Contents lists available at ScienceDirect

Measurement: Digitalization

journal homepage: www.elsevier.com/locate/meadig

Language-guided zero-shot segmentation with multi-angle reprojection for point cloud analysis

Abiodun Ayodeji [a,*], Ahmed Teyeb [a], Mohmmad Ali Asgar Abbas [a], Paul Bass [b], Emma Bass [b], Prasanna D. Bandara [b], Udari K. Jayasinghe [b], Jamie Griffiths [b], Evelyne El Masri [a]

[a] Brunel Innovation Centre, Brunel University London, Uxbridge UB8 3PH, UK
[b] Tricore Technical Services, Ltd, Villa Jubilant, 34 Falcon Ct, Stockton-on-Tees TS18 3TX, UK

ARTICLE INFO

Keywords:
Virtual reality
Zero-shot 3D object detection
Point cloud labelling
Foundation models
Language-guided segmentation
GroundingDINO
Segment anything model (SAM)
Semantic Labelling
Confidence-weighted Fusion
Augmented Reality

ABSTRACT

Virtual Reality applications increasingly demand accurate 3D representations of real-world environments. While LiDAR point clouds capture physical spaces with high fidelity, they typically lack semantic labels, limiting their direct use for tasks such as object recognition, interaction modeling, and automation in immersive environments or digital twin systems. We present a LAnguage-guided zero-shot 3D SEgmentation and Reprojection tool (LASER), an engineered zero-shot segmentation tool that extends the state of the art by introducing language-guided 3D object detection for enhanced usability and accuracy. Unlike its predecessors, LASER uses an ensemble of GroundingDINO and Segment Anything Model as its backbone to process natural language queries and user-specified object categories, automated multi-view orthophoto generation with dynamic angles for optimal view selection, a confidence-weighted fusion algorithm for efficient 2D-3D reprojection, and a semantically labelled mesh output.

The LASER pipeline begins by collecting point cloud data using LiDAR sensors, filtering the point cloud into ground and non-ground components, improving segmentation efficiency. It then generates multi-angle 2D orthophotos and perspective views, incorporating a user-guided angle selection module to optimise scene coverage. Then GroundingDINO detects objects based on textual descriptions, and Segment Anything Model subsequently refines these into segmentation masks. The core innovation of LASER lies in its confidence-weighted reprojection algorithm, which fuses multiple 2D segmentation results back into 3D space, ensuring higher segmentation accuracy and spatial consistency. The resulting semantically labelled assets can be exported in standard formats or iteratively refined through viewpoint adjustments or text prompt modifications.

Our application of LASER to real-world 3D scans of construction sites demonstrates its effectiveness in delivering high segmentation precision, enhanced user interactivity, and seamless integration into virtual reality workflows. To comprehensively evaluate the proposed tool on diverse point cloud scans, we also presented the performance on four different test cases using two different scans (3DSES and Toronto3D) with both indoor and outdoor scenes. The results show consistent performance across scans. Finally, feature-based comparison with state-of-the-art approaches shows that LASER is an optimised tool for enriching static, open-world 3D scans with semantic labels, offering an alternative to existing state-of-the-art methods for niche applications.

1. Introduction

A static three-dimensional (3D) point cloud is typically derived from LiDAR scans at varying scales and resolutions [1]. In many industrial and urban applications, point cloud data provide a dense geometric representation of the environment, capturing geometric features and spatial context [2]. However, point cloud data are essentially collections of points in 3D space. Unlike images or videos where objects are often visually distinguishable and can be more easily labelled, point cloud data do not inherently carry semantic meaning about what the points represent. A cluster of points could represent a wall, a piece of furniture, or any other object, but this information isn't automatically included in the raw point cloud data. This lack of semantic labelling creates limitations for applications in areas such as immersive Virtual Reality (VR)

* Corresponding author.
E-mail address: ayod_abe@yahoo.com (A. Ayodeji).

https://doi.org/10.1016/j.meadig.2025.100016
Received 12 June 2025; Received in revised form 3 September 2025; Accepted 10 September 2025
Available online 10 September 2025

scene design, interactive environment modelling, and digital twin applications [3].

Beyond visualisation, semantic labelling of 3D point clouds provides critical affordances for downstream applications in VR and digital twin environments. By attaching class-specific labels, objects can be given physics or interaction hooks, enabling richer simulation, gaming and safety training scenarios. In construction safety training for instance, semantics allow region-of-interest triggers, such as automatically highlighting safety-critical objects within a prescribed distance of a hazard. In digital twin pipelines, semantic segmentation supports automated asset replacement, for example substituting a scanned scaffold with a corresponding building information management (BIM) element, and enables rule-based monitoring such as enforcing clearance or occupancy regulations. Semantic labelling further facilitates change detection, where newly segmented elements can be compared to historical scans to schedule maintenance or flag deviations. Moreover, semantic search becomes possible within complex environments, allowing queries like “show all storage tanks larger than 2 m in diameter” or “highlight all ladders within 2 m of an edge.” Together, these affordances demonstrate that semantic labelling is not only valuable for interpretation but also foundational for automation, safety, and interactive applications in immersive environments. The absence of semantic information necessitates additional labelling procedures or task-specific training of learning models, which can be costly, time-consuming, and difficult to generalise to different domains [4].

Conventional approaches to 3D labelling often rely on incremental mapping or multi-view fusion of RGB-D videos, where camera trajectories and frame-by-frame captures help register 2D semantic information into the 3D domain [5]. While these methods have achieved satisfactory segmentation outcomes in structured settings - such as indoor navigation and robotics - they tend to be domain-specific, requiring large annotated datasets or precise knowledge of acquisition protocols [6]. Moreover, many techniques focus on continuous mapping in real-time, leaving post-hoc segmentation of static point clouds underexplored. Consequently, new methods that allow zero-shot annotation of 3D data without expensive manual labelling or strong assumptions about the capture process are of increasing interest.

Mature techniques have been proposed for 2D object segmentation, and a number of research works have attempted to expand the 2D-based approaches in 3D domain [7]. Recent advances have also extended foundation models like the Segment Anything Model (SAM) to 3D domains through various prompt-based approaches, including multi-directional video interpretation [8], transformer-based point prompting [9], and multi-view frame projection methods [10]. While these developments demonstrate significant progress in automated 3D segmentation, they remain constrained by indoor scene limitations, computational complexity with large point clouds, and occlusion handling and degraded output challenges in real-world environments.

To address these challenges, this work presents a Language-guided zero-shot 3D SEgmentation and Reprojection tool (LASER)**,** in a pipeline that integrates text-prompted, zero-shot segmentation with multi-view 2D rendering and reprojection techniques. First, the 3D point cloud is filtered and the ground/non-ground points are separated. Then, the point cloud is sampled from multiple angles using a virtual camera to generate both orthophotos and perspective views. The angles are strategically chosen to maximise coverage of potential occlusions or intricate surface details. Next, a text prompt (for instance, “window”, “silo” or “excavator”) is converted into initial bounding boxes using a language-driven Grounding DINO detection model, which are then refined via a state-of-the-art segmentation network implemented with Segment Anything Model (SAM) [11]. The resulting 2D semantic masks are subsequently re-projected into the 3D space and merged through a confidence-weighted fusion process that enforces consistent labelling across overlapping views.

Unlike previous methods that depend on camera rigs, continuous trajectories, or complex fusion techniques, LASER allows practitioners to define new viewpoints on-the-fly, apply zero-shot segmentation, and iteratively refine the scene. In doing so, we build upon and extend the capacity of foundation models like SAM, and Grounding-DINO methods and zero-shot strategies bringing their benefits to bear on practical VR development scenarios. Also, in contrast to prior methods that rely on costly manual annotation, large-scale domain-specific datasets, or camera trajectory data, LASER operates solely on the 3D scan, thereby offering a flexible, post-hoc solution applicable to existing point clouds. Notably, the pipeline supports straightforward iteration: minor adjustments to viewpoints or text prompts can lead to refined segmentation without retraining or relabelling. By automatically associating geometric coordinates with semantic meaning, LASER produces a semantically labeled mesh output that can be exported to standard 3D formats and seamlessly integrated into interactive or dynamic rendering workflows.

This work makes the following contributions**:**

i. A zero-shot 3D segmentation and object extraction framework that leverages text prompts to identify and label objects within static 3D scans, minimising reliance on domain-specific training data.
ii. A multi-view sampling strategy designed to detect and mitigate occlusion problems in complex scenes that might be missed by single-angle approaches.
iii. A confidence-weighted fusion mechanism for merging 2D semantic masks into a coherent 3D point cloud, reducing inconsistencies across overlapping segments.
iv. Demonstrations of a semantically labelled mesh output that integrate directly with immersive environments, supporting rapid prototyping and more intuitive management of annotated point clouds.

The rest of this paper is organised as follows: Section 2 gives a broader view of the existing work in relation to the proposed tool. Section 3 discusses the theoretical framework of the LASER tool, including multi-view orthophoto creation, language-based detection, mask reprojection, and fusion. In Section 4, we present experimental results on real-world datasets, along with ablation studies examining sensitivity to viewpoint selection and text-prompt variations. Section 4 also discusses the broader implications of zero-shot 3D segmentation in industrial and VR contexts, while Section 5 offers direction for future work.

2. Background

The field of 3D semantic segmentation has evolved from early reliance on carefully crafted sensor arrangements and domain-specific training to a stage where zero-shot approaches, large-scale pre-trained models, and flexible pipelines are becoming increasingly prevalent. Classical methods, such as PointNet [12,13] pioneered the direct processing of raw point sets and paved the way for a wealth of deep learning architectures that operate directly on irregular 3D data. These early networks, however, typically required substantial training on dedicated datasets, limiting their applicability to novel domains or use-cases where annotated data are scarce.

Subsequent work on semantic mapping from RGB-D streams or integrated sensor rigs often assume incremental data capture and task-specific models. Techniques like SemanticFusion and PanopticFusion relied on continuous camera trajectories and dedicated training regimes, while approaches such as PointPainting incorporated camera imagery to enrich LiDAR data with semantic cues. Although effective in robotics and navigation scenarios, these solutions demanded stable sensor fusion and domain-specific annotation efforts. Recent research has addressed these limitations. For instance, Michele et al. [14] explored zero-shot paradigms to label 3D data without labelled examples from the target domain, while studies like Wang et al. [15] and Zhu et al. [16]

integrated knowledge from large-scale vision-language models (e.g., CLIP) into the 3D domain, enabling open-world semantic understanding without conventional training steps.

