# SAM4Tun

Ye, Zehao; Lin, Wei; Faramarzi, Asaad; Xie, Xiongyao; Ninic, Jelena

*DOI:*
10.1016/j.tust.2025.106401



*Document Version*
Publisher's PDF, also known as Version of record

*Citation for published version (Harvard):*
Ye, Z, Lin, W, Faramarzi, A, Xie, X & Ninic, J 2025, 'SAM4Tun: No-training model for tunnel lining point cloud component segmentation', *Tunnelling and Underground Space Technology*, vol. 158, 106401. https://doi.org/10.1016/j.tust.2025.106401

Tunnelling and Underground Space Technology 158 (2025) 106401

Contents lists available at ScienceDirect

Tunnelling and Underground Space Technology

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

# SAM4Tun: No-training model for tunnel lining point cloud component segmentation

Zehao Ye [a], Wei Lin [b], Asaad Faramarzi [a], Xiongyao Xie [b], Jelena Ninić [a,*]

[a] University of Birmingham, UK
[b] Tongji University, China

ARTICLE INFO

*Keywords:*
Tunnel lining
Zero-shot model
Point cloud segmentation
Segment anything model
Prompt engineering

ABSTRACT

Asset management ensures the safety and longevity of structures through regular maintenance. Reality capture technologies are increasingly being used for asset inspections to obtain information by generating point cloud data, which is becoming more prevalent in tunnel asset management for precise documentation of tunnel geometry and condition. Integrating semantic information from point clouds is crucial for creating accurate as-built Building Information Models (BIM), essential for project delivery, maintenance, and operations. In this paper, we propose SAM4Tun, a zero-shot automated instance segmentation method for tunnel lining segments. It is based on a Large Vision Model (LVM), prompt-based Segment Anything Model (SAM), and various point cloud and image processing techniques, enabling accurate instance segmentation without requiring any training. The process starts by unfolding tunnel point clouds to generate 2D panoramic images, enabling SAM to be extend its capabilities to point cloud segmentation. To enhance performance, we propose: (i) a local point cloud density-variation method to filter out non-segment parts, and (ii) a geometry feature-guided multi-step point cloud up-sampling method to address uneven point cloud density during projection. Then, we focus on prompt engineering, using traditional image processing techniques to automatically generate template prompt, enabling SAM's zero-shot ability to achieve precise instance-level segmentation of tunnel linings. The results demonstrate that our no-training model achieved highly accurate instance segmentation, even surpassing supervised learning algorithms. The proposed method addresses the issue of data dependency and serves as the foundation for component-level damage localization and displacement monitoring in tunnel. Our code is available at https://github.com/zxy239/SAM4Tun.

## 1. Introduction

Expansion of underground transportation infrastructure is gaining attention as a solution to rapid urbanization, driven by the need for eco-friendly transportation and high-speed mobility (Huang et al., 2021b; Ninić and Meschke, 2015). An average of 5200 km of tunnels are being built annually worth an investment of €125 billion (ITA-AITES, 2021), and thanks to advancements in Tunnel Boring Machine (TBM) technology that handle diverse ground conditions more efficiently (Ninić et al., 2017; Gong et al., 2016). However, challenges in ensuring the safety and quality of tunnel construction and operation remain key concern for the industry and a primary interest of academia (Ninić et al., 2020; Li et al., 2019; Lin et al., 2023a). Comprehensive inspections are essential to guarantee safe subway construction and operation.

Traditional tunnel inspections primarily rely on periodic manual patrols, where inspectors visually detect structural deterioration (like cracks, seepage, spalling) or deformations, manually mapping and recording problem areas, which makes the process time-consuming, labour-intensive, and heavily dependent on the inspector's experience (Attard et al., 2018; Huang et al., 2021b). In response to these challenges, point cloud-based automated monitoring is gaining popularity in tunnels due to its high accuracy and efficiency in measuring cross-sectional convergence (Xie and Lu, 2017). Terrestrial Laser Scanning (TLS) generates point clouds as digital replicas of tunnels, capturing spatial coordinates, intensity, and colour to evaluate structural conditions (Menendez et al., 2018). In recent years, Mobile Laser Scanners (MLS) have been driving innovation in tunnel measurement technology. Mounted on mobile trolleys on rails, these systems use laser scanners as the primary sensor and integrate some other devices like odometers, displacement sensors, Inertial Measurement Units (IMUs), to efficiently collect tunnel point cloud data on the move (Sun et al., 2020; Cui et al., 2019). Overall, in subway tunnel point cloud acquisition, TLS features fixed equipment, minimizing the impact of

* Corresponding author.
*E-mail addresses:* zxy239@student.bham.ac.uk (Z. Ye), j.ninic@bham.ac.uk (J. Ninić).

https://doi.org/10.1016/j.tust.2025.106401
Received 24 October 2024; Received in revised form 10 January 2025; Accepted 10 January 2025
Available online 24 January 2025

motion and vibration, resulting in high consistency and better accuracy. However, it is less efficient, requiring frequent station setups, complex operations, and post-processing for data stitching. In contrast, MLS, on the other hand, offers high efficiency and flexibility, making it suitable for large-scale, rapid scanning along tracks. However, it is limited by the positioning system within tunnels, and the accuracy may be affected by vibration and movement speed during scanning. Therefore, TLS is ideal for high-precision local monitoring and small-scale detailed modelling, while MLS is better suited for overall structural mapping, particularly for long, linear, and continuous scanning. Both technologies are commonly applied, and the choice of the most suitable one depends on the primary objectives of the survey.

These technology provides crucial technical and data support for both dimensional deformation analysis and defect detection in railway tunnels as well as their full lifecycle management (Cui et al., 2024; Huang et al., 2021b). For example, geometric algorithms can measure longitudinal displacements and spatial differences between lining rings (Lin et al., 2022), while segment-wise position calculations provide detailed cross-sectional deformation data, including segment joints (Cao et al., 2021; Yi et al., 2019; Zhang et al., 2024). The structural data obtained from deformation measurements is valuable for reverse mechanical analysis (Xu et al., 2019; Lin et al., 2023b; Xu et al., 2023). For defect detection, such as seepage (Chen et al., 2024b; Menendez et al., 2018) and spalling (Xie et al., 2021; Huang et al., 2020; Zhou et al., 2021), it can identify and quantify defects more effectively compared with 2D images, including calculating spalling depth and leakage areas. Overall, the use of point clouds for structural assessment is well-established, covering the measurement of various deformations and the detection of multiple defects, providing comprehensive and rigorous data sources of structural conditions.

Furthermore, all the mentioned applications can benefit from a crucial prerequisite: point cloud segmentation, that enables automated identification of the points corresponding to linings or segments ahead of deformation calculation or defect detection. This is important as precise segmentation of structural elements enables the acquisition of higher-fidelity deformation measurements and more accurate segment-wise localization of defects, thereby enhancing the infrastructure management workflow (Lin et al., 2024; Liu et al., 2022; Li et al., 2024). Segmented tunnel lining have a relatively straightforward classification and different material compositions (Zhang et al., 2022; Lin et al., 2024). Many researchers have implemented and continuously improved tunnel component segmentation performance, and attempt to automatically provide reliable data sources for constructing as-built BIM that align with tunnel construction management and operations (Cheng et al., 2019; Hegemann et al., 2020). Tunnel lining segmentation primarily includes three methods: (i) direct segmentation based on point clouds, like ASPCNet (Zhou et al., 2023), Seg2Tunnel (Lin et al., 2024), SPCNet (Ji et al., 2023), etc., (ii) voxel-based segmentation (Cheng et al., 2019; Ji et al., 2022), and (iii) projection-based method (Zhang et al., 2022; Chen et al., 2024b). The above-mentioned methods all require the process of training. Such emerging deep learning algorithms show promising performance, however its development is hinged by the scarcity of training data. In the available literature there is only a small portion of research that combine prior knowledge and do not require training for segmentation, relying solely on traditional digital image processing techniques (Yi et al., 2019; Duan et al., 2021; Zhang et al., 2024). However, these methods generally have significant limitations, which are suitable mainly for very simple environments with only shield segments and require high point cloud density.

Recently, a Large Vision Model (LVM), the prompt-based Segment Anything Model (SAM), has been released by MetaAI (Kirillov et al., 2023). Pre-trained on the extensive SA-1B dataset (11 million images with 1 billion masks), SAM showcases impressive zero-shot segmentation capabilities by generating masks based on user prompts. The user prompts here refer to some auxiliary information or cues, such as text prompts, graphical prompts (points, bounding boxes, rough boundaries), or rough mask prompts, used to guide or assist the model in recognizing and segmenting target objects (Lüddecke and Ecker, 2022). While SAM claims to segment any object, its performance can vary across different scenarios. Specifically, there are some concerns about applications such as medical image segmentation, camouflaged object detection, mirror/transparent object detection, etc. However, specific fine-tuning for SAM can still successfully accomplish these tasks (Wang et al., 2023a). Similarly, recent studies utilizing SAM in civil engineering area have explored and also been divided into two directions due to variations in the types of objects being detected. The first direction involves using ground truth (GT) masks to generate similarity maps to guide the generation of prompt points (Wang et al., 2024), or manually input initial prompts and then use the overlapping parts to propagate the prompt (Kang et al., 2024). While the other direction is introducing fine-tuning modules to train SAM on specific dataset (Ye et al., 2024; Ge et al., 2024). Both methods have yielded state-of-the-art (SOTA) results due to SAM's powerful pre-training foundation for image segmentation.

In this paper, we propose a novel approach to tunnel lining point cloud component segmentation, called "SAM4Tun", by leveraging the zero-shot segmentation capability of LVM and the highly structured nature of TBM tunnels to achieve automatic segmentation with superior accuracy. Unlike previous methods that relied on similarity-based prompts or fine-tuning SAM with specific dataset, we achieve segmentation purely through SAM's zero-shot capabilities by projecting point clouds into images and focusing on image pre-processing and prompt engineering. Our hypothesis is that the tunnel lining segments we aim to detect are characterized with clear and distinct features, enabling acceptable results to be achieved without incorporating fine-tuning. The automatic generation of prompts also comes from training-free digital image processing techniques. To enhance SAM's instance segmentation performance for lining segment, we introduce noise filtering based on local point cloud density variation and geometry-guided multi-step point cloud up-sampling to improve the quality of images input to SAM. The final results demonstrate that our no-training model outperforms even supervised learning algorithms. The remainder of this paper is structured as follows. Section 2 discusses related work, including projection-based TBM tunnel point cloud segmentation and SAM's prompt engineering in the field of civil engineering. Section 3 explains our proposed method in detail. Section 4 and Section 5 demonstrate the implementation of our models and present a comparison of the results using the Seg2tunnel dataset (Lin et al., 2024) against other methods, and we also show the model's generalization capability across different tunnels and data types (TLS (Lin et al., 2024) and MLS (Cui et al., 2024)). Finally, Section 6 summarizes this paper and introduces limitations and future work.

## 2. Related works

Projection-based approach, presents 3D point cloud as 2D images, by using intermediate representation to convert the unstructured and irregular point cloud into structured and regular images, allowing seamless integration with many advanced and diverse image-based segmentation algorithms (Zhang et al., 2022). The application of this method to tunnel point clouds typically begins by extracting the centre line to use as the projection axis. This is followed by mapping the points using a cylindrical coordinate system based on the obtained projection axis, and finally generating the unwrapped image.

Qiu and Cheng (2017) used boundary point extraction, the random sample consensus (RANSAC) algorithm (Fischler and Bolles, 1981), and least squares adjustment to extract horizontal and vertical alignments from point cloud to define tunnel stationing coordinate system. Similarly, Zhang et al. (2024) fitted boundaries of tunnel on both sides using cubic polynomials from two projection directions (top view and side view), and then, by extracting middle points, parametric equation of tunnel centre line was calculated. However, this method is affected

by the bolt holes in the segment depressions and becomes even more unstable when the segments do not fit well. In addition to using boundary points, another method involves utilizing the fitted centre points of tunnel cross-sections. These cross-sections, or slices of the point cloud, are typically obtained by extracting point clouds within a specified width using two perpendicular planes aligned with a specified axis (Zhang et al., 2024). Zhang et al. (2022) extracted peak intervals along the tunnel direction, or directly used design centre line (Yi et al., 2019), to obtain point cloud slices, and then determine their centres through least-squares circle fitting, or RANSAC algorithm (Shi et al., 2023), to update the centre axis with a spatial curve. Subsequently, coordinate transformation is performed based on this axis, converting the 3D tunnel point cloud into a 2D unfolded point cloud. Some researchers (Cui et al., 2024) also attempt to project the tunnel point cloud into a 2.5D format, allowing the unfolding method to be applicable not only to cylindrical tunnels but also to retain specific local features. However, regardless of the method, perfect axis fitting remains impossible due to the tunnel's complex curvature, the reliance on limited local point sampling, and more fundamentally, construction errors that result in imperfect geometric cross-sections.

The next step is how to convert unwrapped point cloud into images. Specifically, first, set a resolution to create a blank canvas, then map the unfolded point cloud to corresponding pixel positions, and then fill the pixels based on attributes of points like intensity, depth (distance from the cylindrical coordinate system's Z-axis), colour, etc. During the filling process, when multiple points are mapped to the same pixel, the mean (Zhang et al., 2022), or median (Duan et al., 2021) of the corresponding attributes of these points is calculated. Additionally, various colour channel selections for projected image generation have been tried. Yi et al. (2019) used intensity of laser. While Zhang et al. (2022) and Duan et al. (2021) concluded that the intensity is not stable enough due to various factors such as incident angle, reflective material, distance, etc, so they chose to establish a depth map. Shi et al. (2023) tested different channel combinations including intensity, depth, and 2.5D local depth, finding that combinations involving local depth typically yielded better segmentation results for most of segmentation algorithms. They also recognized the issue of poor adaptability with segmentation algorithms developed based on RGB channels. Zhang et al. (2024) though that intensity is easily disturbed inside tunnel, while the accuracy of depth maps is influenced by the precision of axis fitting. Therefore, they propose a new method by using the angle between the normal vector of the unfolded plane and the point to generate binary value for filling. Furthermore, most studies assume a sufficient point cloud density or use MLS, and emphasize importance of sufficient density (Duan et al., 2021). However, in some cases, mismatches between point cloud and canvas resolution still happened leading to unfilled pixels. For pixels not associated with any points and thus left unfilled, some studies (Zhang et al., 2022; Duan et al., 2021) leave them empty, while others use surrounding values for interpolation filling (Cui et al., 2024). In general, previous research has explored various channel combinations for generating unfolded views, and preserving local features consistently yields better results. Additionally, since the image-based methods used are designed for RGB, they come with certain limitations. Furthermore, projection-based method also have high demands for point cloud density.

Overall, we reviewed the fitting of the tunnel axis and the methods for generating unfolded views. The projection process results in some loss of point attribute information, but it also achieves noise reduction. However, due to various influencing factors, it is impossible to perfectly fit the central axis. Additionally, preserving as many local features of the point cloud as possible generally yields the better results. This is reasonable because deviations in the central axis or construction are generally insignificant in the local regions of the unfolded view. Furthermore, from an image segmentation perspective, maintaining and enhancing local differences is clearly beneficial for segmentation. For gappy and poor quality point clods, the question of whether to leave gaps empty or fill them through interpolation requires further exploration, particularly when there is significant variation in point cloud density. Empty pixels can either provide valuable information or mislead the segmentation model. Nevertheless, regardless of the approach, the images obtained after this process can be combined with advanced and diverse image segmentation algorithms, such as SAM (Ke et al., 2023), to achieve leading segmentation results. Moreover, the algorithms previously used were all designed for RGB and pre-trained on RGB images. Although LVM is same, it has been shown to generalize well to images composed of non-RGB channels, thanks to its pre-training on the largest dataset to date (Kang et al., 2024; Ding et al., 2024).

