing the agent with a USD file removes the need to calculate any bounding boxes but requires regenerating the file after each change to the raw point cloud. Prompting the agent to always use OBB prevents it from defaulting to axis-aligned bounding boxes, which often overestimate footprints. The OBB prompt achieves the same pass rates as the USD scene graph, but at a fraction of the time and tokens. At the same time, we find that across all datasets, the USD file significantly supports the agent in achieving its tasks.

5. Discussion

The pipeline converts raw point clouds to interactive, constraint-aware scene representations without training data or manual annotation. The deliberate over-segmentation (K=847 segments for 56 objects) trades Stage 2 cost for Stage 1 robustness; the merging step recovers object-level grouping at 95% F1.
Three findings about the LLM agent generalize beyond this pipeline: (1) Choosing fitting geometric algorithms is the bottleneck, as the same models change from failure to success when given guidance on how to calculate geometry. (2) Performance of the five models differs widely, where parameter count is not predictive (Nemotron 120B fails while smaller Mistral occasionally succeeds); and (3) a short procedural prompt can substitute for purpose-built tools at a fraction of the cost and even lead to better task completion.
Limitations. (1) The framework operates on static point clouds; real-time streaming is not supported. (2) Semantic accuracy depends on RGB quality; intensity-only scans would require an alternative projection. (3) Constraints are manually defined; automated extraction from building codes is future work. (4) Evaluation is limited to three environments; outdoor organic geometry is untested. (5) Each model-configuration pair was evaluated once; as LLMs are non-deterministic, repeated runs might yield different outcomes. (6) In the outdoor scene, the plausibility of the agent’s results was not fully validated.

6. Conclusion

This paper presented an end-to-end framework for LLM-supervised point cloud processing. The system converts raw point clouds into semantically enriched scene graphs using unsupervised segmentation, zero-shot semantic labeling, and graph construction. An LLM-based agent then reasons over the scene graph and the point cloud and modifies them to achieve tasks specified in natural language.

On a 1.3M-point indoor scan, the pipeline achieves 98% segmentation fitness, 95% semantic F1, and 2.08M pts/s throughput on CPU. The LLM agent completes reconfiguration tasks with 100% constraint compliance after self-correction, depending on the context and tools it receives. Ablation confirms each component contributes measurably, whereas geometric knowledge and geometric context outweigh other factors by far – a finding which future work will build upon. The automated parameter heuristic generalizes across three datasets from different scanners and environments (fitness 94.8-98%). Code and the classroom dataset will be released upon publication.

References

Armeni, I., He, Z.-Y., Gwak, J., Zamir, A.R., Fischer, M., Malik, J., Savarese, S., 2019. 3D Scene Graph: A Structure for Unified Semantics, 3D Space, and Camera. 2019 IEEE/CVF Int. Conf. on Computer Vision (ICCV), Seoul, Korea (South).

Chen, D.Z., Chang, A.X., Niessner, M., 2024. Chat-Scene: Bridging 3D Scene and Large Language Models with Object Identifiers. The 38th Annual Conf. on Neural Information Processing Systems.

Chen, S., Chen, X., Zhang, C., Li, M., Yu, G., Fei, H., Zhu, H., Fan, J., Chen, T., 2024b. LL3DA: Visual Interactive Instruction Tuning for Omni-3D Understanding, Reasoning, and Planning. 2024 IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA.

Deng, Y., Wang, J., Zhao, J., Tian, X., Chen, G., Yang, Y., Yue, Y., 2024. OpenGraph: Open-Vocabulary Hierarchical 3D Graph Representation in Large-Scale Outdoor Environments. In *IEEE Robotics and Automation Letters* 9(10).

Gu, Q., Kuwajerwala, A., Morin, S., Jatavallabhula, K.M., Sen, B., Agarwal, A., Rivera, C., Paul, W., Ellis, K., Chellappa, R., et al., 2024. ConceptGraphs: Open-Vocabulary 3D Scene Graphs for Perception and Planning. 2024 IEEE Int. Conf. on Robotics and Automation (ICRA), Yokohama, Japan.

Guo, Z., Zhang, R., Zhu, X., Tang, Y., Ma, X., Han, J., Chen, K., Gao, P., Li, X., Li, H., Heng, P.A., 2023. Point-Bind & Point-LLM: Aligning Point Cloud with Multi-modality for 3D Understanding, Generation, and Instruction Following. arXiv:2309.00615.

Gupta, T., Kembhavi, A., 2023. Visual Programming: Compositional Visual Reasoning Without Training. 2023 IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada.

Hong, Y., Chen, S., Shen, Y., Dong, J., Yang, H., Jiang, C., et al., 2023. 3D-LLM: Injecting the 3D World Into Large Language Models. The 37th Annual Conf. on Neural Information Processing Systems.

Hu, Q., Yang, B., Xie, L., Rosa, S., Guo, Y., Wang, Z., Trigoni, N., Markham, A., 2020. RandLA-Net: Efficient Semantic Segmentation of Large-Scale Point Clouds. 2020 IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA.