This trend is further exemplified by research adapting foundation models and prompt-able architectures for 3D scenes. PointCLIP V2 [16] repurposes CLIP's versatile representations for point cloud segmentation, illustrating how large pre-trained 2D models can be extended to complex 3D environments. Similarly, Wang et al. [15] leveraged CLIP's high-level semantic understanding for zero-shot segmentation of point clouds, while Zhao et al. [17] introduced a divide-and-conquer strategy for 3D instance segmentation that does not strictly rely on per-instance human annotations. Michele et al. [14]. took further steps toward unlocking generalisable segmentation without narrowly defined datasets, a broader move away from heavily supervised 3D pipelines.

Moreover, the emergence of the Segment Anything Model (SAM) [11] transformed 2D segmentation by enabling zero-shot capabilities across diverse image domains. Building upon SAM's generality, recent works have attempted expanding the 2D-based SAM and SAM2 [18] into 3D domain. One such effort is the segment anything 3D implementation which gives a state-of-the-art result but lacks language implementation [7]. A number of research works have also integrated prompt-able modules on SAM for 3D segmentation. Such efforts include SAM2Point [8], PointSAM [9], and SAMPro3D [10].

SAM2Point framework adapts SAM for zero-shot and prompt-able 3D segmentation by interpreting 3D data as a series of multi-directional videos, leveraging SAM2 for segmentation. Point-SAM introduces a transformer-based 3D segmentation model designed for interactive guidance through point prompts. The framework extends the SAM to the 3D domain and is trained using a data engine that generates 3D segmented objects from 2D SAM outputs. SAMPro3D segments 3D indoor scenes applying the pretrained SAM to 2D frames derived from multiple posed 2D images of 3D scenes. The approach involves locating 3D points in scenes as natural 3D prompts to align their projected frames, ensuring frame-consistency in both pixels prompts and their SAM-predicted masks. These attempts give a significant advancement over the conventional 2D SAM but does not work out of the box using the 3D scans alone. There have been other attempts to use multi-angle approach by synthesizing different 2D views and using SAM as a universal segmentation tool. While these attempts successfully showcased the utility of SAM in providing semantic cues without retraining, they frequently focused on proof-of-concept implementations rather than flexible, user-driven workflows. Also, the reprojection is ad-hoc, and does not solve the occlusion issue in large scenes.

Our work, LASER, aligns with these emerging lines of inquiry. Like the zero-shot approaches discussed above, we aim to avoid domain-specific training. By leveraging SAM's general-purpose segmentation abilities, we create a pipeline that only requires a static point cloud and user-defined viewpoints to produce semantic annotations in open-world 3D scans.

3. Method

Building upon the foundation of the prior works discussed in Section 2, our method integrates foundation models with user-defined viewpoint selection to create an efficient, flexible pipeline for semantic 3D segmentation in an optimised manner. First, unlike SAM2Point, which interprets 3D data as multi-directional videos, LASER operates directly on 3D point clouds, reducing computational overhead and preserving spatial information inherent in the 3D data. Secondly, the current work leverages foundation models like Grounding DINO, enabling language-guided segmentation. This integration allows for more intuitive and flexible interactions, as users can employ natural language prompts to guide the segmentation process. Third, the LASER pipeline supports modularity and iterative refinement, allowing users to adjust viewpoints or segmentation parameters post-hoc. This flexibility facilitates the creation of semantically labelled assets that can be easily exported in standard formats. Moreover, unlike methods such as SAMPro3D, which rely on multiple posed 2D frames and camera trajectories as inputs, the current work does not require any camera motion information. LASER operates solely on 3D data without the need for accompanying 2D images or RGB-D information. This independence simplifies the workflow and broadens the applicability to static 3D scans.

This section discusses the conceptual mathematical formulations as key components of LASER. The pipeline is designed as a sequential process, with each phase addressing specific challenges involved in transforming an unsegmented 3D point cloud into a refined, segmented, and virtual reality-ready asset.

3.1. Pre-processing and ground filtering

The LASER pipeline begins by accepting a static 3D point cloud (in . ply) as its input. The initial step involves pre-processing and ground filtering, where the point cloud is first denoised and outliers are removed. The input point cloud is represented as a set of 3D points:

$$\mathscr{P} = \{p_i = (x_i, y_i, z_i) | i = 1, \ldots, N\} \tag{1}$$

This is usually with colour information $c_i = (r_i, g_i, b_i)$. Then the point cloud is divided into ground and non-ground subsets. The ground/non-ground separation step is done using the Cloth Simulation Filter (CSF). This classifies each point cloud into ground $\mathscr{P}_{ground}$ and non-ground $\mathscr{P}_{non\_ground}$ subsets. By focusing on each section separately, we reduce computational load and improve segmentation accuracy, as non-ground objects (buildings, vegetation, vehicles) are segmented more distinctly. Each point can be represented with the relation:

$$\mathscr{P}_{non\_ground} = \{p_i \in \mathscr{P} | \text{isGround}(p_i) = \text{False}\} \tag{2}$$

3.2. Multi-view orthophoto generation

Following this, a virtual camera is used to generate orthophotos from multiple views of the point cloud, for both ground and non-ground points. Each ground and non-ground points are then rotated and translated based on these camera parameters to create a unique perspective of the scene. To achieve this, we select a set of viewing angles $\theta j\{\theta_j\}$ and a camera position P_{cam} from which to generate orthophotos and perspective images. For each angle, we define a rotation R_j and optionally a translation t_j, which together form an extrinsic camera matrix:

$$R_j : R^3 \rightarrow R^3, \quad t_j \in R^3 \tag{3}$$

For each point $p_i \in \mathscr{P}_{non\_ground}$, its rotated coordinate is defined by Eq. (4).

$$p'_{i,j} = R_j(p_i - P_{cam}) \in R^3 \tag{4}$$

Similarly, the orthographic projection to a 2D image plane is defined by Eq. (5).

$$(x_{2D}, y_{2D}) = \Pi\left(p'_{i,j}\right) = \left(\frac{x'_{i,j}}{\Delta}, \frac{y'_{i,j}}{\Delta}\right) \tag{5}$$

Where Δ is the resolution factor. This maps each 3D point to a pixel in a 2D image $\mathscr{I}_j$. The result is a set of images $\{\mathscr{I}_j\}$, each representing a unique viewpoint of the non-ground scene. This is implemented with a special function that allows user-defined viewing angles. The default angles [(−60, 0), (60, 0), (0, −60), (0, 60)] were found to generate a complete view for most real-world tested with LASER. However, provisions have been made for an optional adaptive view selection via a view selection algorithm formalised as follows:

From a candidate set of azimuth-elevation pairs $\mathscr{A} = (\phi,\theta)$, we select K views by maximising expected coverage and informativeness:

$$\underbrace{S(\phi,\theta) = \sum_{p \in \mathscr{P}} r_{\phi,\theta}(p)\, \rho(p)}_{\text{visible coverage}} + \underbrace{\lambda H\left(\hat{C}_{\phi,\theta}\right)}_{\text{label entropy}}, \tag{6}$$

where $r_{\phi,\theta}(p) \in \{0,1\}$ is the visibility of point p (from the z-buffer), $\rho(p)$ is a density weight (e.g., local kNN density), and $H(\widehat{C})$ is the Shannon entropy of provisional class predictions from a low-resolution render. We use greedy sub-modular maximisation with a diversity penalty to avoid redundant views.

3.3. Bounding box detection using text prompts

Given an input text prompt, Grounding DINO predicts a set of objects bounding boxes B within the generated multi-view orthophotos $\mathscr{I}_{j}$. The model assigns a confidence score s_b to each detected bounding box:

$$B_{j} = \{(b_k, s_k)|k = 1,\ \ldots.,K\},\ b_k = (x_{min}, y_{min}, x_{max},\ y_{max}) \tag{7}$$

Where b_k is the bounding box for object k, s_k is the confidence score (τ_dino) for the detection, below which the scene is discarded. The $(x_{min}$, $y_{min})$ and $(x_{max},\ y_{max})$ define the top-left and bottom-right corners of the box. Each bounding box localises an object in 2D space, serving as an initial region of interest for segmentation. Once Grounding DINO detects object regions, these bounding boxes B_{j} are passed to SAM, which further refines them into precise segmentation masks defined by Eq. (8) below:

$$\mathscr{M}_{j,k} \ = \ SAM(I_{j}, b_k) \tag{8}$$

Where $\mathscr{M}_{j,k}$ is the segmentation mask for object k, and SAM operates within the bounding box b_{k} ensuring accurate per-object segmentation.

3.4. 2D segmentation with segment anything model (SAM)

Once Grounding DINO detects object regions, these bounding boxes serve as input for SAM, which operates within each detected region to refine object masks with pixel-level precision. SAM uses the generated orthophotos $\mathscr{I}_{j}$ for 2D segmentation, and produces segmentation masks, $\mathscr{M}_{j,k}$ for various object-like regions within each view indexed with k. Each mask is assigned a confidence score (τ_sam ϵ $c_{j,k}$) derived from SAM's predicted Intersection over Union (IoU) and stability metrics. The confidence $c_{j,k}$ for each mask segment is defined by:

$$c_{j,k} = \frac{\text{predicted\_IoU}_{j,k} + \text{stability\_score}_{j,k}}{2} \tag{9}$$

In LASER, language priors enter via GroundingDINO's text to image alignment and SAM's prompt-able mask refinement, applied to multi-view orthophotos of the 3D scan. Intuitively, the language prompt provides a class-conditional prior over 2D regions, and multi-view reprojection transports this prior back into 3D To answer the question of point-wise labeling formalisation, let p be a 3D point, and let $\mathscr{V}(p)$ denote the set of rendering views in which p is visible (determined by z-buffer visibility in orthographic renders). Each view $v \in \mathscr{V}(p)$ yields a candidate class c with confidence score $\kappa_{v,c}(p)$, obtained by combining the detection confidence from GroundingDINO and the mask quality score from SAM:

$$\kappa_{v,c}(p) \ = \ \sigma\left(\alpha\, s_{v,c}^{\text{dino}} \ + \ \beta\, s_{v,c}^{\text{sam}}\right)\cdot g\left(\theta_{v,p}, d_{v,p}\right), \tag{10}$$

where $s_{v,c}^{\text{dino}}$ is the detection score for class c from GroundingDINO in view v, and $s_{v,c}^{\text{sam}}$ is SAM's predicted IoU or stability score. The function

$$g(\theta, d) = \cos^{\gamma}(\theta)\cdot\exp(-\lambda d) \tag{11}$$

discounts unreliable evidence from grazing angles θ and large depth values d. The squashing function $\sigma(\cdot)$ ensures bounded confidence values.