Recent advancements in different image segmentation task have increasingly utilized LVM with sophisticated architectures (Wang et al., 2023b; Yu et al., 2023; Chen et al., 2021). These models benefit from scaling laws (Kaplan et al., 2020), where performance improves with larger pre-trained datasets and model sizes, enhancing accuracy and adaptability across different tasks. SAM is one of the most representative LVMs currently. Its structure consists of three main components shown in Fig. 1: image encoder, prompt encoder and mask decoder (Kirillov et al., 2023). Image encoder is a Vision Transformer (ViT) architecture, for extracting feature from the input image. Prompt encoder is designed for accepting prompts (spare prompts, like points, bounding boxes, and dense prompts like coarse masks) and generate prompt embeddings. This module enables SAM to adapt to a wide range of downstream tasks by guiding the model to generate the desired outputs (Wang et al., 2023a; Chen et al., 2024a). The image and the prompt together form the complete input of SAM (Ng et al., 2023). Finally, mask decoder can map information from the above two sections to corresponding masks. SAM generates three masks (subpart, part, and whole as illustrated in Fig. 1) for each prompt, returning the one with the highest confidence score (Kirillov et al., 2023). More specific negative/background prompts can reduce ambiguities and yield the exact segmentation mask the user needs. Therefore, SAM is a prompt-based universal image segmentation model without providing semantic information.

Based on these characteristics, the most logical task is to focus on how to generate appropriate prompts to obtain desired masks without any model weights updating. Such process is known as prompt engineering (Wang et al., 2023a). Furthermore, visual prompting refers to interacting with a pre-trained model to achieve a specific task (Ng et al., 2023). Some studies have already been conducted based on this approach. Given a user-provided reference image, PerSAM (Zhang et al., 2023) utilized the output feature maps from SAM image encoder to compute cosine similarity. This process enables the generation of foreground and background prompt point, achieving user-defined segmentation. Wang et al. (2024) introduced a method using a limited number of Mechanical, Electrical, and Plumbing (MEP) pipeline masks, employing DINOv2 (Oquab et al., 2024) to generate similarity maps between the desired detection image and these masks, and then selecting high-similarity pixels as prompt points to guide SAM in segmentation. Kang et al. (2024) sliced the mining tunnel point cloud along its direction, projecting these slices onto a 2D image, and then manually set prompt points to achieve semantic segmentation of the entire tunnel progressively and recursively. Combining TBM tunnel point cloud segmentation tasks with SAM remains a highly novel area of research. Notably, TBM tunnel point clouds are generally regular in shape and highly suitable for prompt generation, presenting significant potential.

In summary, tunnel point clouds have broad applications in project construction, operation, monitoring, etc. Capturing their semantic and geometric information in an automated way for establishing as-built BIM is highly significant, particularly using advanced algorithms for segmenting tunnel lining segments. Supervised algorithms require detailed annotated datasets, while unsupervised ones perform worse and still need training. Traditional image processing train-free methods can

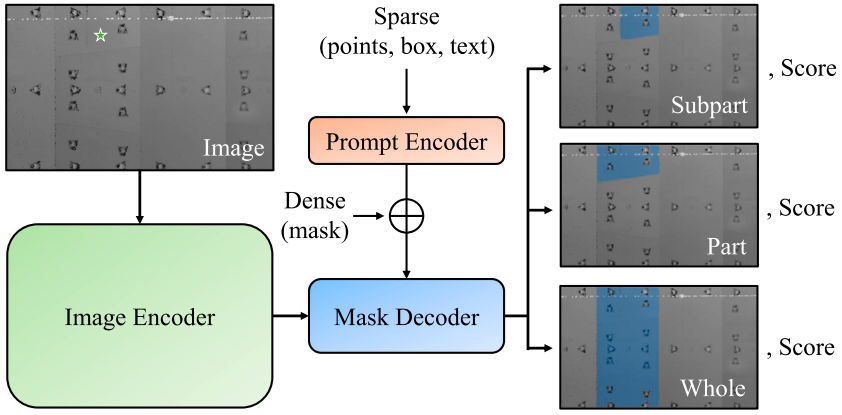


**Fig. 1.** Basic architecture of the Segment Anything Model (Kirillov et al., 2023).

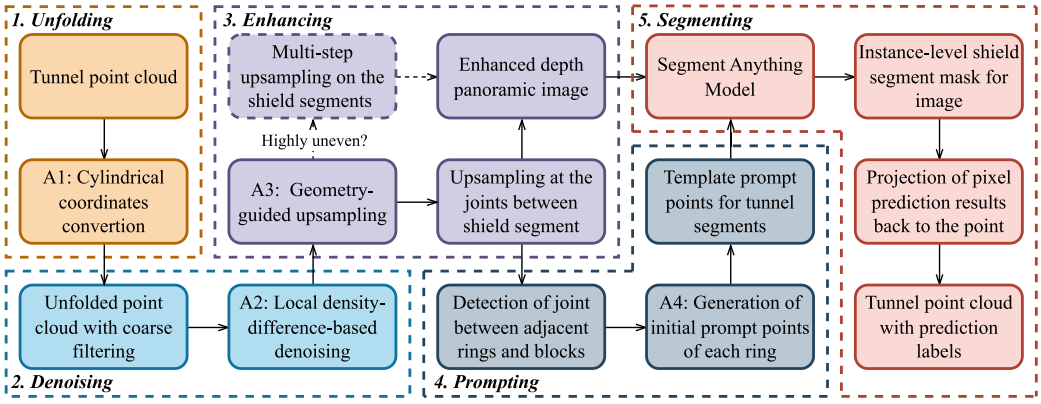


**Fig. 2.** Overview of proposed methodology.

only handle tunnels with minimal interference with high point cloud density. Using methods based on 2D unwrapped projection images can utilize diverse and advanced image segmentation algorithms, including state-of-the-art LVM like SAM, showing significant advancements when accept suitable prompts and requiring no training. To achieve better segmentation results, the process of generating images through projection needs careful examination. From literature review, it is crucial to explore how to preserve as much local point cloud information as possible during projection to enhance segmentation. Additionally, when point cloud density does not match canvas resolution, strategies for preserving more information when filling in empty pixels require further investigation. Overall, introducing zero-shot method provides a completely new approach for tunnel lining point clouds instance segmentation, which addresses data dependency and eliminates the need for training.

## 3. Methodology

### 3.1. Overview

This paper proposes a novel framework for instance-level tunnel segment identification based on point cloud projection and SAM, without requiring any training. The proposed method, SAM4Tun, consists of five steps: unfolding, denoising, enhancing, prompting, and segmenting. It is primarily composed of four algorithms (A1 to A4) along with SAM, as illustrated in Fig. 2. In general, we perform unfolding of the point cloud, denoising by removing non-segment points, enhancing boundaries during the projection into a panoramic image, automatically prompting SAM for segmenting, and finally re-projecting the results back into the point cloud.

In the unfolding method, we introduce a method that uses the RANSAC algorithm (Fischler and Bolles, 1981) to fit the central curve of the tunnel point cloud, and then unfolded point cloud based on this central curve (see Algorithm 1, Salmi et al. (2025)). Subsequently, for denoising we first perform some coarse filtering, and then we introduce a new method for fine denoising (Algorithm 2) of points corresponding to tracks, pipelines, supports, and other non-segment components based on local point cloud density variations. The next step is to introduce enhancing algorithm, which includes a outlier detection method for extracting and enhancing segment joints (Algorithm 3-1), and a multi-step geometry feature-guided point cloud up-sampling method to enhance the surface of shield segment when the point cloud density of the tunnel is highly uneven (Algorithm 3-2). This can enhance the subsequent generation of the unfolding panoramic image (Algorithm 3-3), facilitating improved segmentation. Following that, we focus on generating prompt for SAM (Algorithm 4). We utilize the results from the up-sampling at the joints, and apply prompting algorithm, using the Hough Transform (Illingworth and Kittler, 1988) to detect oblique boundary segments between K-block and B-block within each ring. The centre point of the K-block is chosen as the initial prompt point for each ring, and some template prompts are created based on these points. The initial points-based cropped sub-image, which derived from enhanced panoramic image, along with the corresponding prompts are then sent to SAM for instance segmentation. Finally, we obtain pixel-level segmentation results and re-project them onto the tunnel point cloud. The following sections will detail each step.

### 3.2. Conversion of TBM point cloud to cylindrical coordinates

Fig. 3(a) shows the original tunnel point cloud, which is a curved-axis cylinder. To achieve the unfolding, parallel point cloud slices

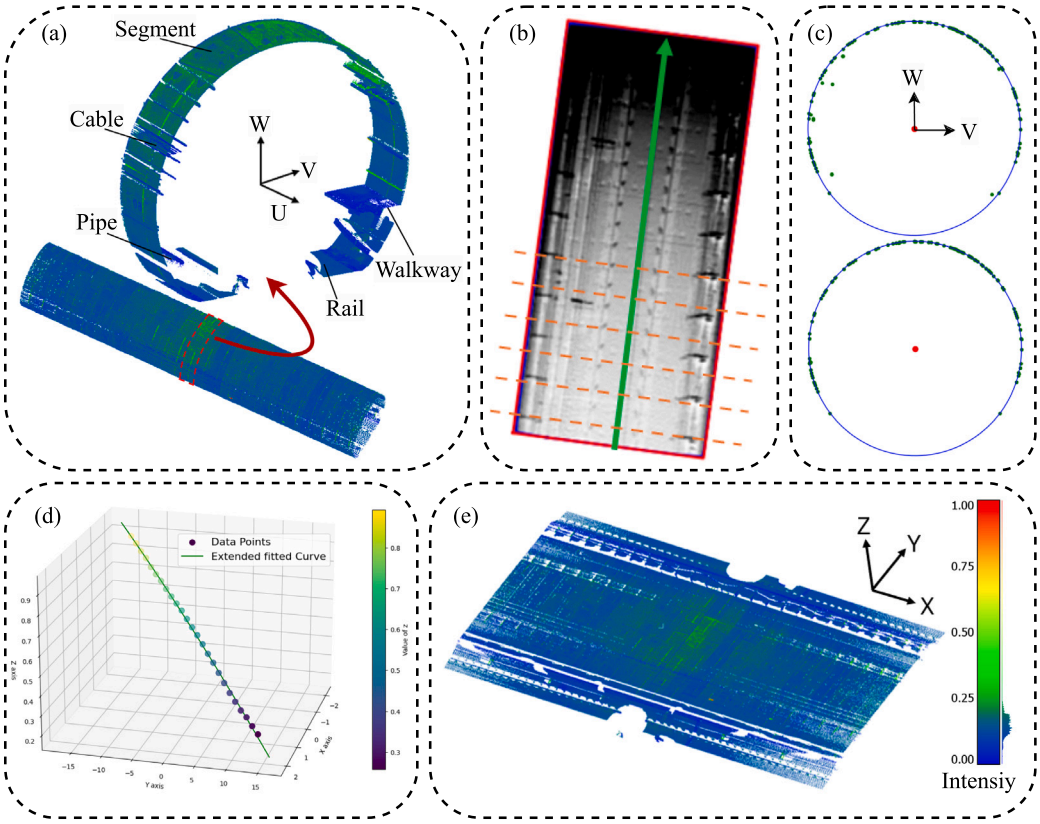


**Fig. 3.** Diagram illustrating the process of unfolded tunnel point cloud: (a) original point cloud; (b) sliced plane (orange one) and direction vector (green one); (c) two-pass RANSAC ellipse fitting (blue circle is the fitted circle; green points are the input for fitting; red point represents the centre of the fitted ellipse); (d) centre point and curved central axis; (e) unfolded TBM point cloud. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

(orange) were generated based on a single direction vector (green), as illustrated in Fig. 3(b). We use such a simple single-direction vector and parallel slicing method for fitting the central axis because our approach places a greater emphasis on local features, significantly reducing the impact of central axis fitting errors. Based on the review, we used the RANSAC algorithm to fit the ellipse of the slice point clouds and the parameter equation of the centre curve. RANSAC's advantage is its ability to provide a good fit under noisy conditions and its lower sensitivity to outliers. During the central axis fitting process, we initially apply a rough W-value filter in the W direction of the 2D (W-V plane) point cloud to exclude points below the walkway. Then we iterated twice for centre points, and one time for centre curve, as shown in Fig. 3(c) and (d). According to the testing, two iterations were sufficient and additional steps do not bring significant improvements. Subsequently, cylindrical coordinates for each point in the TBM point cloud were calculated relative to the fitted centre curve. For the specific process, please refer to Algorithm 1. Since our goal for projection is to obtain the geometrical features of the unfolded point cloud and generate a panoramic unfolded map, we introduce a fast and sufficiently accurate approximate projection method. This involves two steps: pre-computation and parallel computation of points.

In the pre-computation step, we sample points at 1 mm intervals along fitted spatial curve segments, designating these as point set $\{B\}$. For each point $B$, we calculate:

1. the arc length $l_B$, measured from the starting point of the curve segment to point $B$;
2. the tangent vector $\vec{T}_B$, which is tangent to the curve and directed from the starting point towards the endpoint of the curve;
3. point $C$, where $\vec{BC} = \lambda\vec{T}_B + \vec{Z}$ with $\lambda$ being a scaling constant, and $\vec{Z}$ is the unit vector in the Z direction. Point $C$ lies on the plane $M_B$, which is perpendicular to $\vec{T}_B$ and passes through point $B$;
4. a K-dimensional Tree (KD-Tree) based on the set of points $B$.

As for parallel point computation step, for any point $A$ in the point cloud, we first use the KD-Tree to find the corresponding nearest point $B$. Then, we obtain the cylindrical coordinates of each point $A$:

1. height coordinate $(z)$ : $l_B$;
2. radial coordinate $(r)$ : the distance $A'B$, where $A'$ is the projected point of $A$ onto the corresponding plane $M_B$;
3. angular coordinate $(\theta)$ : the angle $\angle A'BC$. If point $A$ is directly below point $B$, it is set as 0 degrees, and the degrees increase in a clockwise direction as viewed along the centre curve.

The final unfolded representation used height coordinate as the $X$-axis, angular coordinate (scaled by circumference) as the $Y$-axis, radial coordinate as the $Z$-axis, and the tunnel was sliced from directly below (0 degree), shown in Fig. 3(e).

### 3.3. Denoising based on local point cloud density differences

Non-segment parts within tunnel structure point cloud, such as cables, walkways, rails, etc., (see Fig. 3(a)) can be directly filtered by passthrough filtering on the unfolded point cloud based on a specific Z value. However, regardless of the method used, it cannot perfectly determine the central axis for cylindrical unfolding, and as a result, the segments are typically not flat Cui et al. (2024). Additionally, the installation of segments also leads to some unevenness due to construction errors, like Fig. 4(a). Therefore, simply using only one global Z value cannot filter these non-segment parts with sufficiently accurate.

There is extensive research on point cloud filtering, but strategies typically need to be tailored to specific situations (Han et al., 2017). Therefore, inspired by Density-Based Spatial Clustering of Applications with Noise (DBSCAN) (Ester et al., 1996), for our unfolded tunnel case, we propose a simple method based on local point cloud density

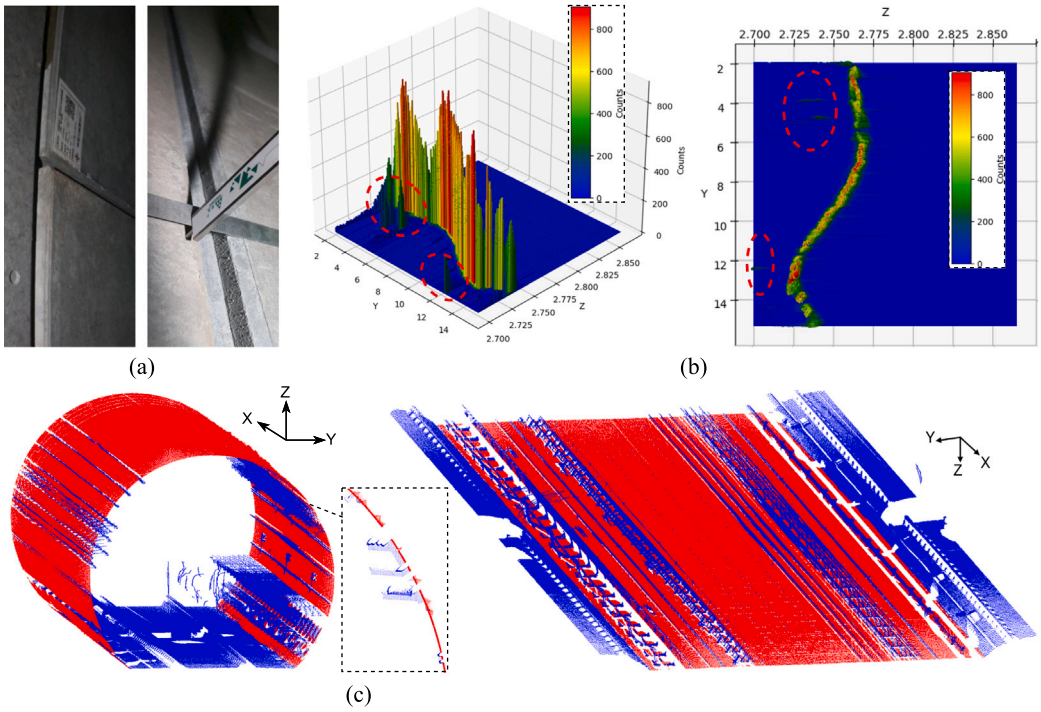


**Fig. 4.** On-site images of segment installation errors and final filtering results: (a) segment installation error; (b) 3D and 2D illustrations of point cloud density variation; (c) visualization results after denoising with non-segment part (blue) and segment part (red). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

variations to select appropriate local segmentation Z-values for filtering non-segment parts. This is because more points accumulate on the segments, while fewer points are on non-segment areas. A density variation statistics for 10 rings is shown in Fig. 4(b) in 2D and 3D. Clearly, the area marked in red in both 2D and 3D perspectives shows that, in the Z direction, it is not directly connected to the region of highest density (which can be inferred as the location of segments) and can be considered a non-segment part. In a local area, this variations can easily determine a Z value for separating segment and non-segment parts. The filtering result is in Fig. 4(c). The details of the algorithm is in Algorithm 2.