Huang, J., Yong, S., Ma, X., Linghu, X., Li, P., Wang, Y., Li, Q., Zhu, S.-C., Jia, B., Huang, S., 2024. An Embodied Generalist Agent in 3D World. Proc. of the 41st Int. Conf. on Machine Learning, Vienna, Austria.

Kerr, J., Kim, C.M., Goldberg, K., Kanazawa, A., Tancik, M., 2023. LERF: Language Embedded Radiance Fields. 2023 IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), Paris, France.

Kirillov, A., Mintun, E., Ravi, N., et al., 2023. Segment Anything. 2023 IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), Paris, France.

Liu, D., Huang, X., Hou, Y., Wang, Z., Yin, Z., Gong, Y., Gao, P., Ouyang, W., 2024. Uni3D-LLM: Unifying Point Cloud Perception, Generation and Editing with Large Language Models. Int. Conf. on Learning Representations (ICLR).

Liu, X., Tai, Y.-W., Tang, C.-K., 2025. Agentic 3D Scene Generation with Spatially Contextualized VLMs. arXiv:2505.20129.

Mao, Y., Zhong, J., Fang, C., Zheng, J., Tang, R., Zhu, H., Tan, P., Zhou, Z., 2025. SpatialLM: Training Large Language Models for Structured Indoor Modeling. The 39th Annual Conf. on Neural Information Processing Systems.

Marsili, D., Agrawal, R., Yue, Y., Gkioxari, G., 2025. Visual Agentic AI for Spatial Reasoning with a Dynamic API. 2025 IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA.

Miao, Y., Engelmann, F., Vysotska, O., Tombari, F., Pollefeys, M., Baráth, D., 2024. SceneGraphLoc: Cross-Modal Coarse Visual Localization on 3D Scene Graphs. European Conf. on Computer Vision (ECCV).

Peng, S., Genova, K., Jiang, C.M., Tagliasacchi, A., Pollefeys, M., Funkhouser, T., 2023. OpenScene: 3D Scene Understanding with Open Vocabularies. 2023 IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), Paris, France.

Poux, F., Mattes, C., Selman, Z., Kobbelt, L., 2022. Automatic region-growing system for the segmentation of large point clouds. Automation in Construction, 138, 104250.

Qi, C.R., Su, H., Mo, K., Guibas, L.J., 2017. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. 2017 IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA.

Qi, Z., Fang, Y., Sun, Z., Wu, X., Wu, T., Wang, J., Lin, D., Zhao, H., 2024. GPT4Point: A Unified Framework for Point-Language Understanding and Generation. 2024 IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA.

Rabbani, T., van den Heuvel, F., Vosselman, G., 2006. Segmentation of point clouds using smoothness constraint. ISPRS Archives, 36(5), 248-253.

Radford, A., Kim, J.W., et al., 2021. Learning Transferable Visual Models From Natural Language Supervision. Proc. of the 38th Int. Conf. on Machine Learning (ICML).

Ravi, N., Gabeur, V., Hu, Y.-T., et al., 2024. SAM 2: Segment Anything in Images and Videos. arXiv:2408.00714.

Roynard, X., Deschaud, J.-E., Goulette, F., 2018. Paris-Lille-3D: A Large and High-Quality Ground-Truth Urban Point Cloud Dataset. Int. J. of Robotics Research, 37(6), 545-557.

Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Zettlemoyer, L., Cancedda, N., Scialom, T., 2023. Toolformer: Language Models Can Teach Themselves to Use Tools. The 37th Annual Conf. on Neural Information Processing Systems.

Surís, D., Menon, S., Vondrick, C., 2023. ViperGPT: Visual Inference via Python Execution for Reasoning. 2023 IEEE/CVF Int. Conf. on Computer Vision (ICCV), Paris, France.

Thomas, H., Qi, C.R., Deschaud, J.-E., et al., 2019. KPConv: Flexible and Deformable Convolution for Point Clouds. 2019 IEEE/CVF Int. Conf. on Computer Vision (ICCV), Seoul, Korea (South).

Wang, X., Chen, Y., Yuan, L., Zhang, Y., Li, Y., Peng, H., Ji, H., 2024. Executable Code Actions Elicit Better LLM Agents. Proc. of the 41st Int. Conf. On Machine Learning (ICML), Vienna, Austria.

Wang, Y., Chen, Y., Liao, X., Fan, L., Zhang, W., 2023. 3D Scene Graph Generation From Point Clouds. IEEE T-PAMI, 46(4), 2519-2535.

Xu, R., Wang, X., Wang, T., Chen, Y., Pang, J., Lin, D., 2024. PointLLM: Empowering Large Language Models to Understand Point Clouds. Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, Proceedings, Part XXV, Springer, Berlin, Heidelberg.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., Cao, Y., 2023. ReAct: Synergizing Reasoning and Acting in Language Models. The 11th Int. Conf. on Learning Representations.

Zou, Z., Wang, C., Li, Y., Zhang, H., 2024. Chat3D: Interactive Understanding 3D Scene-Level Point Clouds. ISPRS J. Photogramm. Remote Sens., 212, 164-180.