The posterior probability of assigning class c to point p, given text prompt T, is then obtained by aggregating evidence across all views in which p is visible:

$$P(c|p,T) \propto \sum_{v\in\mathscr{V}(p)} w_v\, \kappa_{v,c}(p), \tag{12}$$

with per-view normalization weights

$$w_v \ = \ \frac{\sum_{p\in\Omega_v} \kappa_{v,\hat{c}}(p)}{\sum_{v'} \sum_{p\in\Omega_{v'}} \kappa_{v',\hat{c}}(p)}, \tag{13}$$

Where $p = a$ 3D point, $\mathscr{V}(p) =$ set of views where p is visible, $s_{v,c}^{\text{dino}} =$ GroundingDINO detection score for class c in view v, $s_{v,c}^{\text{sam}} =$ SAM IoU/ stability score for class c in view v, $\sigma(\cdot) =$ squashing function (e.g., sigmoid), $\theta_{v,p} =$ incidence angle of p in view v, $d_{v,p} =$ depth of p in view v, Ω_v is the set of pixels/points visible in view v, and $\widehat{c}$ denotes the current maximum a posteriori (MAP) class label. This formulation highlights how LASER uses language as a semantic prior, 2D vision models as proposal generators, and multi-view geometry as a consistency constraint to produce reliable 3D point-wise labeling in a zero-shot setting.

3.5. 2D to 3D reprojection and merging with confidence-weighted fusion algorithm

The segmented 2D data is then re-projected back into the 3D space. To handle varying completeness across viewpoints, each re-projected point carries both its spatial position and segmentation confidence from its 2D view, which helps mitigate occlusions and overlapping segments when merging multi-angle data.

First, for every pixel (x_{2D}, y_{2D}) that belongs to a mask $\mathscr{M}_{j,k}$, we find the corresponding 3D point $p'_{i,j}$ from the inverse projection:

$$p'_{i,j} = \Pi^{-1}(x_{2D}, y_{2D}) \tag{14}$$

and then map it back to the original coordinate system:

$$p_i = R_j^{-1p'_{i,j}} + P_{cam} \tag{15}$$

All points belonging to a given 2D mask $\mathscr{M}_{j,k}$ are assigned the segment's semantic attributes and confidence. Thus, from each viewpoint, we obtain a set of 3D points associated with a particular segmentation mask:

$$\mathscr{P}_{j,k}^{seg} = \left\{ \left(p_i, c_i, c_{j,k}\right) \middle| (x_{2D}, y_{2D}) \in \mathscr{M}_{j,k} \right\} \tag{16}$$

It was observed that multi-view segmentation faces challenges from conflicting assignments due to occlusions and noise. Rather than naive averaging, the confidence-weighted fusion is used in this work. To consolidate multi-view segmented data, the pipeline uses confidence-weighted fusion, clustering nearby points and computing weighted averages of positions and colours. This resolves duplicate points representing the same 3D region with different segment assignments. This strategy also prioritises high-quality segmentations by giving greater weight to points with higher confidence, favouring reliable segmentation while reducing the impact of uncertain assignments.

A key innovation in this work is using the algorithm to map each masked pixel to its corresponding 3D point via reprojection, ensuring accurate 3D representation of 2D segmented regions. For the fusion algorithm, we define a neighbourhood radius r and construct a set of clusters $\mathscr{C}_m$ for some reference point p_{ref} in that cluster where each cluster groups points that lie within radius r of each other:

$$\mathscr{C}_m = \left\{ (p_i, c_i, c_i) \middle| \left| p_i - p_{ref} \right| \le r \right\} \tag{17}$$

To fuse a cluster $\mathscr{C}_m$, we compute a confidence-weighted average of positions and colours:

$$\overline{p_m} = \frac{\sum_{(p_i,c_i)\in\mathscr{C}_m} c_i p_i}{\sum_{(p_i,c_i)\in\mathscr{C}_m} c_i}, \overline{c_m} = \frac{\sum_{(c_i,c_i)\in\mathscr{C}_m} c_i c_i}{\sum_{(c_i,c_i)\in\mathscr{C}_m} c_i} \tag{18}$$

Here c_i is the confidence for the point derived from the mask it originated from, and c_i is its colour. By weighting each point's contribution to the fused point by its confidence, we emphasise higher-quality segmentations and suppress uncertain or noisy assignments. We select the final confidence for the fused point as:

$$c_{fused,m} = \max_{(p_i,c_i)\in \mathscr{C}_m} c_i, \tag{19}$$

The result of this fusion is a cleaner, redundancy-free set of segmented points:

$$\mathscr{P}^{fused} = \left\{\overline{p_m}, \overline{c_m}, c_{fused,m} \middle| m = 1, \dots, M\right\} \tag{20}$$

After back-projection, labeled 3D points are clustered per class using DBSCAN with radius ϵ scaled by local point spacing. Spatial consistency is enforced through (i) clustering in Euclidean space, (ii) majority/MAP labeling within clusters, and (iii) non-maximum suppression across overlapping clusters. The algorithmic implementation of step is summarised in Table 1 below.

3.6. Mesh reconstruction and refinement

Once the confidence-weighted 3D segmentation is complete, the next step is to convert the fused point cloud into a structured surface mesh in order to facilitate VR integration and rendering. This conversion is done with Poisson Surface Reconstruction algorithm to obtain a mesh $\mathscr{M}$:

$$\mathscr{M} = \text{PoissonReconstruction}\left(\mathscr{P}^{fused}\right) \tag{21}$$

The confidence-weighted fusion step refines the segmentation by mitigating redundant or uncertain points, leading to a more accurate and structured 3D representation. This refinement ensures that Poisson Surface Reconstruction operates on a clean, high-confidence dataset, improving the quality of the resulting mesh. By maintaining segmentation confidence throughout the workflow, LASER reduces artefacts in the final semantically labeled model, producing a more stable and visually coherent 3D asset. This mesh is then refined through density-based cropping to remove low-density vertices and noise, enhancing the quality and structural integrity of the model. The end result is a reconstructed mesh representation that retains detailed segment masks through vertex colours and attributes, making it suitable for immersive virtual environments. Taken together, LASER leverages language as a semantic prior, vision as a proposal generator, and geometry as a consistency check, enabling open-vocabulary segmentation without task-specific 3D supervision.

Table 1
Optimised 3D reprojection and confidence-based fusion for LASER segmentation.

Steps	Formulation	Description
1	Extract 2D Segmentation Masks: $M = \{m_i\}_{i=1}^{N}$	Obtain a set of segmentation masks M from multi-view images.
2	Back-Project to 3D: $P_{2D} = K^{-1}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} d$; $P_{3D} = R^{-1}(P_{2D} - T)$	Convert 2D points (x, y) into 3D using intrinsic matrix K and depth d. Transform into world coordinates using camera pose (R, T).
3	Compute Confidence for Each View: $C_i = f(\text{mask quality}, \text{projection consistency})$	Assign confidence scores C_i based on segmentation quality and projection accuracy.
4	Perform Confidence-Weighted Multi-View Fusion: $w_i = \frac{C_i}{\sum C_i}$; $P_{\text{merged}} = \sum_{i=1}^{N} w_i P_{3D}^{(i)}$	Normalize confidence weights for merging. Compute weighted 3D point fusion across views.
5	Prune Redundant Points: $P_{\text{final}} = \text{prune}\left(P_{\text{merged}}, \epsilon\right)$	Remove duplicate points within a spatial threshold ϵ.
6	Output: P_{final}	Refined 3D segmented point cloud P_{final}.

4. Results

To evaluate LASER, we used chunks from a high-density LiDAR point cloud of an active construction site (Fig. 1). The full dataset contains approximately 176 million points in .xyz format and captures structures at different stages of completion, along with site machinery such as excavators, concrete mixers, and a cement silo. It also includes debris, pathways, trailing cables, and other safety-relevant features, making it well suited for VR-based safety training scenarios. In addition, accompanying .obj and .mtl files with texture maps provide comprehensive 3D visualization of the site. For computational efficiency, the point cloud was partitioned into smaller chunks prior to processing.

The goal is to evaluate LASER on its two key functions (a) segment point clouds, and (b) semantically label and extract objects from point cloud with text prompt. The output object (mesh) is then processed, animated and integrated in a VR environment to create dynamic scenarios for safety training using game engines. The evaluation is performed on Dell precision 3660, 64 G RAM and 13th Gen Intel UHD graphics 770 workstation (RTX 4090). To ensure a structured and scalable evaluation, we adopted a phased approach. Initially, a region of interest (ROI) was selected, which is a smaller yet representative subset of the site scan for preliminary testing and segmentation benchmarking. Once the segmentation pipeline was fine-tuned, the best-performing parameters were deployed across a bigger chunk of the dataset, enabling larger-scale point cloud parsing while retaining segmentation accuracy. This strategy ensured that the tool is properly evaluated on real-world construction environments.

4.1. LASER for 3D segmentation workflow

The first evaluation uses LASER for zero-shot 3D segmentation. The . xyz file is converted into .ply file format. Input point clouds (.ply) are processed through the pipeline and the tool outputs semantically segmented 3D objects from point cloud data in .ply, .gltf and .glb formats, facilitating downstream applications. The segmentation workflow is represented by Fig. 2. The workflow in Fig. 2 also shows the segmentation result on the ROI which is a chunk from the scan in Fig. 1. The chunk shows a house under construction, with adjoining garage, trailing cable, and an excavator. The segmentation results indicated with colour masks demonstrate clear delineation of structural components, as shown in Fig. 2, where distinct colours represent semantically meaningful segments such as walls, roofs, doors, windows, garage, and excavator. The ground and non-ground objects are also clearly delineated.