### 3.4. Geometry-guided enhanced projection image generation

This section specifically consists of three components. The first one, referring to Algorithm 3-1, involves using simple local height difference rules to extract joint point. The second one, referring to Algorithm 3-2, supplements the sparse areas of the point cloud, particularly the regions between scan lines. The final one, referring to Algorithm 3-3, merges the results from the two aforementioned enhancements to generate the projection panorama, which is then used for image-based segmentation tasks.

Algorithm 3-1 is based on TLS theoretical error to extract and upsample the joint point cloud between segments, which aids in the subsequent generation of prompt points and enhances the SAM's segmentation performance at the edges. Fig. 5(a), illustrates how the thickness $t$ of the point cloud changes with the incident angle $\theta$. Here we assume that the centre of the scanner is located at the centre of the tunnel cross-section, and that the tunnel is straight. The distance error of the scanner is $2\sigma$. Thus, the thickness $t$ can be expressed as follow:

$$t = \sqrt{\frac{(2\sigma)^2 \cdot r}{dist + r}} \tag{1}$$

where, *dist* means the distance referring to the length between the projection of the point on the tunnel being measured onto the tunnel's centre line and the scanner. The variable $r$ represents the tunnel's radius.

As we can see in Fig. 5(b), the point cloud thickness changes along the tunnel centre line when the distance error of the TLS is 8 mm (taken as an example). We simply divide the point cloud into low-density areas and remaining high-density areas. In low-density areas, the local average height can be calculated using a certain number of nearest neighbour points. This average height can be regarded as approximating the central plane. To prevent sampling points from the segment surface, a height difference threshold of at least half the segment thickness is necessary. While for high-density areas, the large number of points makes the average height calculated from a small number of nearest neighbours unstable, making it difficult to accurately represent the central plane's height. Increasing the number of sampling points or using radius-based searches would significantly increase the computational load. Therefore, to account for the worst-case scenario, it is recommended that the height difference threshold be at least greater than the thickness. Additionally, interpolation only happened in the low-density areas. An example result is shown in Fig. 5(c), including extracted points and interpolation points. Here, we display the interpolated points on the left low-density area in red. It is noticeable that the left side has a more defined boundary compared to the right side. Additionally, we use intensity values to represent the extracted points. The vertical metal supports on the segment stand out with significantly high intensity values, appearing as yellow. Overall, it can roughly extract joints and bolts, and enhance low-density joints.

The aim of Algorithm 3-2 is for up-sampling the point cloud on shield segment surface to address the issue of sparse point distribution in distant areas. We design this process based on TLS theoretical density (Huang et al., 2021a), following TLS scanning principles. It is more scientifically robust than directly pixel-based interpolation. TLS scanning occurs in a column-by-column manner, with the point cloud spacing within each column being significantly greater than the spacing between columns. Each column is called a scan line, and the distance between columns is known as the scan spacing, as illustrated in Fig. 6(a). According to the ideal assumptions regarding laser scanning, and assuming that the scanning centre is located on the tunnel axis,

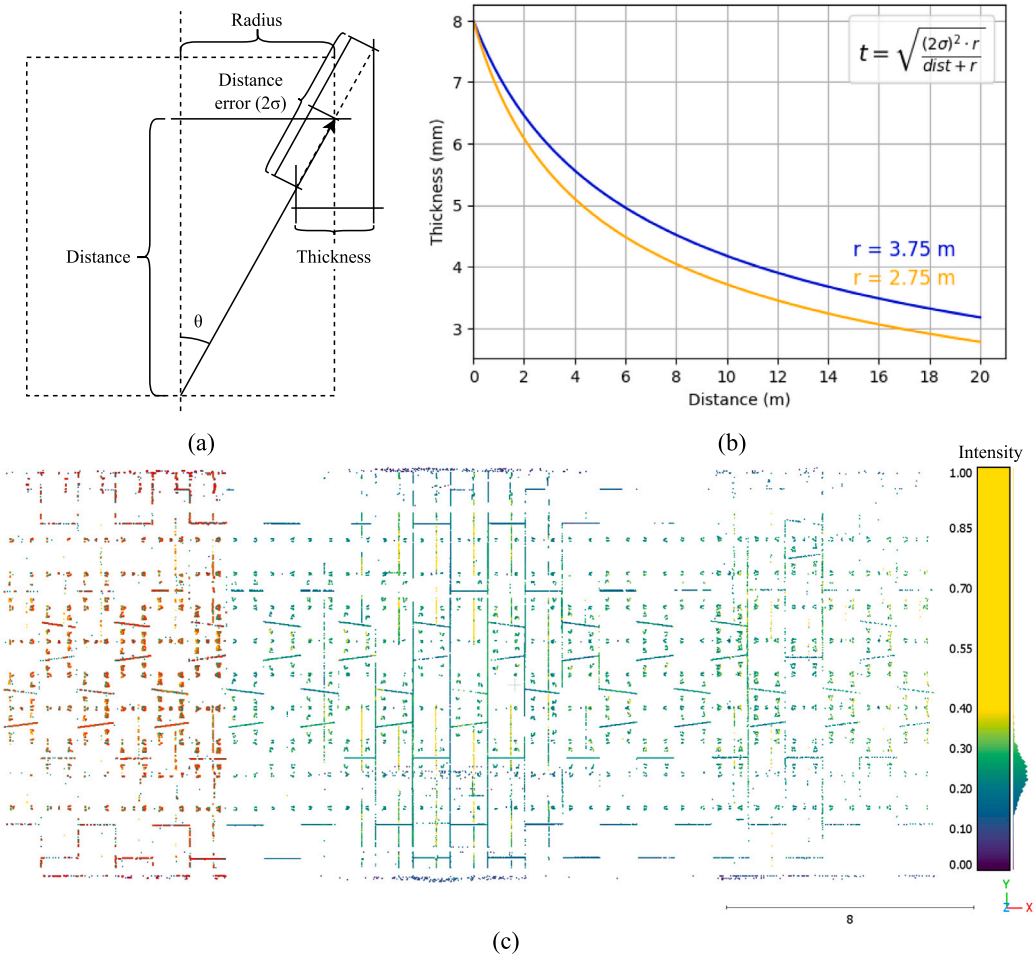


**Fig. 5.** Based on TLS theoretical error to extract and upsample point cloud at joints: (a) diagram of point cloud thickness based on TLS incidence angle; (b) point cloud thickness variation along tunnel centre line distance (error $\sigma$ = 4 mm, and radius(r) = 2.75 m and 3.75 m); (c) results of outlier extraction and enhancement on the left side (extracted points displayed with intensity values, and the interpolation points for enhancement are red). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

the most unfavourable scanning position – where the scan line spacing is maximized – corresponds to the furthest point where the horizontal plane intersecting the tunnel's centre line meets the tunnel, as illustrated in 6(b). The theoretical spacing $S_{\text{line}}$ can be calculated by Huang et al. (2021a), according to pattern observed from scanning the vertical wall:

$$S_{\text{line}} = \frac{d \times \theta}{\cos\left(\arctan\left(\frac{x}{y}\right)\right)} \tag{2}$$

where, the unit of $S_{\text{line}}$ is mm, $d$ represents the distance from the TLS-emitted laser to the most unfavourable point, measured in mm. $\theta$ represents the angular resolution during the TLS scan, in radians. $x$ and $y$ correspond to the coordinates of the most unfavourable point.

The process needs to be repeated in a loop, as shown in Fig. 6(a), with each newly added interpolation also being used along with the original point cloud in the next iteration. The iteration will stop when the search radius in the next step falls below the scan spacing of the theoretical most unfavourable scanning position in the high-density area. We recommend searching within a range of 0.9 to 2 times the specified radius, halving the search radius with each interpolation to gradually fill sparse areas in the point cloud. Subsequent interpolations can also address gaps within the scan lines. Additionally, we set a curvature limit, $Curv_{threshold}$ to ensure interpolation primarily occurs on segment surfaces. This approach helps preserve critical information in areas with significant curvature changes, such as around joints and bolt holes, which are important for segmentation. The final part involves enhanced panoramic images generation. The generation process is detailed in Algorithm 3-3. It involves two steps: first, projecting the result from Algorithm 3-2, and use the mean value for filling due to its lower computational complexity compared to the median, with minimal difference in results. Next, we overlay pixels with result from Algorithm 3-1 to enhance boundaries. Finally, a simple sliding window interpolation fills small gaps while keeping most of the empty pixels.

### 3.5. Template prompt generation

After robust unfolding, denoising and enhancing methods have been developed, the next step is to devise the process of generating prompts. Since our task is to identify the point cloud of tunnel segments at the instance level, according to SAM's input requirements, we need to place the prompt points on the pixels corresponding to the segment surface after projection. Additionally, we must exclude non-segment areas such as bolts and adjacent segments. Therefore, the regions we need should be marked with positive prompts, while others should be marked with negative prompts. Overall, the goal of this section is to automate the generation of prompt points.

The proposed method is outlined in Algorithm 4. The main idea is to extract the central point in the K-block as the initial point, and then use it to generate corresponding prompt points of a complete ring.

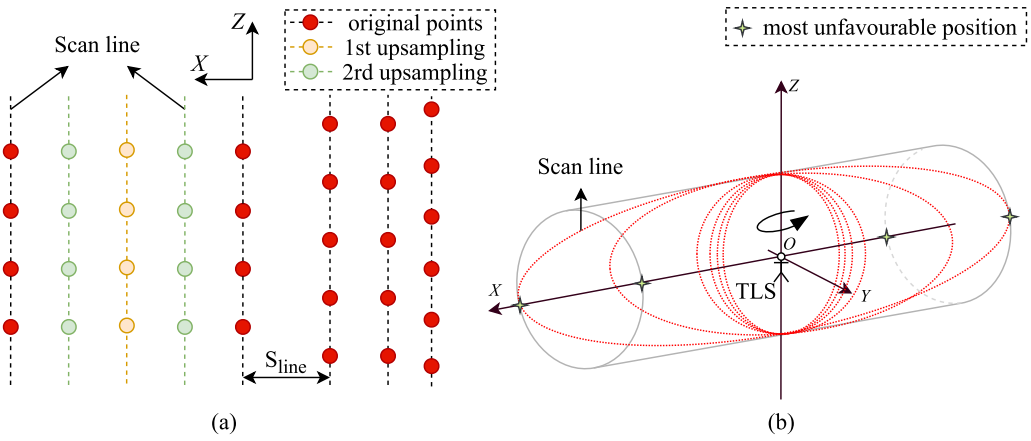


**Fig. 6.** Based on TLS theoretical density to upsample point cloud for segment surface: (a) phase-wise point cloud up-sampling to simulate points onto scan lines; (b) the most unfavourable position in tunnel scanning.

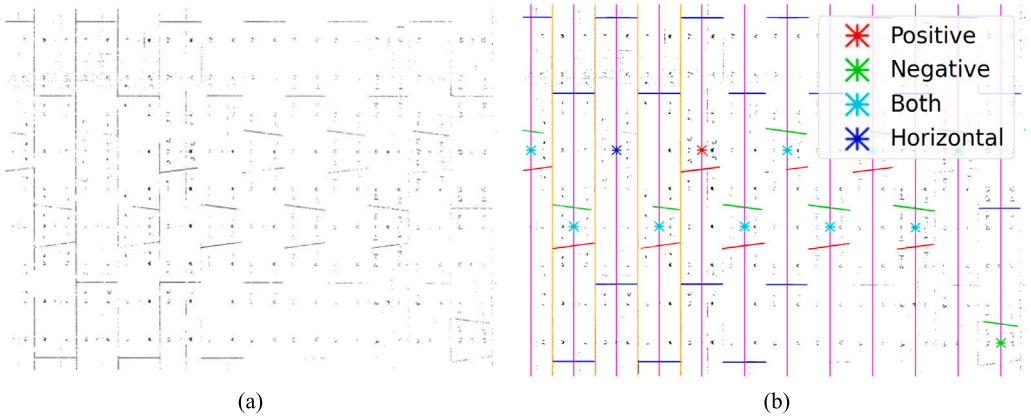


**Fig. 7.** A schematic of enhanced local outlier points and initial prompt point generation: (a) project the enhanced local outlier points onto the 2D image; (2) initial prompt point generation based on Hough transform (Illingworth and Kittler, 1988) detection (orange lines represent the detected vertical joints in the high-density area; magenta lines indicate the centre lines of each ring; green/red segments correspond to detected oblique segments within the rings, representing positive/negative angles, respectively; the star markers with different colours represent the initial points generated by different approaches, referring to Algorithm 4). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

As we can see, Fig. 7(a) visualizes extracted and enhanced points that projected onto 2D image, while Fig. 7(b) shows the initial prompt point generation. We can find out that the detection of vertical joints in high-density areas is highly stable (only interference pixels come from the metal supports, which can be directly filtered by intensity if needed), and the detection of oblique segments with restricted angles shows strong robustness as well. Even if one side is affected by background interference, the other side can still be detected to support prompt generation.

After obtaining the initial point of each ring, the corresponding template prompt points are generated as shown in Fig. 8. Inside the prompt template box, the blue points represent positive prompts (foreground), and the pink points represent negative prompts (background). In addition to prompt points, we have also introduced a coarse mask logits prompt wrapped by polyline segments (Archit et al., 2023), all based on the lining segment design parameters. Overall, the prompt template should be designed according to the specific tunnel's as-designed information, and design principle is to place appropriate positive and negative prompts close to the edges. Meanwhile, some redundancy should be included to account for errors from the initial points. One example of prompt template in Fig. 8 is clearly redundant, while adding more prompt points enhances segmentation stability without affecting inference speed. More details can be found in Section 5.3

### 3.6. LVM-based zero-shot segmentation

The final step we employ zero-shot large vision model, SAM, to complete the final segmentation task, avoiding the need for training. First, since SAM is trained at a resolution of $1024 \times 1024$, all input images are resized to this resolution before performing segmentation. Therefore, before feeding the panoramic image into SAM, it needs to be cropped to ensure that more information is transferred. We input the cropped images into SAM block by block, ring by ring, as shown in Fig. 8. Based on the initial points and their types, the corresponding rectangular portion is cropped from the panoramic image centred around these points. This cropped portion, along with the corresponding template prompt, is then input into SAM for segmentation. The result returned is a binary mask, score, and pixel-level logits.

Additionally, for different tunnel designs, each ring is composed of several blocks. As an example, if the segmented lining ring has six blocks, we denote them as K (key), B1, B2, A1, A2, and A3, considering the scenario where unfolding starts from directly below. When the bottom is minimally obstructed and all labels have been assigned, the two edge blocks will be merged into a single instance. When the bottom is completely unobstructed, the images from one end can be stitched to the other end before being input into SAM for segmentation. Overall, it is necessary to design the prompting process and image cropping and segmentation process based on specific circumstances. Finally, all instance segmentation tasks are completed ring by ring.