4.1.2. Parameter selection and optimisation strategy

The parameter selection follows a principled approach combining default values established through empirical evaluation with adaptive strategies for varying point cloud densities. For dynamic angle optimisation, the system employs a greedy strategy that maximizes a coverage function S(φ,θ) subject to geometric diversity constraints, ensuring orthographic projections capture complementary perspectives while avoiding redundant viewpoints. The number of views K is set between 4–8 based on scene complexity, with detection thresholds τ_dino ∈ [0.35,0.5] and segmentation thresholds τ_sam ∈ [0.3,0.9] on predicted IoU scores. The large τ_sam threshold is to accommodate for edge cases. The minimum mask area *a_min* is scaled proportionally to image resolution to maintain consistent filtering across different projection scales.

For clustering and fusion, the DBSCAN radius ε is set to 1.5× the median nearest-neighbour spacing to adapt to local point density variations, while maintaining a minimum cluster size of $m = 30$ points to suppress noise. The system incorporates density-aware adaptations for challenging scenarios where point density falls below ~50 pts/m^2. In such cases, the method increases ε to accommodate sparser clustering, raises τ_dino to reduce false positive detections, and prioritizes

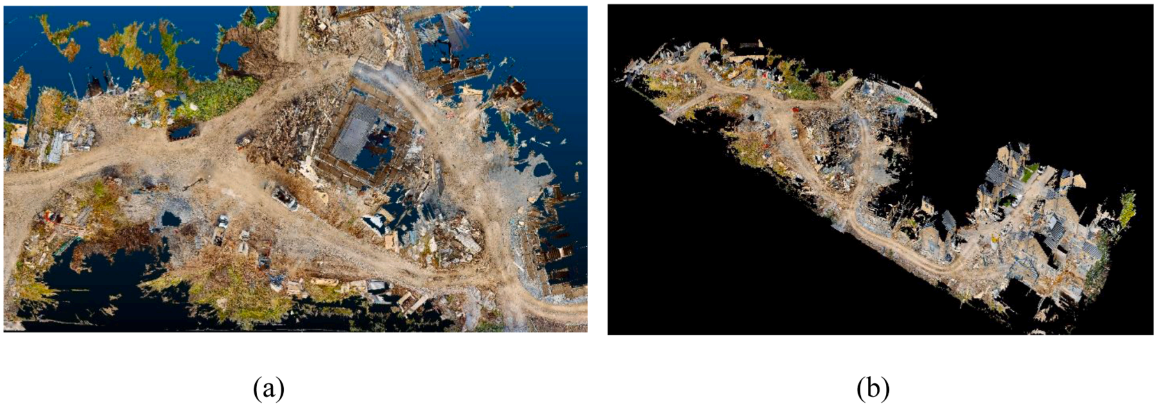

Fig. 1. LiDAR Scan of a construction site with (a) aerial view and (b) Dollhouse view.

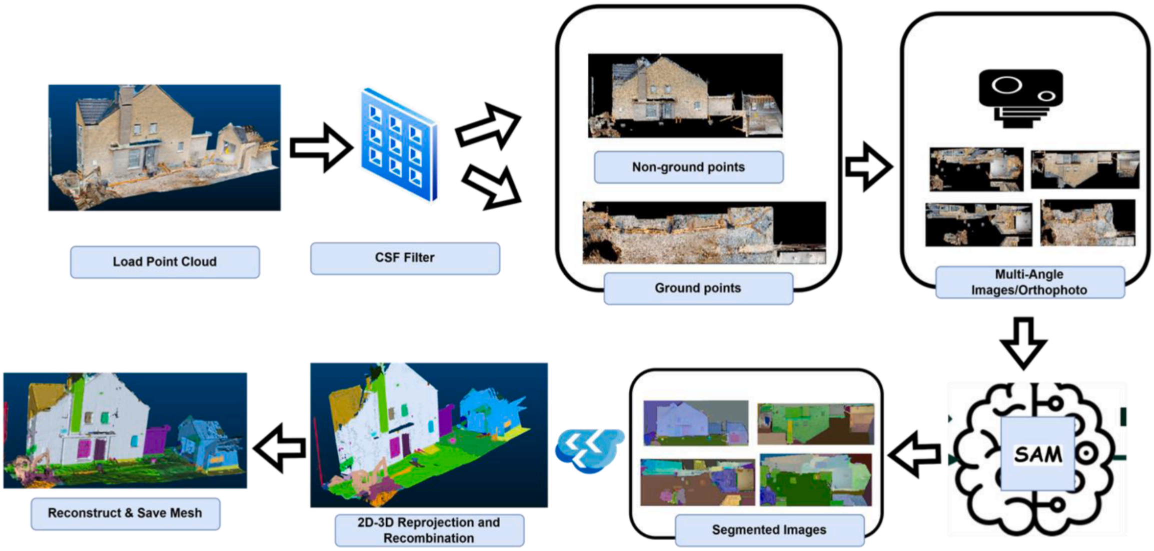

Fig. 2. LASER workflow for zero-shot 3D segmentation.

viewpoints with higher expected surface incidence angles. Additional mitigation strategies include using coarser semantic class prompts, applying cluster-level confidence filtering, and optionally employing mesh rasterization to densify critical regions. This adaptive parameter framework ensures robust performance across diverse point cloud qualities while maintaining computational efficiency. The Table 2 below shows the input/output dataflow interface for each module of the LASER workflow.

4.1.3. *Comparison of LASER output and manual segmentation*

To properly evaluate the performance of LASER on segmentation tasks, we further compare the manual annotation done by human annotators on two representative ROI from the scans with the LASER segmentation output, as shown in Fig. 3 below. Analysis reveals the tool successfully identify major structural elements, as evidenced by the distinct color-coded regions (middle) broadly corresponding to the manual segmentation reference (right). Principal components such as walls (white), ground plane (green), and architectural features were consistently identified in both segmentation methods. However, notable differences emerge in the granularity and boundary precision, segmentation accuracy and efficiency.

Manual annotation shows finer segmentation of structural details, particularly in areas of geometric complexity such as window frames (blue regions) and architectural transitions. LASER's segmentation exhibits more generalised boundaries between segments, with some oversimplification of complex geometric transitions. This is particularly evident in the treatment of the door area and the roof-wall transitions, where manual annotation shows more detailed partitioning. Moreover, LASER's performance declined for fine-grained objects, such as pipes, drainages and construction debris. Segmentation noise was observed in regions with sparse point density or overlapping objects, indicating sensitivity to scene complexity. These differences highlight the trade-off between automated processing efficiency and the fine-grained accuracy achievable through manual annotation, suggesting potential areas for algorithmic refinement in future iterations of LASER.

In terms of boundary precision, LASER produced smoother boundaries compared to the manually refined edges, occasionally over-segmenting irregular structures (e.g., silo in Fig. 3) into smaller clusters. Conversely, under-segmentation occurred in regions with low contrast between adjacent objects (e.g., cars parked near walls). Manual annotations preserved finer details, particularly in complex elements (e.g. the adjoining garage) where LASER's geometric heuristics struggled to resolve intricate features.

In terms of computational efficiency, LASER processed the entire point cloud with 11,004,983 points in approximately 4 min, significantly faster than the 120 min required for manual annotation. This efficiency advantage positions LASER as a practical tool for large-scale applications. However, the trade-off between speed and precision is evident in its handling of ambiguous regions, where human annotators applied contextual reasoning (e.g., distinguishing between pipes,

Table 2
LASER workflow's input/output interface and dataflow.

Stage	Input	Output
System Input	Point cloud (.ply or .xyz), natural language prompt(s), and system parameters: number of views K, thresholds τ_dino, τ_sam, and DBSCAN radius ε	-
Preprocessing	Raw point cloud	Cleaned point cloud with normals (outlier removal and ground/non-ground separation)
Rendering	Cleaned point cloud with normal	Orthographic projection images and index maps linking pixels to 3D points from selected views
Detection	Projection images, natural language prompts	Bounding boxes with scores from GroundingDINO (NMS applied, boxes below τ_dino discarded)
Segmentation	Bounding boxes, projection images	Refined masks from SAM (masks below predicted-IoU threshold τ_sam or area a_min removed)
Back-projection	Masks, index maps	3D points mapped from masks with per-point confidence scores
Fusion	3D points with confidence scores	Labeled 3D point cloud (DBSCAN clustering and confidence-weighted merging)
Optional Meshing	Labeled 3D point cloud	Class-wise Poisson reconstructed mesh exported as .ply/.glb with semantic labels

construction debris and floor) that LASER could not replicate.

Common error modes in LASER output includes segmentation accuracy dropped by ~20 % in areas with point densities below 50 pts/m^2. Moreover, objects with similar geometric profiles but differing textures (e.g., window hedges vs. low walls) were occasionally conflated, and objects with complex geometric interactions (e.g., pipes and ground) were occasionally merged. While LASER reduces annotation labour by ~96 % which reduce manual segmentation time, its reliance on geometric features alone limits performance in cluttered or texture-dependent scenarios. It is pertinent to note that these analyses were done with default parameters across all LASER inputs. Post-processing refinement steps or user-guided corrections and angle selection may further bridge the accuracy gap with manual methods. Fine-tuning the backbones on domain-specific datasets could also mitigate these gaps.

4.2. LASER object extraction workflow

The LASER tool's object extraction pipeline takes similar input and the outputs are in similar format as the segmentation workflow. The task here is to use the tool to process and semantically segment point clouds into meshes to isolate individual 3D objects. Building on the segmentation output, the object extraction phase uses the backbone models to identify connected components within each semantic class (e.g., drainage, excavator). These components are then exported as discrete 3D assets in *.ply* and *.glb* formats, preserving semantic labels and spatial coordinates.

Input to the object extraction module consists mostly of default parameters from the segmentation workflow, along with the text prompt, as illustrated in Fig. 4. Overlapping clusters are resolved via voxel-based occupancy checks, prioritizing larger connected regions. Extracted objects are output as both labelled point clouds and meshes, enabling direct integration with 3D modelling software and game engines (e.g., Blender/Unreal Engine, and Unity). The .glb format ensures texture and semantic metadata retention, critical for applications in VR, augmented reality (AR), and digital twins.

The final output consists of extracted 3D objects, which are reconstructed and saved as individual assets for downstream applications. Fig. 5 below shows the input point cloud and text prompts to extract windows and doors from an input point cloud of the building ROI. As shown in Fig. 5, it is seen that, despite the occlusions and holes in the original point, the tool is able to extract all instances of windows and doors in the point cloud. Similar extractions are observed in other ROI tested using the tool to extract object of interests. This result shows the capability of the tool for zero-shot object search in large, occluded scene common in real-world scans. It was also observed that there were no false-positives in the tool output when evaluated using false prompt samples. For instance, prompting the tool for 'basket' in the region of interest shown in Fig. 5 gives no false positive.

This analysis shows the LASER tools capability in both the segmentation result and language-guided object extraction. The results also indicate potential applications in digital twin creation, architectural visualisation, and heritage documentation, where automated segmentation can substantially reduce development time compared to manual methods.

4.3. LASER's quantitative performance evaluation

To further analyse the performance of the LASER tool proposed in this study, the quantitative performance of the tool applied to a 3D point cloud dataset is done, focusing on detecting window structures. By comparing model predictions with manually labelled ground truth data, key evaluation metrics such as Intersection over Union (IoU), accuracy,

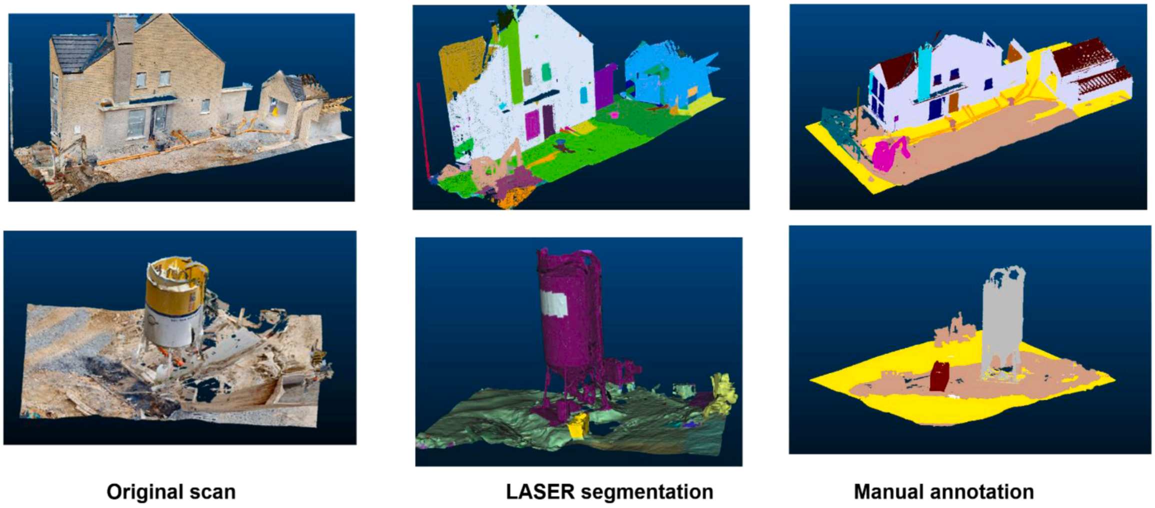

Fig. 3. LASER output vs manual annotation on two ROI: a house (up) and silo area (down).

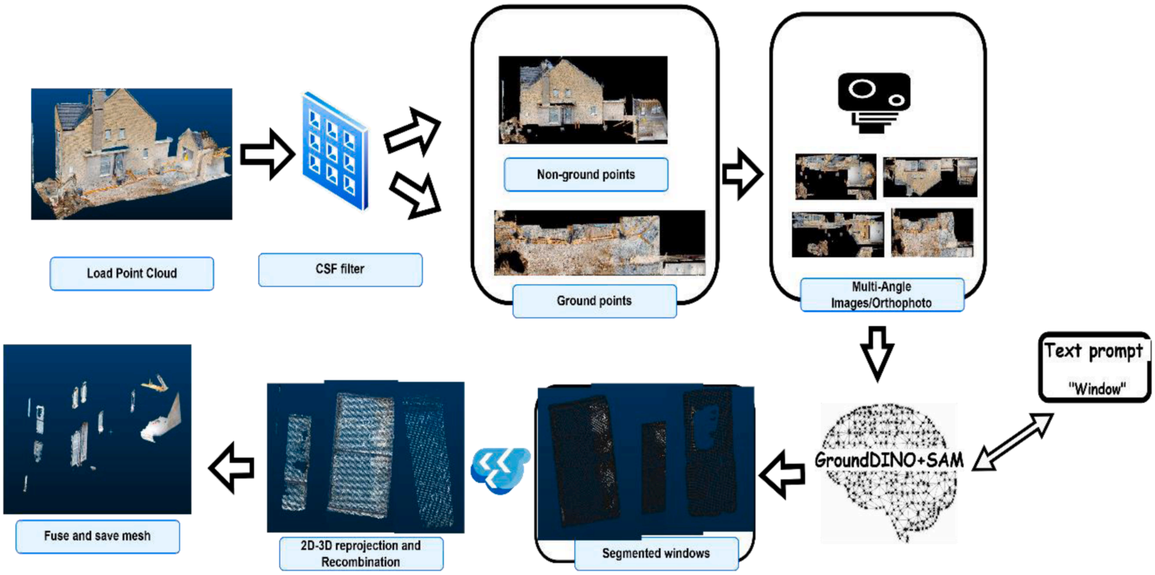

Fig. 4. LASER for 3D object extraction (with text prompt).

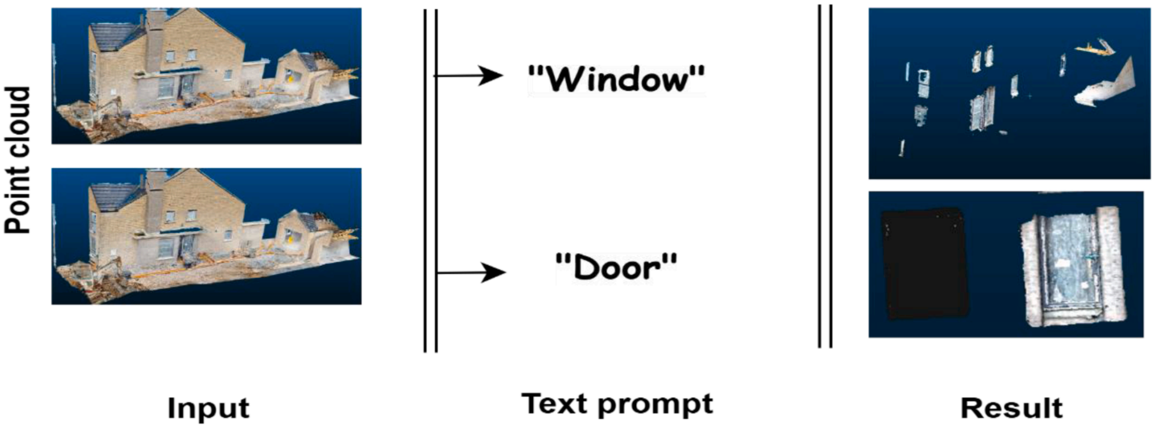

Fig. 5. LASER for object extraction with sample text prompts.

precision, recall, and F1-score were computed as shown in Table 3 and Fig. 6 below.

The evaluation involved pre-processing two-point cloud datasets: one containing manually annotated ground truth labels and another with model-generated predictions. Using NumPy and Open3D, labels corresponding to windows were extracted, and a nearest-neighbour search (KDTree) was used to establish correspondences between predicted and ground truth points. The Iterative Closest Point (ICP) algorithm was employed to align the two datasets, ensuring minimal misalignment. Quantitative assessment against manual extraction benchmarks revealed LASER achieved accuracy of 97.15 % and a recall of 77.79 %, indicating the model's strong ability to detect window points. However, the precision (56.61 %) and IoU (48.73 %) suggest the presence of false positives (noise), affecting segmentation quality. A high specificity (97.85 %) confirms the model's effectiveness in identifying non-window regions, with a low false positive rate of 2.15 %.

Table 3
LASER's performance evaluation against manually annotated scan.

Metric	Formula	Value
IoU	TP / (TP + FP + FN)	0.4873
Accuracy	(TP + TN) / (TP + FP + TN + FN)	0.9715
Precision (PPV)	TP / (TP + FP)	0.5661
Recall (Sensitivity)	TP / (TP + FN)	0.7779
F1-Score	2 * (Precision * Recall) / (Precision + Recall)	0.6553
Specificity	TN / (TN + FP)	0.9785
Balanced Accuracy	(Recall + Specificity) / 2	0.8782
Negative Predictive Value (NPV)	TN / (TN + FN)	0.9919
False Positive Rate (FPR)	FP / (FP + TN)	0.0215
Dice Coefficient	2 * TP / (2 * TP + FP + FN)	0.6553

4.4. *LASER validation on diverse 3D scenes*

To comprehensively evaluate the proposed tool and prove its open-world application, it is pertinent to explore the performance of the tool on diverse 3D scenes for both indoor and outdoor scans. Consequently, this section discusses the evaluation of LASER on two different datasets: i. 3DSES - Golden Section S165 (internal): We extracted the S165 "golden section" sample from our 3DSES collection. The sample was trimmed to expose the interior region of the point cloud for clearer inspection of object boundaries and occlusions. ii. Toronto3D (public): Large-scale outdoor mobile LiDAR captured in Toronto, Canada, released with point-wise labels for 8 classes. The data was collected using a Teledyne Optech Maverick MLS (32-line LiDAR), with RGB from a Ladybug 5 camera, and stored as *.ply* file with 10 attributes. The

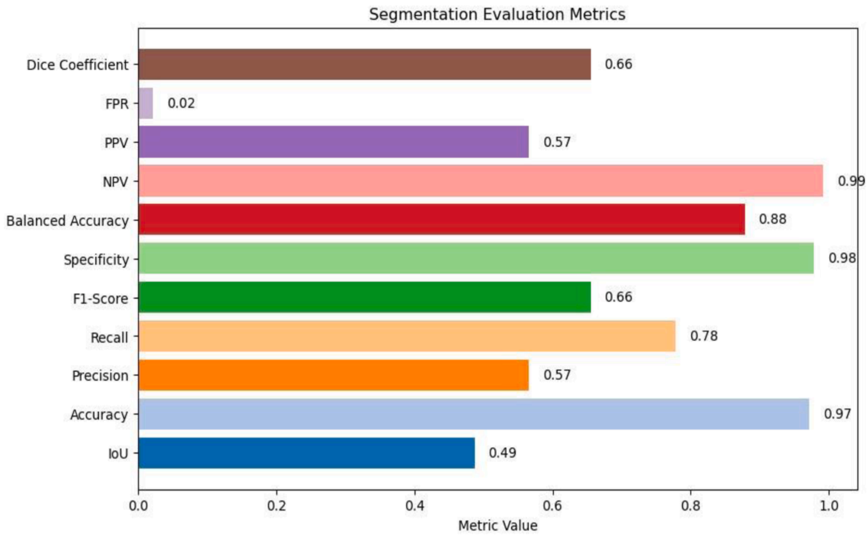

Fig. 6. Graphical illustration of the LASER 3D segmentation metric.