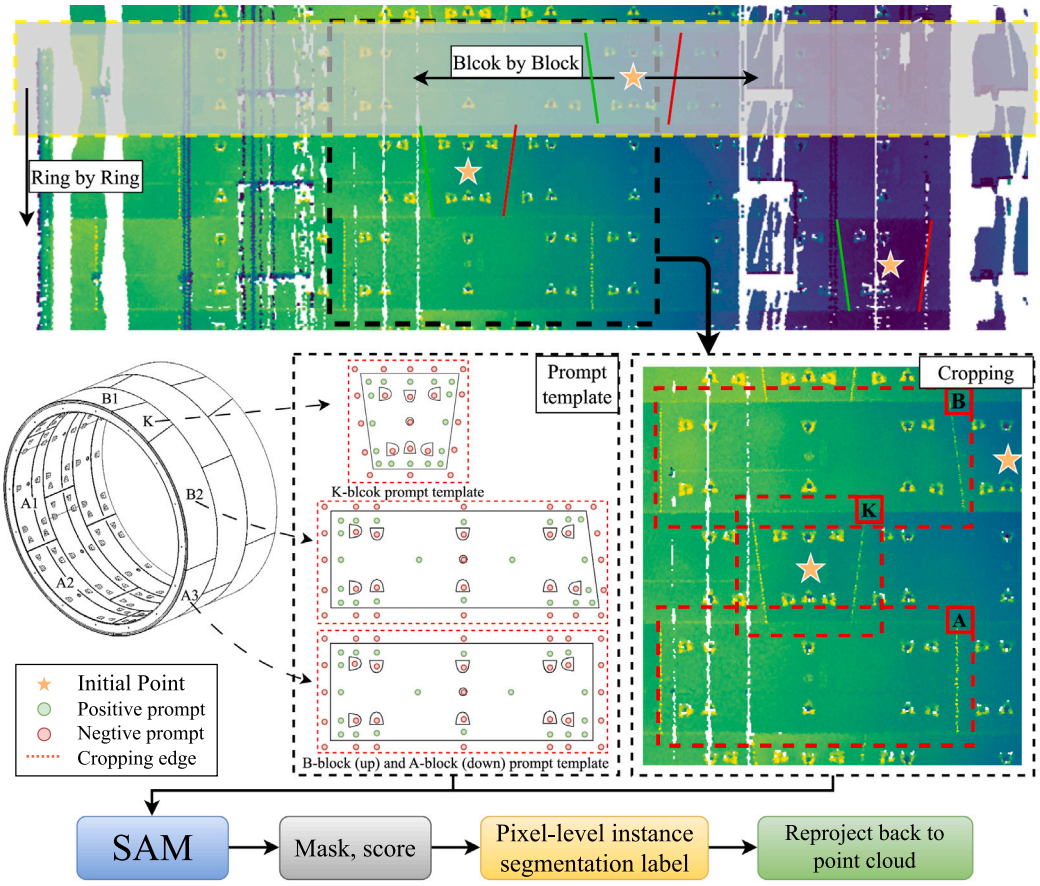


**Fig. 8.** SAM-based TBM point cloud instance segmentation process from a panoramic unfolded image. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

After all segmentation are completed, the pixel-level segmentation results are mapped back onto the panoramic unfolded image. In some cases, overlapping regions caused by the cropping process may result in multiple instance segmentation outcomes. We assign the final label to the result with the highest confidence (predicted mask logits from SAM). Ultimately, after all pixels have obtained segmentation results, the results are back-projected onto the point cloud, assigning the corresponding labels to the point cloud. The labels of non-segmented point clouds, which were filtered out earlier, remain unchanged.

## 4. Model implementation

### 4.1. Data description

In this section, we demonstrate our proposed workflow using the publicly available TBM segmental lining dataset, Seg2Tunnel (Lin et al., 2024) and STSD (Cui et al., 2024). To enable direct comparison with deep learning benchmarks, we use the T1&T2 tunnel test sets from Seg2Tunnel. Additionally, T3, T4, and T5 tunnels from Seg2Tunnel represent different scenes. T3 tunnel point cloud was generated through multi-station scans and registration, resulting in a relatively uniform point cloud distribution. While T4&T5 were scanned using single stations, involving a larger radius with more segments per ring. The Seg2Tunnel was collected using TLS, while we also introduce the STSD, acquired through MLS, considering that MLS is likely to become a more mainstream method for rail-based environments. We randomly selected data from the circular tunnel case of STSD. For aligning with our task, we annotated the tunnel segments based on the original dataset's annotations. They will be discussed in Section 5.2.

For T1&T2 cases, these tunnels have a designed inner diameter of 5.5 m, with each ring consisting of 6 segments and a width of 1.2 m. The TLS data were collected using the Leica C10 scanner, and the format comprises single-station scanner data, with each station containing approximately 14 million points. The difference between the densest ring and the sparsest ring is approximately 100 times. The degree resolution during scanning is 0.0573 degree and the standard deviation of the instrument is 4 mm. We perform instance segmentation on the segmented tunnel linings, with different types of segments treated as individual instances, while the remaining parts are classified as background. The details are shown in Table 1.

### 4.2. Evaluation method

The evaluation method for point cloud instance segmentation follows the widely recognized COCO (Microsoft Common Objects in Context) official evaluation criteria (Lin et al., 2014). Although COCO evaluation is designed for pixels in 2D images, it can also be used for point cloud instance segmentation evaluation by adapting its evaluation metrics to points in a point cloud. A prediction is considered accurate if the IoU (Intersection Over Union) between the predicted instance's result and the ground truth exceeds a certain threshold $T$, and the predicted category matches the ground truth category. We offer $mAP$ and $mAP@50$ as primary evaluation metrics. The Mean Average Precision ($mAP$) is mean value of Average Precision ($AP$) without background classes, which computed across various IoU values, ranging from 0.50 to 0.95 with increments of .05. While $mAP@50$ is calculated at a single IoU of 0.50. AP is calculated via recall and precision:

$$Recall_i = \frac{TP_i}{TP_i + FN_i} \tag{3}$$

$$Precision_i = \frac{TP_i}{TP_i + FP_i} \tag{4}$$

**Algorithm 1:** Conversion of $Cloud_{TBM}$ to Cylindrical Coordinates

**Input:** TBM tunnel point cloud $Cloud_{TBM}$
**Output:** $Cloud_{TBMwithCC}$
**begin**
  **1. Determine direction vector**: Project $Cloud_{TBM}$ onto a horizontal plane, then find the minimum enclosing rectangle of the convex hull to get the direction vector $\vec{V}_d$, pointing from the midpoint of one short side to the other.
  **2. Generate point cloud slices**: Along $\vec{V}_d$, setting point cloud slicing thickness $\delta$ (default to 0.01) and adjacent mid-planes distance $l$ (default to the ring width, with the first mid-plane positioned half a ring from the starting point) to generate a set of 2D point cloud $Cloud_{Slices}$ (Salmi et al., 2025).
  **3. Ellipse centre fitting of $Cloud_{Slices}$**:
  Create empty list of ellipse centre points $Points_{Centre}$;
  **for** *each slice* $s_i$ *in* $Cloud_{Slices}$ **do**
    // Rough filtering of railway point cloud
    Create empty list $Slice_{Filtered}$;
    **for** *each point* $p_i$ *in* $s_i$ **do**
      **if** $|p_i[w] - w_{max}| \leq W\text{-}value$ **then**
        Add $p_i$ to $Slice_{Filtered}$;
      **end**
    **end**
    Use RANSAC algorithm to fit an ellipse based on $Slice_{Filtered}$, and then roughly filter out the noise to get $Slice_{SecondFiltered}$. Fit a new ellipse using RANSAC algorithm based on $Slice_{SecondFiltered}$, and obtain centre point;
    Add centre point to $Points_{Centre}$;
  **end**
  Return $Points_{Centre}$;
  **4. Cylindrical coordinate system conversion**: Use $Points_{Centre}$ and RANSAC algorithm to fit the parametric equation of $Curve_{centre}$, and then convert each points in $Cloud_{TBM}$ to cylindrical coordinates;
  **5. Return**: $Cloud_{TBMwithCC}$
**end**

**Algorithm 2:** Non-segment part filter based on local point cloud density differences

**Input:** $Cloud_{TBMwithCC}$
**Output:** $Cloud_{TBMwithCC}$ with non-segment parts
**begin**
  **1. Initial filtering**: Perform a coarse filtering by setting some radial length (Z value in the unfolded point cloud), closing to tunnel's inner radius, as a threshold;
  **2. Create grids and count points**: Define grid sizes along X (default to ring width), Y (default to 0.5m), and Z (default to 1mm) directions for remaining point cloud. Calculate the number of points in each grid;
  **3. Determine $Z_{cutoff}$ values**:
  Create empty matrix $Z_{cutoff}$;
  **for** *each X bin* **do**
    **for** *each Y bin* **do**
      Start from the grid with the highest point density in Z bin, and move in the direction of decreasing Z values;
      **if** *the gradient stops decreasing or starts increasing* **then**
        Add this Z value to $Z_{cutoff}$, which representing edge of the tunnel segment;
      **end**
      Interpolate and smooth the $Z_{cutoff}$ values in Y direction to handle gaps and inconsistencies, and reducing by an value (default to 3 mm, could be larger) for laser scanning or construction errors;
    **end**
  **end**
  Return smoothed $Z_{cutoff}$;
  **4. Filter out non-segment parts**: Apply the smoothed $Z_{cutoff}$ values to filter out non-segment parts for each grid;
  **5. Return**: $Cloud_{TBMwithCC}$ with non-segment parts
**end**

$$AP = \frac{1}{101} \times \sum_{r \in 0, 0.01, \ldots, 1} p_{interpolation}(r) \tag{5}$$

where $TP_i$, $FN_i$ and $FP_i$ denote the true positive, false negative and false positive instances of lining segment $i$, and $p_{\text{interpolation}}(r)$ is the precision obtained through interpolation at the given maximum recall level $r$. The precision is averaged over the set of 101 equally spaced recall levels $[0.0, 0.01, 0.02, \ldots, 1.0]$. Here, $i = 1, 2, 3, \ldots, n$, where $n$ is the total number of segment for each ring.

In addition, we also provide an evaluation for semantic segmentation, including F1 Score, Overall Accuracy (OA) and mean IoU (mIoU), which takes the background class into account. In contrast to instance segmentation, now $i$ start from 0 to $n$ with 0 means background, and $TP$, $FP$, $FN$ here for points:

$$\text{F1 Score} = 2 \times \frac{Precision_i \times Recall_i}{Precision_i + Recall_i} \tag{6}$$

$$OA = \frac{TP_{total}}{TP_{total} + FP_{total} + FN_{total}} \tag{7}$$

$$mIoU = \frac{1}{i} \sum_{i=0}^{n} \frac{TP_i}{TP_i + FP_i + FN_i} \tag{8}$$

### 4.3. Parameter settings

In this study, we first needed to determine the projection resolution and evaluated three options: 10 mm, 5 mm, and 2.5 mm. We based our evaluation on the statistical results from Fig. 9(b) and (c). At 10 mm, the resolution seems only a small amount of interpolation is needed. However, the input of SAM is 1024 × 1024 resolution, and for A/B-block lengths around 3500 mm (including extensions for segmentation), the resulting of 10 mm resolution with 350 pixels losing too many details. In contrast, the 2.5 mm resolution generates around 1400 pixels, closely matching SAM's input, but results in many empty pixels, especially in distant regions, which also reduces computational efficiency due to the large projection map size. As a compromise, we selected the 5 mm resolution, which provides around 700 pixels to represent the A/B-block longitudinally. Even in distant regions, around one original point can be maintained within each 10 × 10 grid, resulting in a final depth map of 3454× ~ 6000 pixels. This strikes a good balance between computational efficiency and acceptable accuracy.

Based on the 5 mm resolution, we generated a panorama without interpolation (see Fig. 9(a)). From the visualization results, the 11 rings closest to the scanner are clearly identifiable and do not require little interpolation. Therefore, we selected this region as the high-density area. The rest of the area is considered as the low-density area. The maximum point cloud thickness in high-density and low-density area is around 8 and 4.5 mm (see Fig. 5(b)). Therefore, the threshold for outlier selection we set is 10 mm (at least larger than 8 mm) in the high-density area and 3 mm (at least larger than half of 4.5 mm) in the

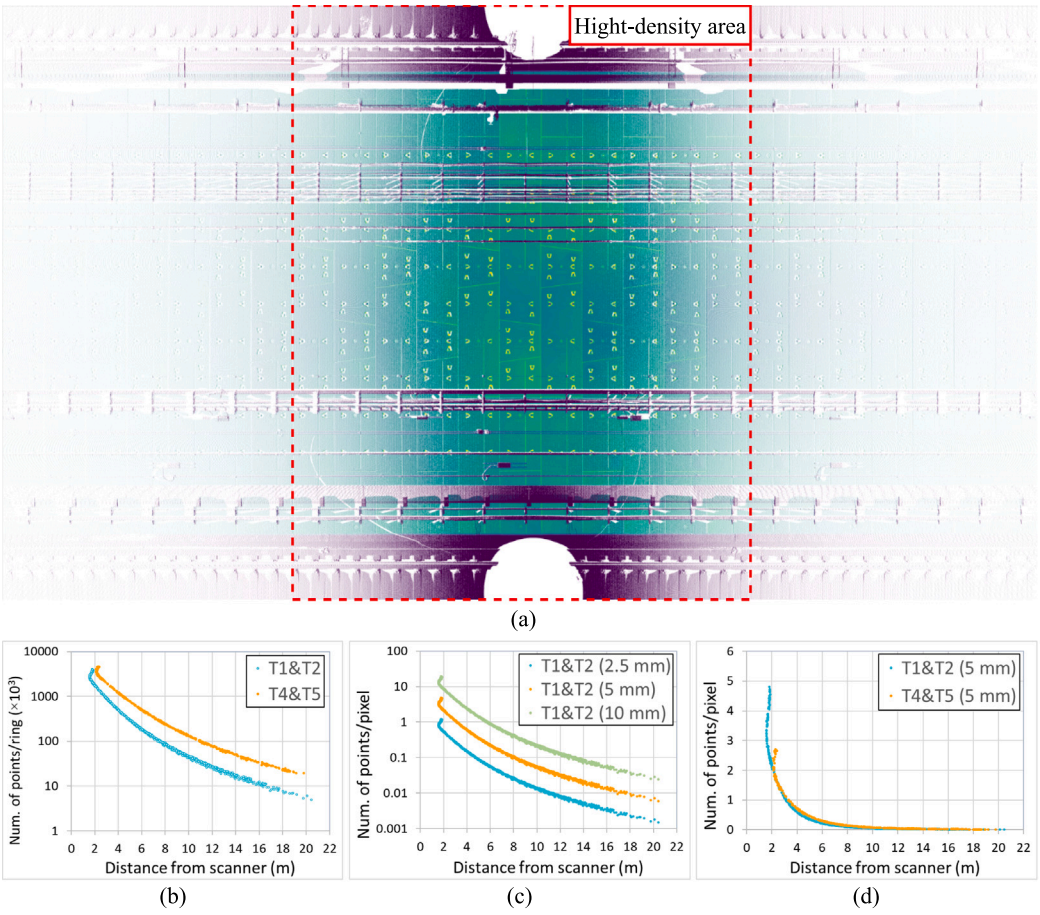


**Fig. 9.** Distinguishing point cloud high-density areas and low-density areas: (a) unwrapped projection depth map without interpolation; (b) number of points per ring vs. distance from scanner; (c) number of points per pixel vs. distance from scanner (different resolution); (d) number of points per pixel vs. distance from scanner (when resolution is 5 mm).