dataset spans ~1 km of roadway, ~78.3 M points, organised in four ~250 m sections, with average ground density of ~1000 pts/m^2 and coordinates in NAD83/UTM 17 N.[1]

4.4.1. *Validation protocol and data pre-processing*

Validation was performed on the two benchmark datasets described in Section 4.4. Each dataset provides point clouds in PLY format, where each record corresponds to a single 3D point with its Cartesian coordinates (x,y,z) and a classification label ID. For every test case, both a ground-truth file and a corresponding prediction file were obtained. A custom script was used to isolate the target object class by filtering points based on the specified label ID. This produced two sets of points: positives (belonging to the target object) and negatives (all remaining points). The loaded XYZ coordinates were converted into *open3d.geometry.PointCloud* objects for analysis. For consistency with LASER's input requirements, surface normals were first estimated for all points using a KD-tree search (radius = 0.05 m, max_nn = 30). To ensure a fair comparison, predicted point clouds were rigidly aligned to the ground truth in two stages:

i. **Coarse alignment.** Both clouds were voxel-downsampled at 0.02 m resolution. On these reduced sets, normals were re-estimated (search radius = 0.04 m), and Fast Point Feature Histograms (FPFH) were computed with a 0.10 m radius. RANSAC-based feature matching (maximum correspondence distance = 0.03 m, edge-length check = 0.9, minimum 4 correspondences) produced an initial transformation.
ii. **Fine alignment.** The RANSAC result was refined using point-to-point Iterative Closest Point (ICP), with a maximum correspondence distance of 0.04 m.

The resulting 4 × 4 rigid transformation was then applied to the full-resolution prediction cloud. This two-stage coarse-to-fine registration ensures that evaluation metrics reflect semantic segmentation quality, rather than being confounded by residual translational or rotational offsets between predicted and ground-truth point clouds.

4.4.2. *Individual test case analysis*

A detailed breakdown of each test case is provided below to illustrate LASER's performance on individual object classes. i. Test Case 1: Segmentation of a "vest" in 3DSES

For this case, the input text prompt was "vest". The ground truth consists of a labeled set of points corresponding to the vest object, which are shown in green in Fig. 7(b). Visualizations of both the ground-truth and predicted segmentations are provided in Fig. 7, and the quantitative results are as shown in Fig. 8.

As shown in Figs. 7 & 8, LASER achieved strong and well-balanced performance. Precision was 0.9160 and recall was 0.9170, yielding an F1-score (Dice coefficient) of 0.9170 and an Intersection over Union (IoU) of 0.8460. The overall classification accuracy was 0.9670, while the balanced accuracy was 0.9480. Qualitatively, the vest was captured cleanly, with very few noise and minor misses. Remaining errors are likely concentrated along thin boundaries or in small fragmented regions, but overall the segmentation is robust and reliable for this object class. ii. Test Case 2: Segmentation of a 'Pile of Book' in 3DSES

When prompted with the phrase "Pile of Books", LASER successfully identified nearly all of the ground-truth book points. The model achieved a very high recall of 0.9720, indicating that the majority of true book points were correctly captured, as shown in Figs. 9& 10.

However, this came at the expense of precision: many non-book points were also included, resulting in a relatively low precision of 0.4470. The combined metrics reflect this imbalance, with an F1-score of 0.6130 and an IoU of 0.4420 (Fig. 10).

Qualitatively, the segmented output (Fig. 9) shows that the bulk structure of the book pile is preserved, but the boundaries are over-extended into surrounding ground points. As a result, this segmentation is best suited for scenarios where a high-recall first pass is desired, such as human-in-the-loop annotation support, rather than for fully automated measurement or precise object extraction. Additional filtering or refinement would be needed to improve precision before deployment in downstream tasks. iii. Test Case 3: Segmentation of 'Car' in Toronto3D

With the prompt "Car", LASER identified vehicles present in the scene, producing the segmentation shown in Fig. 11. Quantitatively, performance was mixed: recall reached 0.6840, indicating that approximately 68 % of car points were successfully detected, while precision was lower at 0.4040, meaning that only 40 % of predicted car points were correct. The combined metrics reflect this imbalance, with an F1-score (Dice) of 0.5080 and an IoU of 0.3400 (Fig. 12).

Despite these limitations, background rejection was excellent. Specificity was 1.000 (false positive rate =0.000) and the negative predictive value (NPV) was also 1.0001. This yields an overall accuracy of 1.0001 however, as with other imbalanced datasets, this value is inflated by the dominance of negative points. The balanced accuracy of

[1] https://ar5iv.labs.arxiv.org/html/2003.08284.

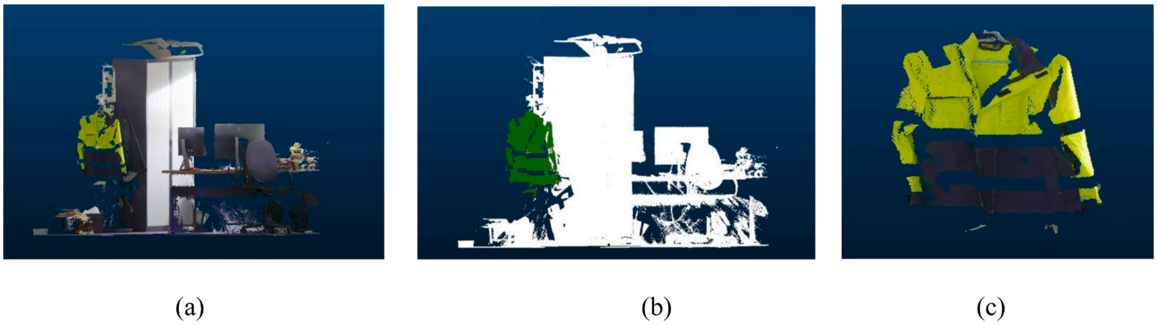

Fig. 7. (a) Input .ply file, (b) Ground Truth, and (c) LASER segmentation result.

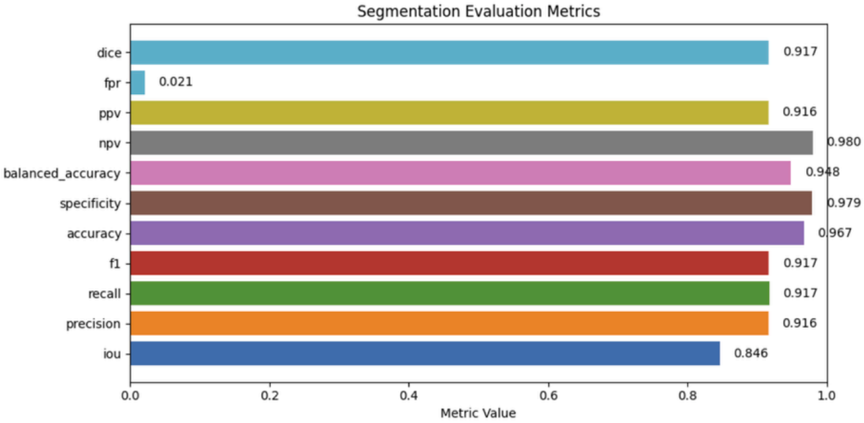

Fig. 8. LASER metrics for Test Case 1 (‘vest’) segmentation in 3DSES.

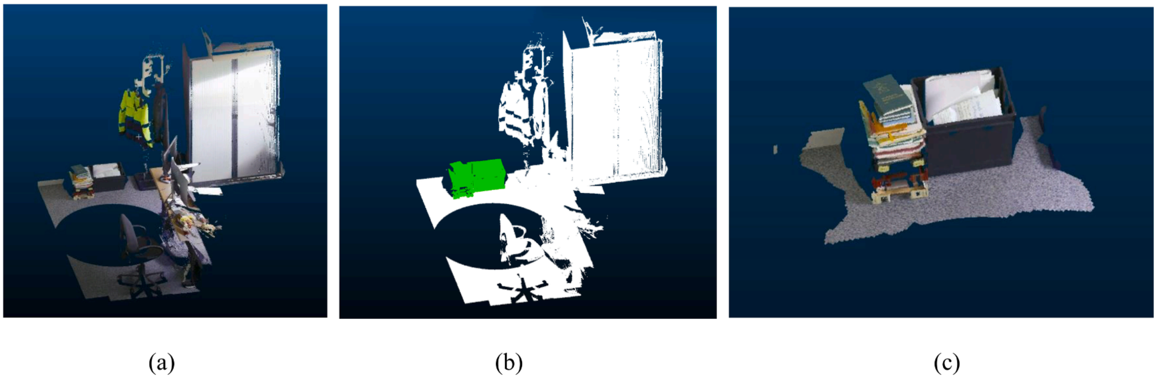

Fig. 9. (a) Input .ply file, (b) Ground Truth, and (c) LASER 3D segmentation result.

0.8420 gives a fairer representation of the result. Qualitatively, the segmentation mask is relatively clean, with few spurious points outside the target regions, but it misses about 31.6 % of the true car points due to the sparsity of the point cloud. This suggests that LASER’s performance depends on point density, as it provides reliable coarse detection of cars, but recall remains the limiting factor for fine-grained or complete object extraction. Nevertheless, within the context of object counting in 3D, LASER’s high recall ensures that most objects are represented. iv. Test Case 4: Segmentation of ‘Trees’ in Toronto3D

The text prompt “Trees” led LASER to segment vegetation across the scene, as illustrated in Fig. 13. In this case, recall was strong at 0.8750, showing that the majority of tree points were correctly identified, while precision was more modest at 0.5280. This indicates that nearly half of the predicted tree points were false positives, understandably so as the electric poles are also classified as trees. The combined F1-score (Dice) was 0.6590, with an IoU of 0.4910 (Fig. 14).