**Algorithm 3-1:** Upsampling at the joints of shield segment

**Input:** $Cloud_{TBMwithCC}$ with non-segment parts
**Output:** $points_{EnhancedJoint}$

**begin**
  **1. Outlier point extraction**:
  Calculate the difference in height between the average height of the nearest neighbour points (default $k = 20$) and the selected point. Points with a difference greater than the threshold are selected. These points, named $points_{peak}$, include the majority of the points at the joint ;
  **2. Outlier point enhancement**:
  Initialize $points_{EnhancedJoint}$ as an empty set ;
  **for** *each point in* $points_{peak}$ *in low density area* **do**
    **for** *each pair of points (p1, p2) in* $points_{peak}$ **do**
      **if** *distance(p1, p2)* $\leq$ *radius* **then**
        // Insert points between two points that meet the criteria ;
        $insert\_points \leftarrow \text{Insert}(p1, p2, radius)$ ;
        **if** *insert_points is not too close to* $points_{peak}$ **then**
          Add *insert_points* to $points_{EnhancedJoint}$ ;
        **end**
      **end**
    **end**
  **end**
  Return $points_{EnhancedJoint}$ ;
  **3. Return**: $points_{peak}$, $points_{EnhancedJoint}$ ;
**end**

**Algorithm 3-2:** Upsampling for shield segment surface

**Input:** $Cloud_{TBMwithCC}$ with non-segment parts
**Output:** $points_{EnhancedSegment}$

**begin**
  **1. Curvature calculation**: Calculate the curvature of $Cloud_{TBMwithCC}$ using the original Cartesian coordinate system ;
  **2. Segment surface enhancement**: ;
  Initialize $points_{EnhancedSegment}$ as an empty set ;
  **for** *each pair of points (p1, p2) in* $Cloud_{TBMwithCC}$ **do**
    **if** $0.9 \times radius \leq distance(p1, p2) \leq 2 \times radius$ ***and*** $Curv_{diff} \leq Curv_{threshold}$ *(default = 0.0005)* **then**
      // Insert one points between two points ;
      $insert\_points \leftarrow \text{Insert}(p1, p2, radius, Curv_{diff})$ ;
      **if** *insert_points is not too close to* $Cloud_{TBMwithCC}$ ***and*** *insert_points* **then**
        Add *insert_points* to $points_{EnhancedSegment}$ ;
      **end**
    **end**
  **end**
  **3. Return** $points_{EnhancedSegment}$ ;
**end**

low-density area. During data collection, to ensure uniform scanning, the scanner's centre is positioned as close as possible to the centre of the tunnel cross-section. Then we use Eq. (2) to calculate the most

**Algorithm 3-3:** Enhanced projection panoramic image generation

**Input:** $Cloud_{TBMwithCC}$, $points_{EnhancedSegment}$, $points_{EnhancedJoint}$, $points_{Peak}$
**Output:** Enhanced tunnel projection panoramic image
**begin**
- **1. Canvas Creation**: Set the resolution (real size of each pixel) and create a canvas grid based on $Cloud_{TBMwithCC}$;
- **2. Project Points**: Project $Cloud_{TBMwithCC}$ (excluding the segment part) and $points_{EnhancedSegment}$ onto the canvas. Use the mean value of corresponding attributes for pixels that overlap;
- **3. Project Joint Points**: Project $points_{EnhancedJoint}$ and $points_{Peak}$; replace existing pixel values if they are present;
- **4. Pixel Interpolation**: Apply a sliding window (default is 9x9) to detect null pixels. If a null pixel is found and there are other pixels within the window, fill the null pixel with the nearest neighbour value; otherwise, leave it empty;
- **5. Return**: Enhanced projection panoramic image

**end**

**Algorithm 4:** Template prompt point generation

**Input:** $points_{EnhancedJoint}$
**Output:** template prompt for each rings
**begin**
- **1. Project onto 2D image**: project only $points_{EnhancedJoint}$ onto a 2D image, by using Algorithm 3-3;
- **2. Joints detection**: Perform two types of Hough transform detection (Illingworth and Kittler, 1988): *vertical lines* (restricted to high-density areas), which indicate joints between rings; and *oblique line segments*, which identify joints between K-block and B-block;
- **3. Intersection point detection**: Use detected vertical lines (Yi et al., 2019) to generate ring's centre line for each pair of adjacent lines. Use average distance between these adjacent centre lines to generate equidistant centre lines towards the low-density point cloud regions. Obtain intersection points between vertical lines and oblique line segments;
- **4. Initial prompt points generation**: X-coordinates of initial prompt points are the same with X-coordinates of intersection points. Y-coordinate needs to be adjusted:
  - For intersection with one oblique line segment, if the inclination angle is positive/negative, add/subtract half of the K-block mid-height to the original y-coordinate;
  - For intersection with two oblique line segment, take the y-coordinate of the midpoint between the two points;
  - For no intersection with oblique line segment (which are rare), if the distance of any two horizontal line segment is close to the mid-height of the K-block plus twice the mid-height of the B-block, take the y-coordinate of the midpoint between the two points;
- **5. Template prompt**: Generate template prompt based on the initial points, by following Fig. 8;
- **6. Return**: Template prompt for each rings

**end**

unfavourable position for each station. Based on this result, we perform up-sampling interpolation. For example, in low-density areas, the scan spacing in the most unfavourable position is 80 mm, while in high-density areas, it is 19 mm. Therefore, a total of three up-sampling steps need to be performed, corresponding to search radii of 80 mm, 40 mm, and 20 mm, respectively. As with the majority of studies, we chose depth as the colour filled in projection panoramic image in this paper. Considering that we used cropped sub-images for the SAM segmentation tasks, that means we primarily focused on local height, which significantly reduces the impact of axis fitting errors. The remaining parameters were all set to default values, and we summarized them in Table 2.

**Table 1**
Details of the scanned T1&T2 tunnel point cloud test set for case study.

| Tunnel | Ring count | Number of points | Length |
|---|---|---|---|
| T1-4 | 28 | 14,041,185 | 33.6 m |
| T1-12 | 25 | 14,098,396 | 30 m |
| T1-15 | 27 | 14,298,917 | 32.4 m |
| T2-2 | 25 | 14,372,841 | 30 m |
| T2-10 | 18 | 14,291,937 | 21.6 m |
| T2-14 | 23 | 14,455,943 | 27.6 m |

Examples of specific prompts for different segments are illustrated in Fig. 10. Since lower semicircular section of the tunnel are sealed, the corresponding negative points were automatically removed during the process. Prompt points beyond the cropped image boundaries were also automatically removed. For practical cases, we set more safety measures to ensure the generation of initial points for each ring. To avoid interference during the detection of vertical joints, we filtered out the vertical metal brackets based on intensity values. In addition to extracting initial points based on the detection of single or paired oblique segments, we also added the detection of horizontal segments for some rare cases where both oblique segments are both undetectable. If the detection of horizontal segments still fails, considering that the K-blocks in this case are mostly arranged in a simple staggered pattern with only a small portion arranged irregularly, the y-value of the initial point will be estimated by retrieving the y-values of the normally ordered K-block detected earlier.

## 5. Validation and model performance

### 5.1. Validation

We selected T2-14 to display the results in Fig. 11, including the unfolded depth map, pixel-level instance segmentation map, and the re-projected predicted segmentation point cloud. Overall, the visualization closely aligns with the ground truth, with minor differences observed around the bolt holes. Seg2Tunnel (Lin et al., 2024) provides benchmarks for both semantic segmentation and instance segmentation tasks for the entire tunnel scene. Among them, RandLA-Net (Hu et al., 2022) was selected as semantic segmentation benchmark. For instance segmentation we selected, Mask3D (Schult et al., 2023) and TD3D (Kolodiazhnyi et al., 2024) as end-to-end benchmark, and the best benchmark is to first use RandLA-Net (Hu et al., 2022) for semantic segmentation, and then perform instance segmentation for each category based on these results. We conducted comparisons for both, semantic and instance segmentation, as shown in Table 3, and Table 4. Table 5 shows the specific results of each categories. The label order follows the lining forward direction, starting clockwise with the K-block as label 1, and then proceeding sequentially. Label 0 means background.

In the end, we found that our method outperformed supervised learning benchmarks in both semantic and instance segmentation tasks. For semantic segmentation, the *OA* improved from 93.2 to 95.6, a gain of 2.4, while the *mIoU* increased from 83.1 to 91.8, a gain of 8.7. For instance segmentation, the *mAP* rose from 73.3 to 80.9, an improvement of 7.6.

**Table 2**
Default values of parameters during processing.

| Algorithm | Parameter | Value | Description |
|---|---|---|---|
| A1 | Slicing thickness | 0.01 m | Point cloud slices |
| | Intervals of mid-planes | Ring width | |
| | Inlier ratio | 0.75 | For RANSAC |
| | Axis intervals | 1 mm | Along central axis |
| | Coarse filtering | <2.7 m, >2.9 m | Radius is 2.75 m |
| A2 | Grid size | Smooth size = 3, reduction value = 3 mm; X = ring width, Y = 0.5 m, Z = 0.001 m | Smaller grids require more computation but filter more finely |
| A3 | Number of nearest neighbour points | 20 | |
| | Searching range of interpolation points | 0.06 m | Decided by the distance from bolt to edge |
| | Number of interpolation points | 2 | Enhancing joints |
| | Threshold of too close to existing points | 20 mm | Decided by the most unfavourable scan spacing in high-density area |
| | Curvature threshold | 0.0005 | Decide whether it is surface of segment |
| | Sliding window size | 9 × 9 | For depth map generation |
| A4 | Hough segment detection | Minimum line length is 100, maximum line gap for oblique is 50, for horizontal is 20 | Recommended value, can be adjusted slightly |
| | Hough line detection | Minimum line length is 1000 | |

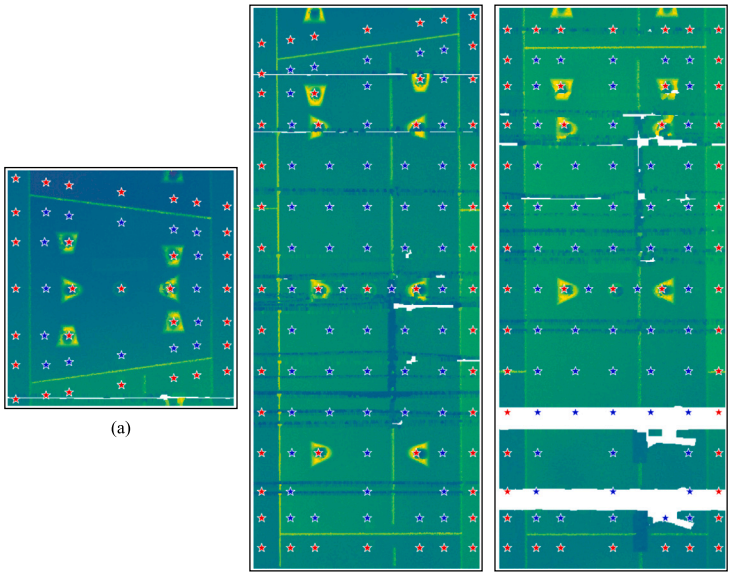


**Fig. 10.** Examples of specific prompts (blue are positive prompt points, red are negative prompt points) for different segments for T1 and T2: (a) K-block; (b) B-block; (c) A-block. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

**Table 3**
Semantic segmentation results compared with benchmarks (Lin et al., 2024).

| Model | *OA* | *mIoU* |
|---|---|---|
| RandLA-Net (H4) | 0.931 | 0.831 |
| RandLA-Net (H5) | 0.932 | 0.830 |
| SAM4Tun | **0.956** | **0.918** |

**Table 4**
Instance segmentation results compared with benchmarks (Lin et al., 2024).

| Model | *mAP* | *mAP@50* | *mAP@75* | *mAP@90* |
|---|---|---|---|---|
| Mask3D | 0.019 | 0.069 | – | – |
| TD3D | 0.270 | 0.336 | – | – |
| RandLA-Net+TD3D | 0.733 | 0.961 | 0.876 | 0.277 |
| SAM4Tun | **0.809** | **0.998** | **0.957** | **0.460** |

## 5.2. Generalization capabilities of proposed method

### 5.2.1. Multi-station registration case

Case study T3 from Seg2Tunnel (Lin et al., 2024) is composed of four independent scanning stations registered together, covering a total length of 119 rings, with each ring having a width of 1.2 m and an inner diameter of 5.9 m. The arrangement of the K-blocks is continuous along the direction of tunnel advancement (see Fig. 12(c)). The scanning was performed using a Leica C40. Since this is a multi-station combination, the point cloud density distribution is relatively uniform. The rings at the end of the scan are covered by the scanning areas of two stations, resulting in density variation for individual rings is less than 10 times. Therefore, no up-sampling was performed to enhance the data. A global outlier detection threshold of 0.01 m was set. The same detection logic as used for T1&T2 was applied to generate initial points. However, if the detection of horizontal segments also fails, the y-value of the previous detected point is directly inherited due to continuous arrangement of segment. We selected the first fifty rings for testing, covering a total length of 60 m and including the scanning range of two stations with around 20,500,000 points. The template prompts used were similar to that of T1&T2, but it has been customized for the specific dimensions of the segments in the T3 tunnel and the locations of the bolt holes. We present here the unfolded point cloud

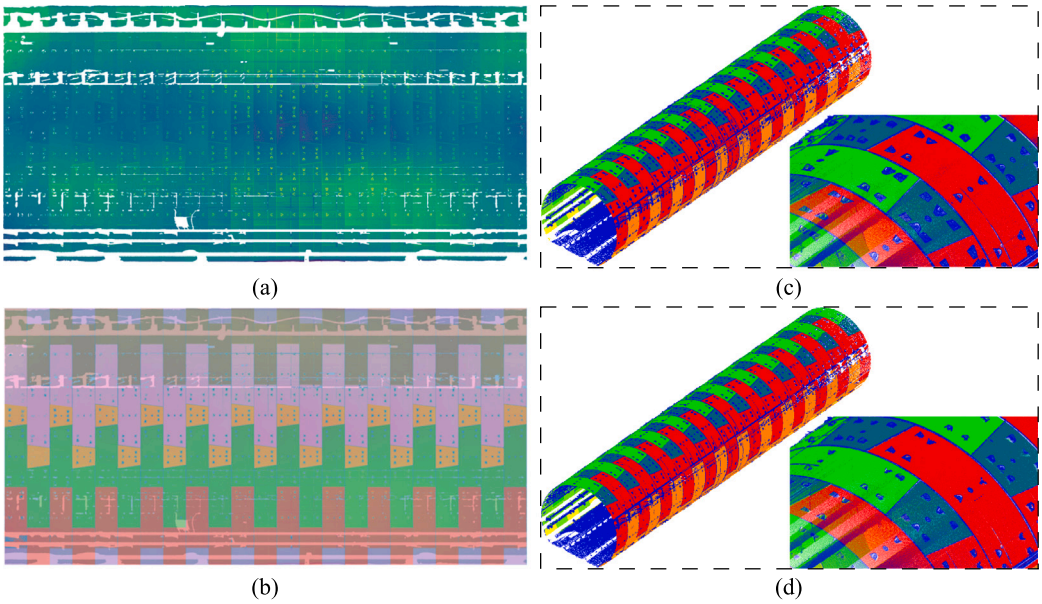


**Fig. 11.** The segmentation results from T2-14 station: (a) unfolded depth map; (b) pixel-level instance segmentation prediction results; (c) ground truth; (d) SAM4Tun.

**Table 5**
Semantic and instance segmentation results of T1 and T2.

| Semantic segmentation | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| *OA* | *F1* | *mIoU* | *IoU* | | | | | | |
| | | | Label 0 | Label 1 | Label 2 | Label 3 | Label 4 | Label 5 | Label 6 |
| 0.956 | 0.943 | 0.918 | 0.867 | 0.936 | 0.948 | 0.935 | 0.876 | 0.924 | 0.942 |
| Instance segmentation | | | | | | | | | |
| *mAP@50* | *mAP@75* | *mAP@90* | *mAP* | *AP* | | | | | |
| | | | | Label 1 | Label 2 | Label 3 | Label 4 | Label 5 | Label 6 |
| 0.998 | 0.957 | 0.460 | 0.809 | 0.793 | 0.865 | 0.840 | 0.679 | 0.823 | 0.898 |

**Table 6**
Semantic and instance segmentation results of T3.

| Semantic segmentation | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| *OA* | *F1* | *mIoU* | *IoU* | | | | | | |
| | | | Label 0 | Label 1 | Label 2 | Label 3 | Label 4 | Label 5 | Label 6 |
| 0.947 | 0.931 | 0.906 | 0.842 | 0.861 | 0.934 | 0.927 | 0.935 | 0.909 | 0.933 |
| Instance segmentation | | | | | | | | | |
| *mAP@50* | *mAP@75* | *mAP@90* | *mAP* | *AP* | | | | | |
| | | | | Label 1 | Label 2 | Label 3 | Label 4 | Label 5 | Label 6 |
| 1.000 | 0.968 | 0.580 | 0.837 | 0.688 | 0.899 | 0.861 | 0.871 | 0.803 | 0.851 |

ground truth compared to the predicted point cloud, shown in Fig. 12. The scores for semantic and instance segmentation are presented in Table 6.

### 5.2.2. Different segment arrangement case

The T4&T5 tunnels from Seg2Tunnel (Lin et al., 2024) consist of 7 segments forming one ring, with each ring having a width of 1.8 m, and the inner diameter of the tunnel is 7.5 m. The arrangement of the K-blocks is a highly complex, interleaved pattern. The scanning was performed using a Leica C10, and the tunnel was under construction at the time of scanning. Theoretically, although the scanner centre in T4&T5 deviate significantly from the axis, the overall scanning distance and the number of points per pixel still can be similar to the conditions in T1&T2. As shown in the statistical results in Fig. 9(c), the differences are within an acceptable range. Therefore, referencing the range of high-density area in T1&T2, we determined the high-density area in this cases to be the 7 rings closest to the scanner. The rest is considered as the low-density area. Since the scanning centre is offset from the tunnel cross-section centre, the most unfavourable position has shifted, making it impossible to use Eq. (2) (Huang et al., 2021a) to calculate it. In this case, we obtained the most unfavourable position scan spacing by directly measuring it from the tunnel point cloud. The outlier threshold for the high-density area is set to 10 mm, while for the low-density area is 4 mm, according to Eq. (1) and Fig. 5(b). The remaining parameter settings are consistent with those of T1&T2 case. We selected stations T4-6 for testing, and the results are shown in Table 7. The segmentation performance is shown in Fig. 13, and visually, the results are highly similar to ground truth. We further present the ground truth, intensity, depth, and predicted unfolded point cloud in detail. It is observed that there are some inevitable errors (over-segmenting of bolt holes) in the ground truth due to manual annotation. While the bolt holes are recognized with higher accuracy using SAM4Tun, facilitated by depth maps.

### 5.2.3. Mobile laser scanning case

We randomly selected 13 consecutive rings from the circular tunnels in the STSD dataset (Cui et al., 2024), with a total length around 20 m.

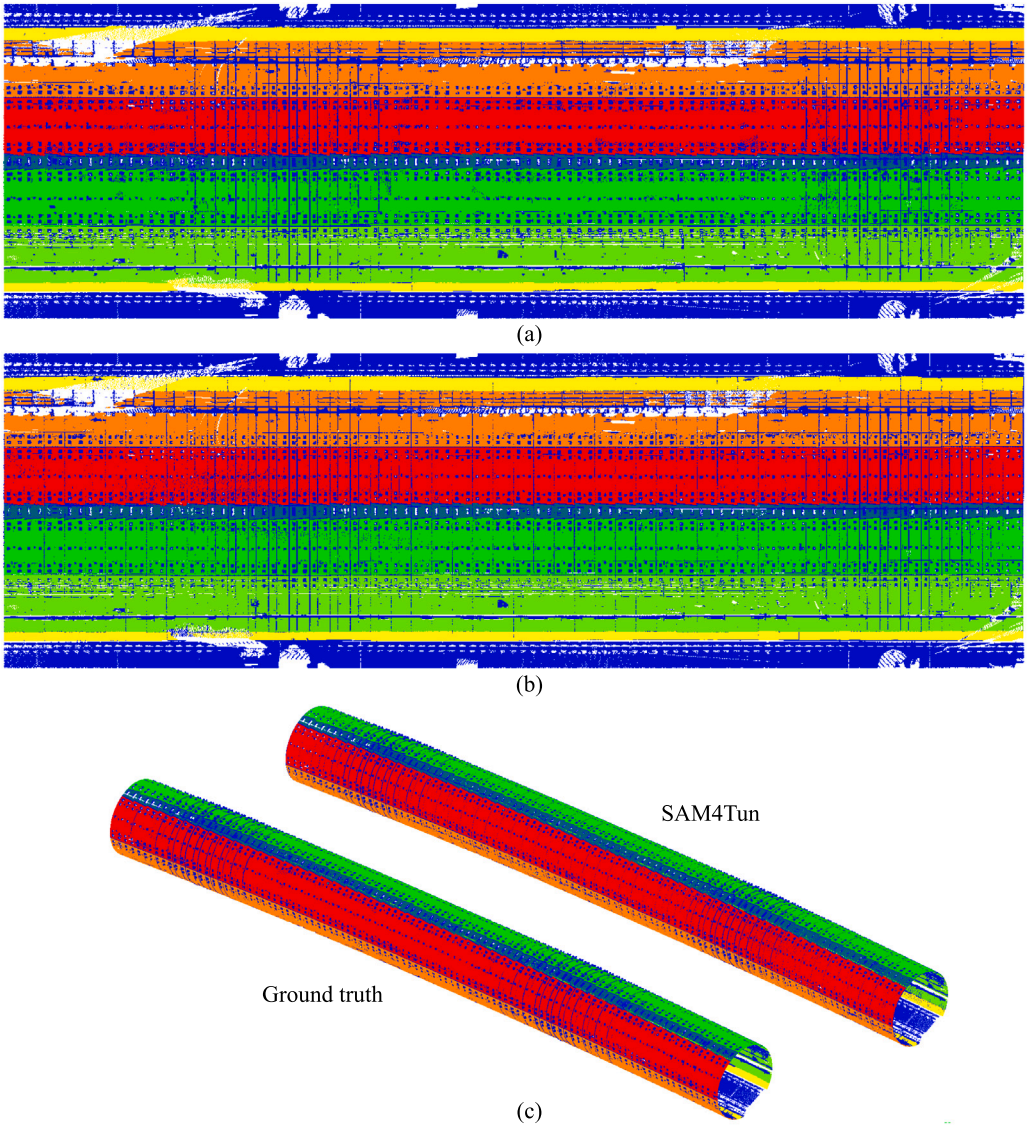


**Fig. 12.** The comparison between the ground truth and the predicted results of T3: (a) unfolded ground truth; (b) unfolded SAM4Tun; (c) in the tunnel morphology.

**Table 7**
Semantic and instance segmentation results of T4-6.

| Semantic segmentation | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| *OA* | *F1* | *mIoU* | *IoU* | | | | | | | |
| | | | Label 0 | Label 1 | Label 2 | Label 3 | Label 4 | Label 5 | Label 6 | Label 7 |
| 0.951 | 0.946 | 0.918 | 0.867 | 0.917 | 0.929 | 0.946 | 0.900 | 0.939 | 0.950 | 0.899 |
| Instance segmentation | | | | | | | | | | |
| *mAP@50* | *mAP@75* | *mAP@90* | *mAP* | *AP* | | | | | | |
| | | | | Label 1 | Label 2 | Label 3 | Label 4 | Label 5 | Label 6 | Label 7 |
| 1.000 | 0.941 | 0.461 | 0.800 | 0.722 | 0.849 | 0.840 | 0.759 | 0.845 | 0.833 | 0.752 |

Each ring has a width of 1.5 m and a diameter of 5.4 m, and the arrangement of the K-blocks is not regular. The original dataset primarily annotated various tunnel accessories, such as pipe, rail, tubes, joints and bolts, but did not include annotations for the tunnel segments. To align with our task, we adjusted the annotations by converting all the original annotations to Label 0 (background), and then performed instance-level annotations on the tunnel segment point clouds. The data for this case comes from MLS, so similar to the T3 case, the point clouds are uniform, and no upsampling was applied during the process. The results are shown in Fig. 14 and Table 8. Similar to the case of T4&T5, the segmentation results of SAM4Tun closely align with the outcomes reflected in the depth maps and demonstrate highly consistent with the ground truth, achieving a mAP of 82.4%. The primary discrepancies are concentrated on the joints, which is attributed to the fact that joints are one of the classification targets in the original dataset. It can be observed that their annotations slightly extend beyond the actual boundaries. In addition, we observed some points with label 0 (displayed in blue) near the segments, which were verified to be noise points collected during the point cloud acquisition process while in motion. Since we used the outlier point extraction method in Algorithm 3-1 to extract joints, these scattered noise points were also extracted and directly assigned a label as background, contributing to obtaining

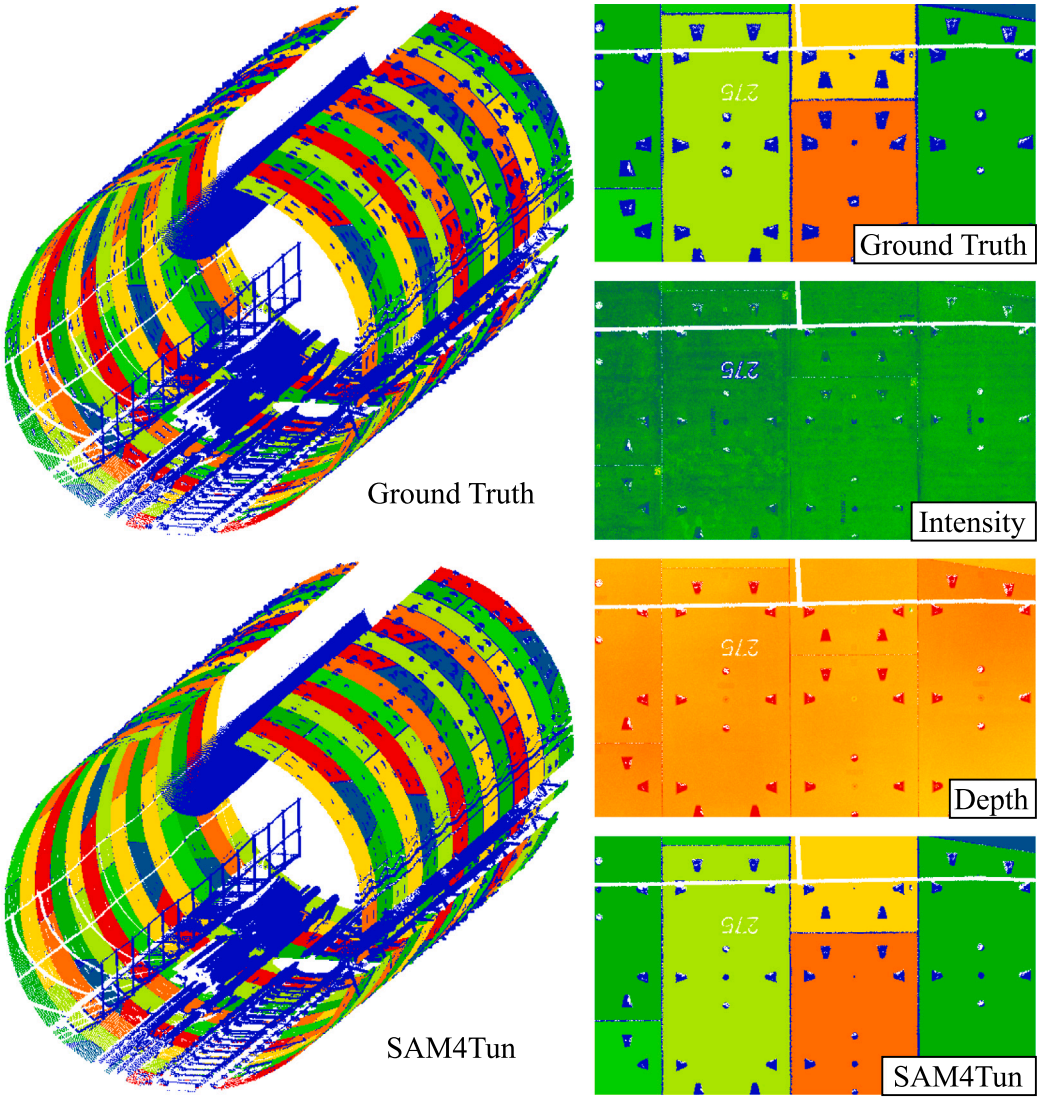


**Fig. 13.** The ground truth and predicted results of T4-6 station.

**Table 8**
Semantic and instance segmentation results of STSD circular tunnels.

| Semantic segmentation | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| *OA* | *F1* | *mIoU* | *IoU* | | | | | | |
| | | | Label 0 | Label 1 | Label 2 | Label 3 | Label 4 | Label 5 | Label 6 |
| 0.952 | 0.935 | 0.907 | 0.906 | 0.883 | 0.891 | 0.931 | 0.929 | 0.918 | 0.911 |
| Instance segmentation | | | | | | | | | |
| *mAP@50* | *mAP@75* | *mAP@90* | *mAP* | *AP* | | | | | |
| | | | | Label 1 | Label 2 | Label 3 | Label 4 | Label 5 | Label 6 |
| 1.000 | 1.000 | 0.346 | 0.824 | 0.754 | 0.809 | 0.862 | 0.829 | 0.854 | 0.833 |

a cleaner instance point cloud for the segments.

Overall, from the cases of the four completely different TBM tunnel scenarios mentioned above, it can be seen that For the T1&T2 tunnels, our SAM4Tun demonstrated a great advantage compared to the provided benchmarks. Furthermore, when applied to different tunnel scenarios, it shows that the obtained results in case T3, T4&T5 were comparable to those in the T1&T2 cases. The results from the STSD dataset, obtained using MLS, exhibit a similar pattern to those observed in the Seg2Tunnel dataset. Experimental results confirmed our hypothesis: due to the highly structured nature of our target, the tunnel lining segment, excellent results can be achieved using an unmodified LVM combined with well-crafted prompt engineering. Overall, the proposed method achieves an $mAP$ value greater than 80 and an $mIoU$ value greater than 90 in all scenarios. This demonstrates great stability and

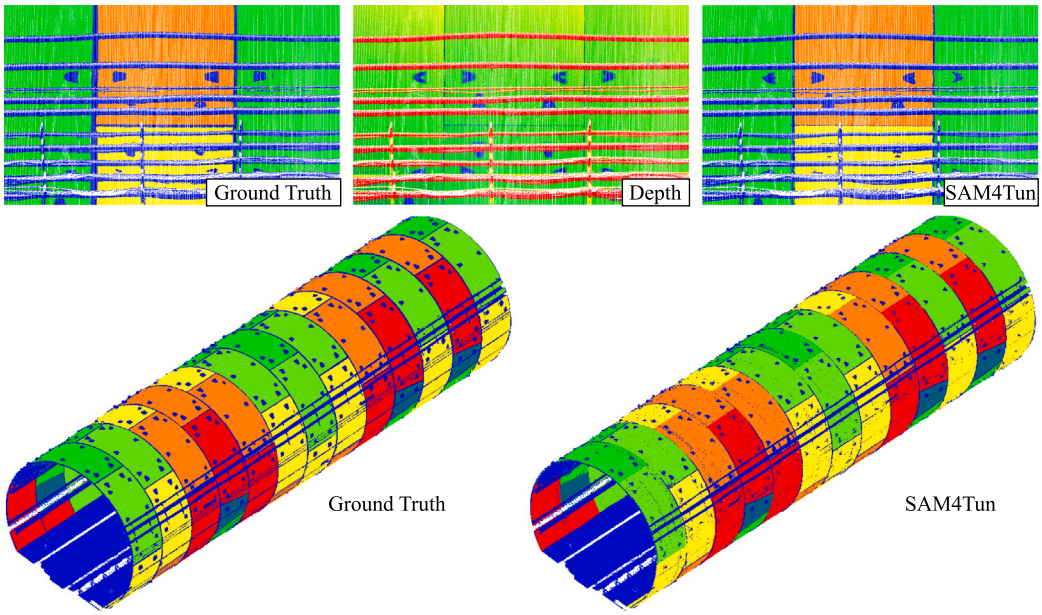


**Fig. 14.** The ground truth and predicted results of STSD circular tunnels.

generalization of our proposed method.

### 5.3. Discussion

#### 5.3.1. Improvement of multiple step up-sampling

Previous research has demonstrated that under conditions with high point cloud density and minimal noise, high-quality depth maps can be generated, allowing effective segmentation using image processing techniques (Duan et al., 2021; Yi et al., 2019). Therefore, one of the main challenges arises when the point cloud density is unevenly distributed or contains a lot of noise. This also means that improving the quality of the depth map in low-density areas would lead to better results. Meanwhile, from the perspective of using SAM, especially when the point cloud density is sparse in distant areas, interpolation must be introduced to fill the depth map. If left empty, prompt points could fall on these empty pixels, severely affecting the generation of the segmentation mask. Therefore, in this paper we introduced a more scientifically grounded interpolation method.