As in the car example, background rejection was essentially perfect: specificity and NPV both measured 1.0001, yielding an apparent overall accuracy of 1.0001. Again, this number is inflated by the imbalance between tree and non-tree points, and the balanced accuracy (0.9380) is more representative. Qualitatively, LASER succeeds in capturing most of the tree structure, missing only about 12.5 % of the ground-truth points. However, the moderate precision reflects the difficulty of distinguishing trees from adjacent objects (such as electric poles) and background clutter in complex outdoor environments. This makes the output valuable for initial localisation of vegetation but suggests the need for refinement to reduce segmentation noise if precise delineation is required. v. Test Case 5: Prompting for irrelevant objects in 3DSES and Toronto3D

To assess LASER’s susceptibility to false positives, we tested several

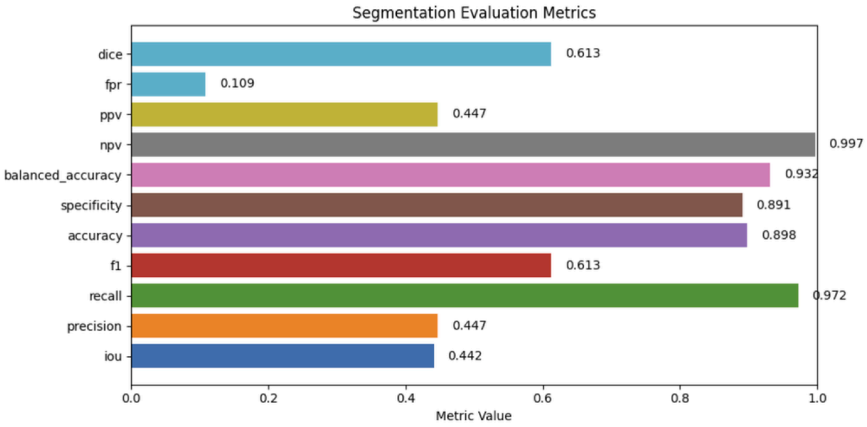

Fig. 10. LASER metrics for Test Case 2 (Pile of Books) in 3DSES.

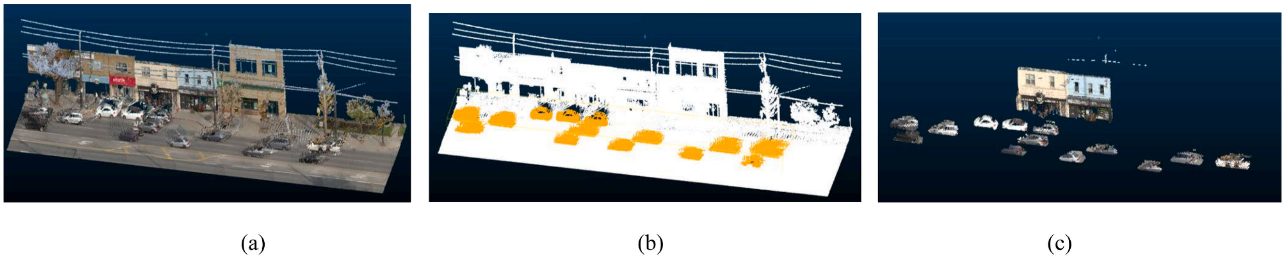

Fig. 11. (a) Input .ply file, (b) Ground Truth, and (c) LASER 3D segmentation result.

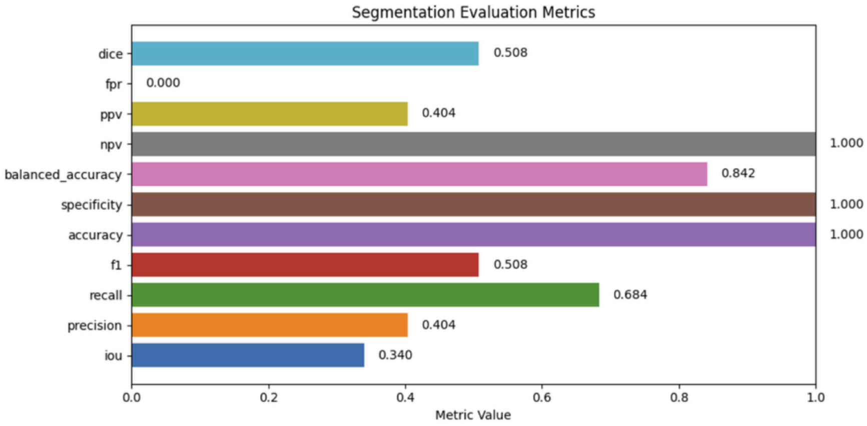

Fig. 12. LASER metrics for Test Case 3 (Cars) segmentation in Toronto3D.

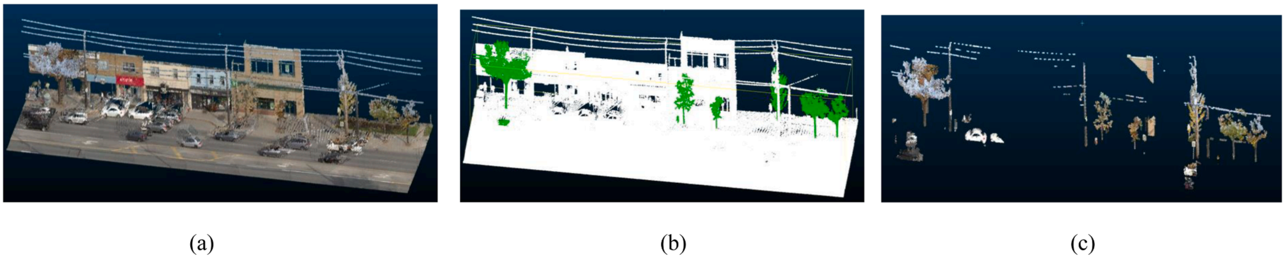

Fig. 13. (a) Input .ply file, (b) Ground Truth, and (c) LASER 3D segmentation result.

irrelevant prompts (e.g., “basket,” “dog,” “bicycle,” “chair”) on representative subsets of the 3DSES and Toronto3D datasets. In all cases, GroundingDINO returned no bounding boxes, meaning that no candidates were passed to SAM and no segmented output was produced. This confirms that LASER does not hallucinate objects in the absence of relevant visual evidence, demonstrating robustness against irrelevant

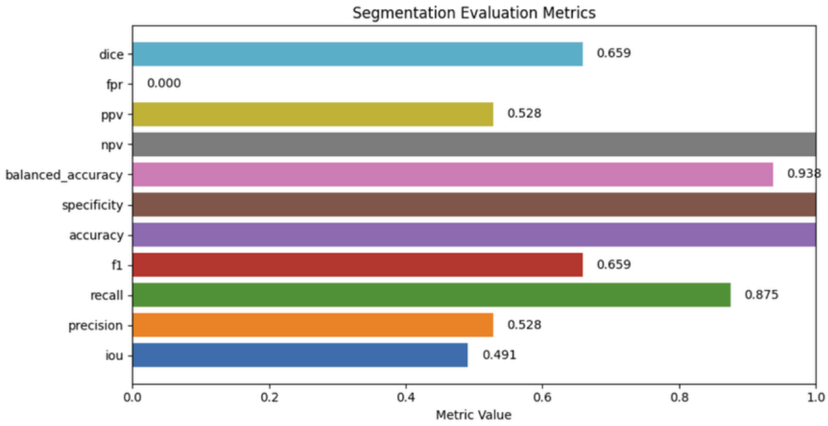

Fig. 14. LASER metrics for Test Case 4 (Trees) segmentation in Toronto3D.

prompts.

Across the five test cases, LASER demonstrated a consistent pattern: high recall ensured that most target objects were captured, but precision varied depending on class geometry and scene complexity. Simple, well-bounded objects such as the Vest were segmented cleanly with both high precision and recall, while more cluttered or ambiguous classes such as Pile of Books, Cars, and Trees showed over-segmentation, with noise arising from background structures. Balanced accuracy proved to be a more reliable indicator than overall accuracy, since class imbalance inflated the latter. Taken together, these results suggest that LASER is well-suited for applications requiring broad coverage and initial localisation of objects in 3D scenes, while tasks demanding precise boundaries or reliable object counts may require additional refinement or post-processing.

4.5. Comparison with the state-of-the-art 3D object segmentation approaches

Traditional evaluation metrics for 3D segmentation, such as mean Intersection over Union (mIoU) and point-wise accuracy, while valuable for benchmarking on standardised datasets, are not directly applicable to LASER's real-world deployment scenario. Instead, we present a feature-based comparison (Table 4) that highlights the system's practical capabilities and emphasizes its unique position in handling real-world point cloud data. Unlike existing methods that rely on standardised datasets or controlled environments, LASER was developed and tested on industrial-scale point cloud scans with varying density, completeness, and noise levels.

This real-world application context presents several challenges that make conventional metrics less meaningful: Industrial scans lack standardised ground truth annotations, making accuracy metrics impractical. Also, LASER's primary object search and immersive environment application emphasizes workflow efficiency and user control over purely algorithmic performance metrics. Moreover, the scan used exhibit significant variations in scale, density, and complexity not well-represented by standard evaluation metrics designed for controlled datasets. Hence, Table 4 presents feature-based comparison of the tool with the state-of-the-art-approaches.

As shown in Table 4, LASER distinguishes itself through its comprehensive integration of key features essential for practical deployment: multi-angle view selection, confidence-weighted fusion, language-guided prompting, and modular refinement capabilities. This combination of features, while not readily quantifiable through conventional metrics, directly addresses the practical challenges faced in real-world 3D segmentation tasks.

In summary, LASER pools together weak but complementary cues from multiple 2D views. Each render provides a noisy, text-guided guess about which points belong to the target class, and these guesses are weighted according to viewing angle and depth reliability. By combining evidence across views, similar to sensor fusion, consistent signals are reinforced while unreliable ones are suppressed. This multi-view consistency is what allows LASER to turn noisy 2D proposals into stable 3D segmentations.

Table 4
Comparison of LASER with Recent Prompt-able 3D Segmentation Methods.