The proposed multiple step up-sampling method offers two benefits. First, from a theoretical standpoint, this approach aligns more closely with logical principles and underlying theoretical rules, as up-sampling is achieved by filling the space between scanning lines, aligning better with the characteristics of laser scanners. The second benefit is that after up-sampling, generating the depth map requires only a smaller sliding window to fill in the remaining small gaps. In contrast, direct interpolation on the depth map requires increasing the sliding window size from $9 \times 9$ to $13 \times 13$, based on tests, to achieve effective filling. According to our algorithm, a larger window increases the judgment range for null pixels, which may introduce more unnecessary interpolation. We illustrate an example of the edge region in Fig. 15. We can observe that interpolation directly on the depth map results in unsmooth joints, which clearly does not reflect the real situation.

#### 5.3.2. Background enhancement

In the previous case, we found that some errors mainly stemmed from the inaccurate detection of bolt holes in the background category. Part of the reason is the discrepancy between the annotations and our input depth map (see Fig. 13), and another part is due to the imbalance between positive and negative prompt points (see Table 9). We cannot improve the former, but we conducted further analysis on the latter. We discovered that this issue was particularly prone to occur when the bolt holes on the segments were sealed. According to our design, we automatically removed the corresponding negative prompt points when the bolt holes were blocked, which reduced the total number of sampled negative prompt points within the entire segment. As a result, the remaining negative prompt points were not sufficient, causing regions that should have been segmented as background to be identified as foreground. This also indicates that the weight of positive prompt points should be greater than that of negative hint points. Therefore, we proposed an improvement. We reversed the types of prompt points within the segment, identifying the detected bolt holes as foreground and the rest as background. We then used the detected mask to update the point cloud labels as background. Using stations T1-12 as an example for testing, we found that both *mAP* and *mIoU* improved, as shown in Table 9. We present an example of initial and improved segmentation in Fig. 16.

#### 5.3.3. Reference execution speed

Our code runs on the Python platform and has not undergone special optimizations. We only used Numba (Lam et al., 2015) to optimize the repetitive and dense computations in the point cloud processing. We also used CPU parallel processing, setting it to use 16 cores. We did not incorporate GPU acceleration technology, and only used the GPU for the final SAM inference step. We provided a reference execution speed by testing with a point cloud of approximately 14 million points. The testing hardware included an Intel i7-12700K CPU, 64 GB of memory, and NVIDIA GeForce RTX 3070 Ti. A rough reference time is provided as following:

- Unfolding: Projection coordinate generation - 0.8 min
- Denoising: Filter out background through local density variation – 0.2 min; Outlier point extraction – 0.5 min
- Enhancing: Curvature calculation and multi-step up-sampling – 2.5 min; Projection image generation – 2.0 min
- Prompting and segmenting: Pormpt generation and SAM-based segmentation - 2 min; Re-projection – 0.5 min

Therefore, it takes approximately 8.5 min to complete the zero-shot instance-level segmentation of the tunnel lining segment point cloud for one single scanning station.

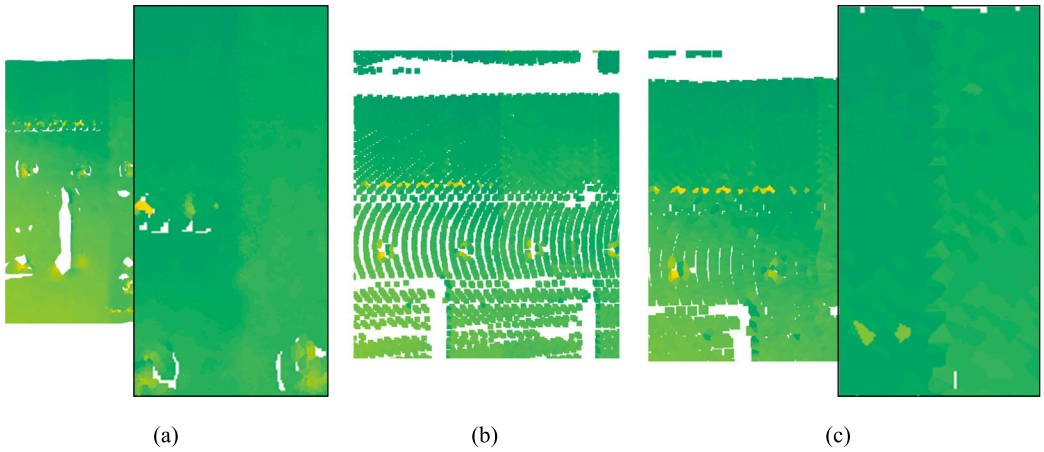


**Fig. 15.** Different interpolation methods for depth map generation: (a) our up-sampling method with a window size of 9 × 9; (b) direct interpolation with a window size of 9 × 9; (c) direct interpolation with a window size of 13 × 13.

**Table 9**
Testing the effect of reversing prompt points for background enhancement using T1–12 as an example.

| Method | *mIoU* | *IoU* | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | Label 0 | Label 1 | Label 2 | Label 3 | Label 4 | Label 5 | Label 6 |
| Original | 0.920 | 0.849 | 0.929 | 0.0.951 | 0.938 | 0.904 | 0.943 | 0.928 |
| Background enhanced | 0.926 | 0.865 | 0.936 | 0.954 | 0.942 | 0.904 | 0.947 | 0.937 |
| | *mAP* | *AP* | | | | | | |
| | | – | Label 1 | Label 2 | Label 3 | Label 4 | Label 5 | Label 6 |
| Original | 0.830 | – | 0.778 | 0.903 | 0.913 | 0.744 | 0.805 | 0.838 |
| Background enhanced | 0.835 | – | 0.794 | 0.906 | 0.913 | 0.744 | 0.806 | 0.844 |

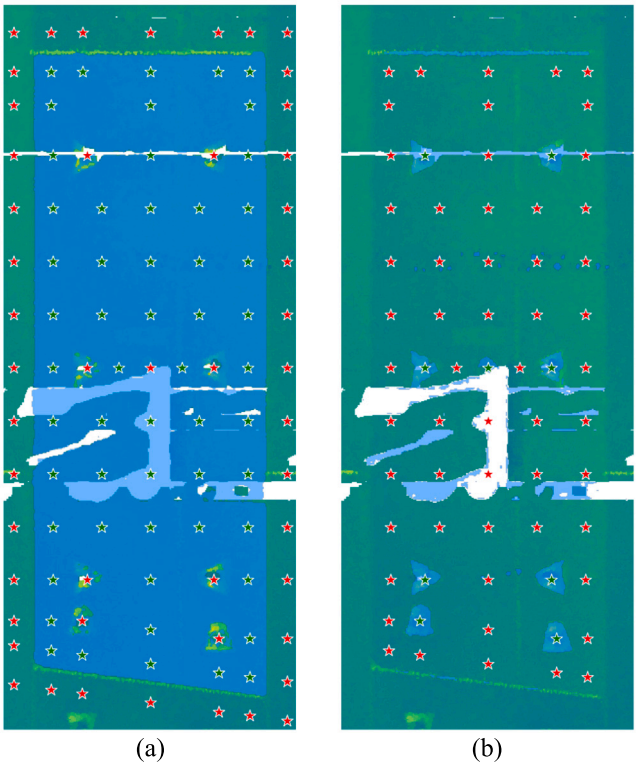


**Fig. 16.** Example of swapping foreground and background prompt points (green are positive, red are negative): (a) original segmentation; (b) exchange positive and negative prompt points. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

### 5.3.4. Advantages and limitations

In this section, we will discuss the advantages and limitations of prompt-based segmentation methods. First, the greatest advantage is that no training is required. The proposed method combines the strengths of previous techniques purely based on digital image processing and deep learning algorithms, making it capable of handling relatively low-density point clouds and complex interference scenarios. Secondly, it offers high robustness. The input for the segmentation algorithm is solely the prompt, which can tolerate a certain degree of deviation. As shown in Fig. 17(a), even if the overall prompts are shifted, as long as the points remain within the acceptable range, it will not significantly affect the results. For example, in Fig. 17(a), although the negative prompt point exceeds the range due to the small background area at the centre, the foreground mask can still be successfully segmented as our required, due to other negative prompt points still sampled within the bolt area.

When determining the central axis of the tunnel, errors are inevitably introduced, which results in imperfect projections. The error in the central axis causes deviations in the length of the unfolded segments and distortions in the joints between rings. For this reason, directly applying fixed templates based on design information to segment depth maps introduces errors (Duan et al., 2021). SAM addresses this issue by automatically retrieving boundaries based on a coarse prompt, regardless of whether the boundaries are warped or deformed. When crossing a boundary, it can be observed on the logits map (see in Fig. 17(d)) that values drop sharply to negative for clear boundaries, while for fuzzier boundaries, the logits value drops closer to zero. However, SAM also performs a segmentation on the opposite side of the boundary, where the logits value inside the boundary will be significantly greater than zero. As a result, when stitching the complete depth map, assigning pixels to the instance based on the highest logits becomes quite robust.

Moreover, the bias introduced by the central axis arises when generating new prompts on both sides of the segment from the initial prompt point. This occurs because our prompt generation method relies solely

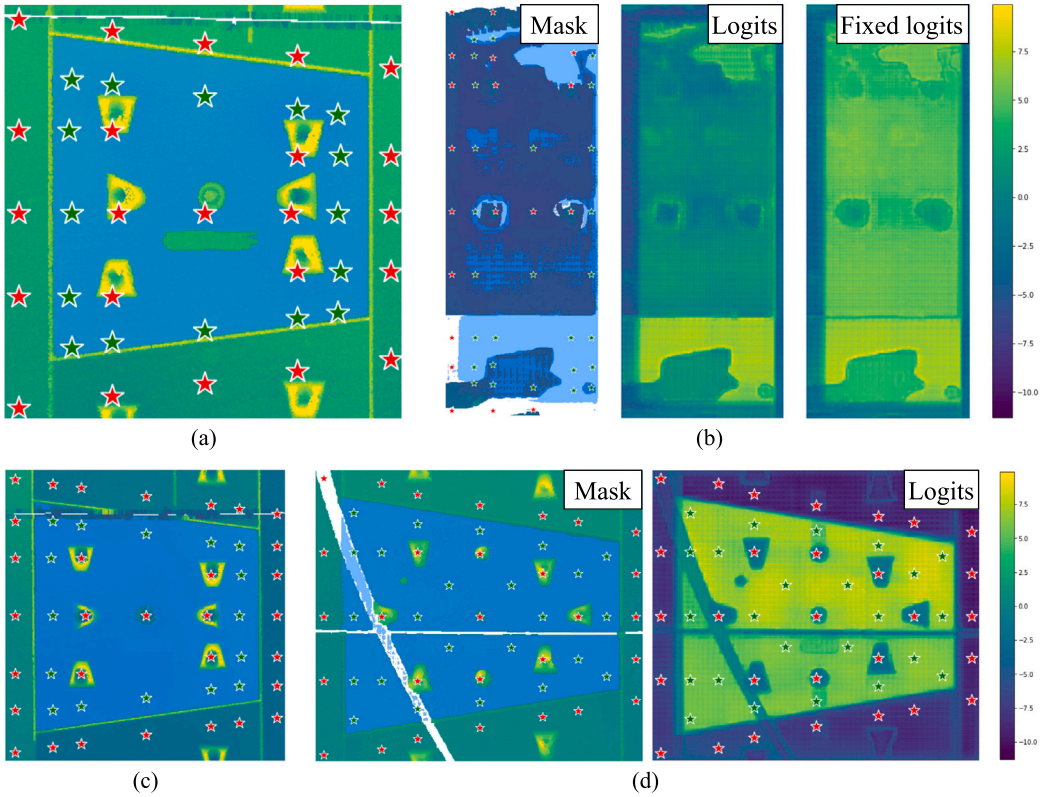


**Fig. 17.** Visualization of advantages and limitations (green are positive prompt points, red are negative prompt points): (a) offset robustness of prompt; (b) increased redundant prompt points enhance robustness; (c) interference occurs at edges; (d) interference occurs at middle. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

on the initial prompt point within each ring to create all subsequent prompts. However, in our experiments, we did not observe this bias to be significant enough to affect SAM's segmentation.

SAM as prompt-based method is lacking semantic understanding and operates entirely based on the provided prompts, requiring time to carefully craft and tailor the prompts. Fig. 17(b) illustrates an example demonstrating a case of failed segmentation, as well as why we introduced a high redundancy of prompt points. In certain cases, due to the prior filtering of non-segment point clouds and occlusions caused during the scanning process, large blank areas can appear in parts of the depth map. If the number of prompt points introduced is insufficient, and a significant number of positive prompt points happen to fall within these blank areas (where we reverse the prompt point attributes instead of removing them when encountering sealed bolt holes in this experiment), this can lead to segmentation issues. In this case the blank areas are incorrectly identified as foreground, as seen in the mask and corresponding logits in Fig. 17(b). However, using the denser prompt points that we later adopted, mitigated this problem, improving the logits score for the segment area and achieving successful segmentation. Additionally, the number of prompts does not affect the inference speed.

SAM will search boundaries automatically in the image only based on prompts, and as shown in Fig. 17(c), when boundaries are obstructed – such as when a cable crosses exactly at the edge of the segment – it causes the mask to fail to extend, resulting in a missing portion in the upper left corner. Adding an additional positive prompt point in the upper left corner can alleviate this issue but significantly reduces the robustness of the prompts for other segmentation. In contrast, as shown in Fig. 17(d), when the obstruction crosses through the middle of the segment, it does not affect the overall segmentation result.

Overall, despite some limitations, this LVM-based zero-shot segmentation method shows great promise. By projecting point cloud data onto images, the data is structured in a way that allows powerful image-based LVM to be applied to point clouds. This is especially beneficial for tunnel point cloud segmentation, where the highly regular and structured nature of TBM tunnels enables robust segmentation with a unified prompt. This provides a foundation for subsequent point cloud-based tasks such as deformation detection, damage monitoring, and maintenance work.

## 6. Conclusion

In this paper, we introduce SAM4Tun, a novel approach to address the segmental tunnel linings automated instance segmentation based on point clouds. We extend a representative LVM model, prompt-based SAM, to point clouds segmentation by projecting unstructured tunnel point clouds onto enhanced structured panoramic unfolded images. In the projection process, we automatically filter out interference, enhance the boundaries between segments, and optimize the issue of point cloud unevenness. We then focus on prompt engineering and use digital image processing techniques to automate template prompt generation, enabling zero-shot segmentation. Our proposed SAM4Tun for segmenting tunnel linings was validated in three different tunnel scenarios, demonstrated excellent generalization and stability. The main contribution of this paper are:

- The proposed framework is fully automated and requires no training, addressing the issue of reliance on annotated data. Through the simple integration of repetitive and dense computation optimization and parallel computing, our model only takes 8.5 min to complete the instance segmentation of tunnel lining with 14 million points and a total length around 30 m.
- In our workflow, to address interference from complex ancillary structures, we incorporated train-free filtering methods based on local point cloud density variations, along with an outlier

extraction method based on local depth variations, to effectively filter out non-segmented points.

- To generate more accurate depth map projection images and address the issue of uneven point clouds, we introduce an interpolation approach tailored to scanner characteristics, specifically designed to interpolate between scan lines. Additionally, based on extracted outlier, we used the Hough transform to identify different types of joints, strengthening segment boundaries on the depth maps and plays a key role in generating the initial prompt points.
- We validated our proposed no-training model, SAM4Tun, on the Seg2Tunnel dataset and found that it outperforms the best supervised learning benchmarks in both semantic segmentation (from 93.2 to 95.6 in *OA*, and 83.1 to 91.8 in *mIoU*) and instance segmentation (from 73.3 to 80.9 in *mAP*) tasks. We also tested it on two other different TBM tunnel point cloud scenarios and achieved comparable results, demonstrating the generalization of the proposed method. The zero-shot capability of the LVM aligns perfectly with segmental tunnel linings segmentation task. The segmentation results lay the foundation for automated displacement monitoring and component-level damage localization at the ring or segment level of the tunnel.

While we have reduced the reliance on annotated data to some extent, this also introduces some limitations. SAM lacks semantic understanding and relies solely on provided prompts. In cases with large blank areas or significant occlusions, insufficient prompt points can lead to segmentation issues. Proposed method dependence on K-block recognition and inference from surrounding rings in the case of occlusions, which may introduce errors.

In the future, we can refine the prompt generation process to produce more accurate prompts. Introducing fine-tuning should improve segmentation performance by enabling models to better grasp the semantics of specific environments. Moreover, it is valuable for exploring more advanced zero-shot LVMs to tackle engineering tasks.

## CRediT authorship contribution statement

**Zehao Ye:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Formal analysis, Data curation, Conceptualization. **Wei Lin:** Writing – review & editing, Data curation. **Asaad Faramarzi:** Writing – review & editing, Supervision. **Xiongyao Xie:** Supervision. **Jelena Ninić:** Writing – review & editing, Supervision, Methodology, Conceptualization.

## 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

The computations described in this research were performed using the Baskerville Tier 2 HPC service (https://www.baskerville.ac.uk/). Baskerville was funded by the EPSRC and UKRI through the World Class Labs scheme (EP/T022221/1) and the Digital Research Infrastructure programme (EP/W032244/1) and is operated by Advanced Research Computing at the University of Birmingham. The computations described in this paper were also performed using the University of Birmingham's BlueBEAR (http://www.birmingham.ac.uk/bear) HPC service, which provides a High Performance Computing service to the University's research community.

## Data availability

Data will be made available on request.

## References

Archit, A., Nair, S., Khalid, N., Hilt, P., Rajashekar, V., Freitag, M., Gupta, S., Dengel, A., Ahmed, S., Pape, C., 2023. Segment anything for microscopy. Pages: 2023.08.21.554208 Section: New Results.

Attard, L., Debono, C.J., Valentino, G., Di Castro, M., 2018. Tunnel inspection using photogrammetric techniques and image processing: A review. ISPRS J. Photogramm. Remote Sens. 144, 180–188.

Cao, Z., Chen, D., Peethambaran, J., Zhang, Z., Xia, S., Zhang, L., 2021. Tunnel reconstruction with block level precision by combining data-driven segmentation and model-driven assembly. IEEE Trans. Geosci. Remote Sens. 59 (10), 8853–8872, Conference Name: IEEE Transactions on Geoscience and Remote Sensing.

Chen, Q., Kang, Z., Cao, Z., Xie, X., Guan, B., Pan, Y., Chang, J., 2024b. Combining cylindrical voxel and mask R-CNN for automatic detection of water leakages in shield tunnel point clouds. Remote Sens. 16 (5), 896, Number: 5 Publisher: Multidisciplinary Digital Publishing Institute.

Chen, K., Liu, C., Chen, H., Zhang, H., Li, W., Zou, Z., Shi, Z., 2024a. Rsprompter: Learning to prompt for remote sensing instance segmentation based on visual foundation model. IEEE Trans. Geosci. Remote Sens. 62, 1–17, Conference Name: IEEE Transactions on Geoscience and Remote Sensing.

Chen, K., Zou, Z., Shi, Z., 2021. Building extraction from remote sensing images with sparse token transformers. Remote Sens. 13 (21), 4441, Number: 21 Publisher: Multidisciplinary Digital Publishing Institute.

Cheng, Y.-J., Qiu, W.-G., Duan, D.-Y., 2019. Automatic creation of as-is building information model from single-track railway tunnel point clouds. Autom. Constr. 106, 102911.

Cui, H., Li, J., Mao, Q., Hu, Q., Dong, C., Tao, Y., 2024. STSD:A large-scale benchmark for semantic segmentation of subway tunnel point cloud. Tunn. Undergr. Space Technol. 150, 105829.

Cui, H., Ren, X., Mao, Q., Hu, Q., Wang, W., 2019. Shield subway tunnel deformation detection based on mobile laser scanning. Autom. Constr. 106, 102889.

Ding, K., Xiang, S., Pan, C., 2024. Smalnet: Segment anything model aided lightweight network for infrared image segmentation. Infrared Phys. Technol. 142, 105540.

Duan, D.-Y., Qiu, W.-G., Cheng, Y.-J., Zheng, Y.-C., Lu, F., 2021. Reconstruction of shield tunnel lining using point cloud. Autom. Constr. 130, 103860.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. KDD '96, AAAI Press, Portland, Oregon, pp. 226–231.

Fischler, M.A., Bolles, R.C., 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24 (6), 381–395.

Ge, K., Wang, C., Guo, Y.T., Tang, Y.S., Hu, Z.Z., Chen, H.B., 2024. Fine-tuning vision foundation model for crack segmentation in civil infrastructures. Constr. Build. Mater. 431, 136573.

Gong, Q., Liu, Q., Zhang, Q., 2016. Tunnel boring machines (TBMs) in difficult grounds. Tunnel Boring Machines in Difficult Grounds, Tunn. Undergr. Space Technol. Tunnel Boring Machines in Difficult Grounds, 57.1–3,

Han, X.-F., Jin, J.S., Wang, M.-J., Jiang, W., Gao, L., Xiao, L., 2017. A review of algorithms for filtering the 3D point cloud. Signal Process., Image Commun. 57, 103–112.

Hegemann, F., Stascheit, J., Maidl, U., 2020. As-built documentation of segmental lining rings in the BIM representation of tunnels. Tunn. Undergr. Space Technol. 106, 103582.

Hu, Q., Yang, B., Xie, L., Rosa, S., Guo, Y., Wang, Z., Trigoni, N., Markham, A., 2022. Learning semantic segmentation of large-scale point clouds with random sampling. IEEE Trans. Pattern Anal. Mach. Intell. 1–1.

Huang, H., Cheng, W., Zhou, M., Chen, J., Zhao, S., 2020. Towards automated 3D inspection of water leakages in shield tunnel linings using mobile laser scanning data. Sensors 20 (22), 6669, Number: 22 Publisher: Multidisciplinary Digital Publishing Institute.

Huang, M.Q., Ninić, J., Zhang, Q.B., 2021b. BIM, machine learning and computer vision techniques in underground construction: Current status and future perspectives. Tunn. Undergr. Space Technol. 108, 103677.

Huang, H., Zhang, C., Hammad, A., 2021a. Effective scanning range estimation for using TLS in construction projects. J. Constr. Eng. Manag. 147 (9), 04021106, Publisher: American Society of Civil Engineers.

Illingworth, J., Kittler, J., 1988. A survey of the hough transform. Comput. Vis. Graph. Image Process. 44 (1), 87–116.

ITA-AITES, 2021. Tunnel market survey 2019 (ITA-AITES).

Ji, A., Chew, A.W.Z., Xue, X., Zhang, L., 2022. An encoder-decoder deep learning method for multi-class object segmentation from 3D tunnel point clouds. Autom. Constr. 137, 104187.

Ji, A., Zhou, Y., Zhang, L., Tiong, R.L.K., Xue, X., 2023. Semi-supervised learning-based point cloud network for segmentation of 3D tunnel scenes. Autom. Constr. 146, 104668.

Kang, J., Chen, N., Li, M., Mao, S., Zhang, H., Fan, Y., Liu, H., 2024. A point cloud segmentation method for dim and cluttered underground tunnel scenes based on the segment anything model. Remote Sens. 16 (1), 97, Number: 1 Publisher: Multidisciplinary Digital Publishing Institute.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T.B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., Amodei, D., 2020. Scaling laws for neural language models. arXiv:2001.08361 [cs, stat].

Ke, L., Ye, M., Danelljan, M., Liu, Y., Tai, Y.-W., Tang, C.-K., Yu, F., 2023. Segment anything in high quality. Adv. Neural Inf. Process. Syst. 36, 29914–29934.

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. In: 2023 IEEE/CVF International Conference on Computer Vision (ICCV). (ISSN: 2380-7504) pp. 3992–4003.

Kolodiazhnyi, M., Vorontsova, A., Konushin, A., Rukhovich, D., 2024. Top-down beats bottom-up in 3D instance segmentation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 3566–3574.

Lam, S.K., Pitrou, A., Seibert, S., 2015. Numba: a LLVM-based python JIT compiler. In: Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC. LLVM '15, Association for Computing Machinery, New York, NY, USA, pp. 1–6.

Li, J., Jing, L., Zheng, X., Li, P., Yang, C., 2019. Application and outlook of information and intelligence technology for safe and efficient TBM construction. Tunn. Undergr. Space Technol. 93, 103097.

Li, P., Wang, Q., Li, J., Pei, Y., He, P., 2024. Automated extraction of tunnel leakage location and area from 3D laser scanning point clouds. Opt. Lasers Eng. 178, 108217.

Lin, W., Li, P., Xie, X., 2022. A novel detection and assessment method for operational defects of pipe jacking tunnel based on 3D longitudinal deformation curve: A case study. Sensors 22 (19), 7648, Number: 19 Publisher: Multidisciplinary Digital Publishing Institute.

Lin, W., Li, P., Xie, X., Cao, Y., Zhang, Y., 2023b. A novel back-analysis approach for the external loads on shield tunnel lining in service based on monitored deformation. Struct. Control Health Monit. 2023 (1), 8128701, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1155/2023/8128701.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L., 2014. Microsoft COCO: Common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (Eds.), Computer Vision – ECCV 2014. In: Lecture Notes in Computer Science, Springer International Publishing, Cham, pp. 740–755.

Lin, W., Sheil, B., Zhang, P., Zhou, B., Wang, C., Xie, X., 2024. Seg2Tunnel: A hierarchical point cloud dataset and benchmarks for segmentation of segmental tunnel linings. Tunn. Undergr. Space Technol. 147, 105735.

Lin, P., Zhang, L., Tiong, R.L.K., 2023a. Multi-objective robust optimization for enhanced safety in large-diameter tunnel construction with interactive and explainable AI. Reliab. Eng. Syst. Saf. 234, 109172.

Liu, S., Sun, H., Zhang, Z., Li, Y., Zhong, R., Li, J., Chen, S., 2022. A multiscale deep feature for the instance segmentation of water leakages in tunnel using MLS point cloud intensity images. IEEE Trans. Geosci. Remote Sens. 60, 1–16, Conference Name: IEEE Transactions on Geoscience and Remote Sensing.

Lüddecke, T., Ecker, A., 2022. Image segmentation using text and image prompts. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7086–7096.

Menendez, E., Victores, J.G., Montero, R., Martínez, S., Balaguer, C., 2018. Tunnel structural inspection and assessment using an autonomous robotic system. Autom. Constr. 87, 117–126.

Ng, A., Morgan, A., Verre, J., Kaiser, C., 2023. DeepLearning.AI - prompt engineering for vision models.

Ninić, J., Freitag, S., Meschke, G., 2017. A hybrid finite element and surrogate modelling approach for simulation and monitoring supported TBM steering. Tunn. Undergr. Space Technol. 63, 12–28.

Ninić, J., Koch, C., Vonthron, A., Tizani, W., König, M., 2020. Integrated parametric multi-level information and numerical modelling of mechanised tunnelling projects. Adv. Eng. Inform. 43, 101011.

Ninić, J., Meschke, G., 2015. Model update and real-time steering of tunnel boring machines using simulation-based meta models. Tunn. Undergr. Space Technol. 45, 138–152.

Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Assran, M., Ballas, N., Galuba, W., Howes, R., Huang, P.-Y., Li, S.-W., Misra, I., Rabbat, M., Sharma, V., Synnaeve, G., Xu, H., Jegou, H., Mairal, J., Labatut, P., Joulin, A., Bojanowski, P., 2024. DINOv2: Learning robust visual features without supervision. arXiv:2304.07193 [cs].

Qiu, W., Cheng, Y.-J., 2017. High-resolution DEM generation of railway tunnel surface using terrestrial laser scanning data for clearance inspection. J. Comput. Civ. Eng. 31 (1), 04016045, Publisher: American Society of Civil Engineers.

Salmi, J., Ye, Z., Ninić, J., Heikkilä, R., 2025. Bim for mining - automated generation of information models using a parametric modelling concept. 186, 106032. http://dx.doi.org/10.1016/j.ijrmms.2025.106032,

Schult, J., Engelmann, F., Hermans, A., Litany, O., Tang, S., Leibe, B., 2023. Mask3D: Mask transformer for 3D semantic instance segmentation. In: 2023 IEEE International Conference on Robotics and Automation (ICRA). IEEE, London, United Kingdom, pp. 8216–8223.

Shi, B., Yang, M., Liu, J., Han, B., Zhao, K., 2023. Rail transit shield tunnel deformation detection method based on cloth simulation filtering with point cloud cylindrical projection. Tunn. Undergr. Space Technol. 135, 105031.

Sun, H., Xu, Z., Yao, L., Zhong, R., Du, L., Wu, H., 2020. Tunnel monitoring and measuring system using mobile laser scanning: Design and deployment. Remote Sens. 12 (4), 730, Number: 4 Publisher: Multidisciplinary Digital Publishing Institute.

Wang, B., Chen, Z., Li, M., Wang, Q., Yin, C., Cheng, J.C.P., 2024. Omni-Scan2BIM: A ready-to-use Scan2BIM approach based on vision foundation models for MEP scenes. Autom. Constr. 162, 105384.

Wang, W., Dai, J., Chen, Z., Huang, Z., Li, Z., Zhu, X., Hu, X., Lu, T., Lu, L., Li, H., Wang, X., Qiao, Y., 2023b. InternImage: Exploring large-scale vision foundation models with deformable convolutions. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Vancouver, BC, Canada, pp. 14408–14419.

Wang, J., Liu, Z., Zhao, L., Wu, Z., Ma, C., Yu, S., Dai, H., Yang, Q., Liu, Y., Zhang, S., Shi, E., Pan, Y., Zhang, T., Zhu, D., Li, X., Jiang, X., Ge, B., Yuan, Y., Shen, D., Liu, T., Zhang, S., 2023a. Review of large vision models and visual prompt engineering. Meta- Radiol. 1 (3), 100047.

Xie, X., Lu, X., 2017. Development of a 3D modeling algorithm for tunnel deformation monitoring based on terrestrial laser scanning. Undergr. Space 2 (1), 16–29.

Xie, X., Tian, H., Zhou, B., Li, K., 2021. The life-cycle development and cause analysis of large diameter shield tunnel convergence in soft soil area. Tunn. Undergr. Space Technol. 107, 103680.

Xu, C., Cao, B.T., Yuan, Y., Meschke, G., 2023. Transfer learning based physics-informed neural networks for solving inverse problems in engineering structures under different loading scenarios. Comput. Methods Appl. Mech. Engrg. 405, 115852.

Xu, J., Ding, L., Luo, H., Chen, E.J., Wei, L., 2019. Near real-time circular tunnel shield segment assembly quality inspection using point cloud data: A case study. Tunn. Undergr. Space Technol. 91, 102998.

Ye, Z., Lovell, L., Faramarzi, A., Ninić, J., 2024. Sam-based instance segmentation models for the automation of structural damage detection. Adv. Eng. Inform. 62, 102826.

Yi, C., Lu, D., Xie, Q., Liu, S., Li, H., Wei, M., Wang, J., 2019. Hierarchical tunnel modeling from 3D raw lidar point cloud. Comput. Aided Des. 114, 143–154.

Yu, Y., Wang, C., Fu, Q., Kou, R., Huang, F., Yang, B., Yang, T., Gao, M., 2023. Techniques and challenges of image segmentation: A review. Electron. 12 (5), 1199, Number: 5 Publisher: Multidisciplinary Digital Publishing Institute.

Zhang, Z., Ji, A., Wang, K., Zhang, L., 2022. UnrollingNet: An attention-based deep learning approach for the segmentation of large-scale point clouds of tunnels. Autom. Constr. 142, 104456.

Zhang, R., Jiang, Z., Guo, Z., Yan, S., Pan, J., Dong, H., Gao, P., Li, H., 2023. Personalize segment anything model with one shot. arXiv:2305.03048 [cs].

Zhang, Y., Ren, X., Zhang, J., Ma, Z., 2024. A method for deformation detection and reconstruction of shield tunnel based on point cloud. J. Constr. Eng. Manag. 150 (3), 04024006, Publisher: American Society of Civil Engineers.

Zhou, M., Cheng, W., Huang, H., Chen, J., 2021. A novel approach to automated 3D spalling defects inspection in railway tunnel linings using laser intensity and depth information. Sensors 21 (17), 5725, Number: 17 Publisher: Multidisciplinary Digital Publishing Institute.

Zhou, Y., Ji, A., Zhang, L., Xue, X., 2023. Attention-enhanced sampling point cloud network (aspcnet) for efficient 3D tunnel semantic segmentation. Autom. Constr. 146, 104667.