Method	Multi-Angle View Selection	Optimised Fusion Algorithm	Language-Guided (Text Prompts)	Zero-Shot (No Model Retraining)	Modular & Iterative Refinement	Independent of Camera Trajectories	Direct 3D Processing (No 2D RGB Required)	Leverages Foundation Models (SAM & GroundingDINO)
LASER (Current work)	✓	✓ (Confidence-weighted)	✓ (Text-Prompted via GroundingDINO)	✓	✓	✓	✓	✓ (Uses both SAM & GroundingDINO)
SAM2Point [8]	× (Fixed multi-view projection)	× (No Fusion)	×	✓	×	✓	✓	✓ (SAM integrated for 3D point selection)
Point-SAM [9]	× (Viewpoints determined by dataset)	× (No Fusion)	×	✓	×	✓	✓	✓ (Uses SAM, but no GroundingDINO)
SAMPro3D [10]	× (Requires fixed 2D frames)	× (No Multi-View Fusion)	×	✓	×	× (Depends on camera poses)	× (Requires 2D RGB images)	✓ (Uses SAM but no direct 3D processing)
Oswald et al. (2018)	✓ (Manually Defined Views)	× (No Fusion)	×	× (Trained networks)	✓	✓	× (Uses RGB-derived 3D)	× (No SAM integration)

4.6. Limitations

The current LASER pipeline is developed to automate object detection, classification and segmentation tasks in 3D scans, hence its reliance on predefined text prompts for segmentation. However, the predefined prompts reduce adaptability to dynamic construction environments and introduce the potential to overlook objects not explicitly described. Additionally, the high computational demands of processing large-scale 3D point clouds significantly increase execution time, making real-time analysis and edge computing applications difficult. Future applications would benefit from implementing self-learning capabilities that automatically generate and refine text prompts based on construction site context, reducing manual input requirements. Moreover, the model demonstrates high accuracy and strong recall, ensuring most prompted objects are detected and extracted. However, moderate IoU and precision values highlight the need for refinement. Potential enhancements include post-processing techniques such as statistical outlier removal to reduce false positives, improved feature extraction using deep learning embeddings for better object differentiation and optimisation of ICP alignment parameters to minimise misalignment artefacts. Performance optimisation through model compression techniques, parallel processing algorithms, and cloud-based pre-processing would also significantly improve computational efficiency for real-time applications on resource-constrained devices.

Also, it is observed that LASER's performance is directly tied to input point cloud density. Sparse clouds produce rendered images with gaps and weak edges, causing SAM to miss targets and generate fragmented masks, resulting in reduced recall and over-segmentation. While mitigation strategies such as higher render resolutions and multi-view fusion provide partial improvements, dense point clouds consistently yield superior detection accuracy. This limitation is evident in the Toronto3D dataset, where sparse vehicle coverage required increased resolution parameters to achieve acceptable segmentation quality.

Finally, although GroundingDINO and SAM are trained purely in 2D, within LASER they serve only to produce aligned proposals in image space. The reprojection and probabilistic fusion steps are what transport these proposals into 3D, effectively disentangling semantic grounding from geometric regularisation. As a result, LASER inherits the open-vocabulary property of the underlying language-vision model, while the multi-view aggregation supplies the missing geometric constraints. Adaptability is therefore guaranteed as long as (i) the target class has language-describable cues, and (ii) its distinguishing visual features are visible in at least one selected view. The result across multiple scans show that LASER failure modes occur when these assumptions break down. Hence it is pertinent to mention a number of failure modes and the mitigations. First, geometry (e.g., repetitive glass façades) can yield uncertain proposals; this can be mitigated by broader prompts (e.g., "window or glass panel") and increasing the number of views K. Secondly, extreme sparsity leads to fragmented back-projections; this is mitigated by relaxing thresholds, increasing clustering radius ϵ, or rasterizing through mesh reconstruction. Moreover, rare-class queries can trigger semantic drift or hallucinations; this is mitigated by negative-control prompts and requiring multi-view consistency before acceptance.

5. Conclusion

This work introduces a novel language-guided 3D segmentation framework (LASER) that integrates multi-angle reprojection and confidence-weighted fusion to achieve high-precision parsing of point clouds. Unlike existing methods that rely on predefined viewpoints or fixed 2D images, LASER dynamically selects multi-view projections and iteratively refines segmentation results through a confidence-aware fusion mechanism. By leveraging Grounded-SAM as its backbone, LASER operates in a zero-shot manner, eliminating the need for model retraining and making it highly adaptable to diverse real-world scans.

Our results demonstrate LASER's robustness in segmenting cluttered and safety-critical environments, particularly for generating 3D models for VR-based construction training applications. The method effectively identifies key site elements without requiring manual annotations, thereby reducing the human effort needed for large-scale scene parsing. Furthermore, the ability to refine segmentation iteratively further ensures that the framework can mitigate occlusions, variable lighting conditions, and sensor inconsistencies commonly found in outdoor construction sites.

Future work will focus on extending LASER to support real-time segmentation, larger-scale urban environments, and dynamic scene analysis. Additionally, integrating more advanced natural language understanding could further enhance the adaptability of text-prompted 3D segmentation. Overall, LASER represents a significant step forward in AI-driven construction site analysis, providing a flexible, modular, and scalable solution for automated 3D scene understanding.

CRediT authorship contribution statement

Abiodun Ayodeji: Writing – original draft, Validation, Software, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Ahmed Teyeb:** Writing – review & editing, Project administration, Methodology, Investigation, Formal analysis. **Mohmmad Ali Asgar Abbas:** Writing – review & editing, Visualization, Validation, Software, Methodology. **Paul Bass:** Writing – review & editing, Investigation, Formal analysis, Data curation. **Emma Bass:** Formal analysis, Data curation. **Prasanna D. Bandara:** Methodology, Investigation, Formal analysis, Data curation. **Udari K. Jayasinghe:** Methodology, Formal analysis, Data curation. **Jamie Griffiths:** Validation, Formal analysis, Data curation. **Evelyne El Masri:** Writing – review & editing, Resources, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported by the SafeXtend project, which was funded by Innovate UK under the Collaborative AI Solutions to Improve Productivity in Key Sectors programme (Grant No. 10102820). The authors would like to acknowledge the contributions of all project partners for their support in data collection and tool testing, which facilitated the development and validation of the proposed methodology.

Data availability

Data will be made available on request.

References

[1] A.M. Esmorís, H. Weiser, L. Winiwarter, J.C. Cabaleiro, B. Höfle, Deep learning with simulated laser scanning data for 3D point cloud classification, ISPRS J. Photogramm. Remote Sens. 215 (2024) 192–213, https://doi.org/10.1016/j.isprsjprs.2024.06.018.

[2] Yang, S., Xu, S., & Huang, W. (2022). 3D Point Cloud for cultural heritage: a scientometric survey. In Remote Sensing (Vol. 14, Issue 21). MDPI. https://doi.org/10.3390/rs14215542.

[3] M. Soori, B. Arezoo, R. Dastres, Digital twin for smart manufacturing, Rev. Sustain. Manuf. Serv. Econ. 2 (2023) 100017, https://doi.org/10.1016/j.smse.2023.100017.

[4] A. Voulodimos, N. Doulamis, A. Doulamis, E. Protopapadakis, Deep learning for computer vision: a brief review. Computational Intelligence and Neuroscience, Hindawi Limited, 2018, https://doi.org/10.1155/2018/7068349. Vol. 2018.

[5] Chen, X., Ma, H., Wan, J., Li, B., & Xia, T. (2017). Multi-view 3D object detection network for autonomous driving. Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017-January, 6526–6534. https://doi.org/10.1109/CVPR.2017.691.

[6] Rebuffi, S.-A., Kolesnikov, A., Sperl, G., & Lampert, C.H. (n.d.), 2020. iCaRL: incremental classifier and representation learning.

[7] Yang, Y., Wu, X., He, T., Zhao, H., & Liu, X. (2023). SAM3D: segment anything in 3D scenes. http://arxiv.org/abs/2306.03908.

[8] Guo, Z., Zhang, R., Zhu, X., Tong, C., Gao, P., Li, C., & Heng, P.-A. (2024). SAM2Point: segment any 3D as videos in zero-shot and promptable manners. http://arxiv.org/abs/2408.16768.

[9] Zhou, Y., Gu, J., Chiang, T.Y., Xiang, F., & Su, H. (2024). Point-SAM: promptable 3D segmentation model for Point clouds. http://arxiv.org/abs/2406.17741.

[10] Xu, M., Yin, X., Qiu, L., Liu, Y., Tong, X., & Han, X. (2023). SAMPro3D: locating SAM prompts in 3D for zero-shot scene segmentation. http://arxiv.org/abs/2311.17707.

[11] Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.-Y., Dollár, P., & Girshick, R. (2023). Segment anything. http://arxiv.org/abs/2304.02643.

[12] Li, C.R.Q., Hao, Y., Leonidas, S., & Guibas, J. (n.d.), 2020. PointNet++: deep hierarchical feature learning on point sets in a metric space.

[13] Qi, C.R., Su, H., Mo, K., & Guibas, L.J. (n.d.), 2020. PointNet: deep learning on point sets for 3D classification and segmentation.

[14] Michele, B., Boulch, A., Puy, G., Bucher, M., & Marlet, R. (2021). Generative zero-shot learning for semantic segmentation of 3D point clouds. http://arxiv.org/abs/2108.06230.

[15] Wang, Y., Huang, S., Gao, Y., Wang, Z., Wang, R., Sheng, K., Zhang, B., & Liu, S. (2023). Transferring CLIP's knowledge into zero-shot point cloud semantic segmentation. MM 2023 - Proceedings of the 31st ACM International Conference on Multimedia, 3745–3754. https://doi.org/10.1145/3581783.3612107.

[16] Zhu, X., Zhang, R., He, B., Guo, Z., Zeng, Z., Qin, Z., Zhang, S., & Gao, P. (2022). PointCLIP V2: prompting CLIP and GPT for powerful 3D open-world learning. http://arxiv.org/abs/2211.11682.

[17] Zhao, W., Yan, Y., Yang, C., Ye, J., Yang, X., & Huang, K. (n.d.), 2023. Divide and conquer: 3D point cloud instance segmentation with point-wise binarization. https://github.com/weiguangzhao/PBNet.

[18] Ravi, N., Gabeur, V., Hu, Y.-T., Hu, R., Ryali, C., Ma, T., Khedr, H., Rädle, R., Rolland, C., Gustafson, L., Mintun, E., Pan, J., Vasudev Alwala, K., Carion, N., Wu, C.-Y., Girshick, R., Dollár, P., Feichtenhofer, C., & Fair, M. (n.d.), 2020. SAM 2: segment anything in images and videos.