Journal of Civil Structural Health Monitoring (2025) 15:3345–3360
https://doi.org/10.1007/s13349-025-00993-z

ORIGINAL PAPER

# Defect segmentation and 3D reconstruction in concrete structures using SAM 2 and 3D Gaussian splatting

Dayou Duan[1,3,4] · Zuocai Wang[2] · Yu Xin[4] · Yajie Ding[1]

Received: 24 February 2025 / Accepted: 7 July 2025 / Published online: 17 July 2025

**Abstract**
Concrete defects usually diminish the structural aesthetics, and more importantly, they may significantly impact the safety and durability of structures. While conventional inspection methods are often subjective and inefficient, modern computer vision techniques offer a more automated and accurate alternative. To capture the spatial information of defects, a framework combining Segment Anything Model 2 (SAM 2) and 3D Gaussian Splatting (3DGS) was proposed for high-quality defect segmentation and 3D reconstruction based on monocular video. The proposed framework demonstrates advantages in surface detail, defect segmentation, and rendering speed compared to existing approaches. Key contributions of this work include the effective integration of SAM 2 and 3DGS, improved surface feature representation, and an effective solution for 3D reconstruction using monocular video. Experimental results highlight the framework's improvements in accuracy, detail, and scalability over existing methods.

**Keywords** Concrete defect segmentation · 3D reconstruction · SAM 2 · 3DGS · Monocular computer vision

## 1 Introduction

As a widely used building material in infrastructures, such as bridges and buildings, concrete is prone to cracks or other defects during construction or long-term operation, which can compromise the durability and safety of the structures. If the defects are not detected and repaired in time, they may develop into structural damage and potentially lead to serious accidents [1]. Defects are key indicators of concrete structure health. By obtaining the geometric and spatial information, their severity can be accurately assessed [2]. However, traditional manual defect detection methods are often inefficient, subjective, and pose safety risks [3]. Therefore, improving the efficiency and reliability of defect inspection has become a critical issue [4].

Recently, computer vision and artificial intelligence technologies have been successfully implemented to detect defects in concrete structures. These technologies demonstrate significant advantages in terms of cost-effectiveness, high resolution, reliability, and applicability [5]. Various methods have been developed for visual defect detection, ranging from traditional image processing algorithms to modern deep learning-based approaches. Traditional detection techniques rely on manually extracted features to convert raw data into meaningful information. In contrast, deep learning-based visual technologies are widely used for defect information extraction due to their high accuracy and automation. However, these techniques typically require large-scale data sets to train convolutional neural networks [6, 7], and the training process often demands substantial time and computational resources.

There are numerous methods for segmenting concrete defects in 2D images, but these segmentations lack 3D spatial information. Currently, various 3D reconstruction methods, such as LiDAR scanning and binocular cameras, have been developed [4]. However, these methods are often expensive, time-consuming, and have limited applicability. In contrast, monocular vision-based methods offer a

✉ Dayou Duan
duandy@hfuu.edu.cn

[1] School of Urban Construction and Transportation, Hefei University, Hefei 230601, Anhui, China

[2] School of Civil Engineering, Anhui Jianzhu University, Hefei 230009, Anhui, China

[3] Anhui Province Architectural Design and Research Institute Co., Ltd., Hefei 230093, Anhui, China

[4] School of Civil Engineering, Hefei University of Technology, Anhui 230009, China

cost-effective and scalable solution for generating 3D models using common video data. In 3D reconstruction, the key challenges include feature matching, position estimation, and texture rendering.

To address these challenges, this study adopts the pre-trained SAM 2 [8] and 3DGS [9] for high-quality 3D defect segmentation and rendering. The recently released SAM 2 has been trained on a massive data set, enabling it to accurately segment zero-shot (not seen during training) objects with minimal prompts. Additionally, it can segment and track objects across video frames. On the other hand, 3DGS is a powerful technique for volumetric rendering and point cloud representation. It models surfaces and structures as a collection of Gaussian splats, enabling high-quality reconstruction with smooth surfaces and fine detail preservation. This technique is particularly well-suited for the irregular and unstructured surfaces of concrete structures, where traditional point-based reconstruction methods often struggle to capture fine details or fill sparse regions.

This study develops a robust and efficient framework for 3D reconstruction and defect segmentation in concrete structures by leveraging the strengths of SAM 2 and 3DGS to overcome the limitations of existing technologies. The proposed framework enhances surface detail reconstruction, enables precise defect segmentation, and accelerates the reconstruction process for large concrete structures. The main contributions of this work include:

(1) Integrating SAM 2 and 3DGS to achieve efficient 3D reconstruction of concrete structures;
(2) Utilizing image segmentation for efficient feature matching and 3D position estimation;
(3) Achieving highly accurate defect segmentation.

This paper is organized into several sections to systematically address the research objectives. Section 2 provides an overview of prior studies in vision-based concrete damage detection and 3D reconstruction methods. Section 3 details the SAM 2 and 3DGS and the framework reconstruct concrete structure from monocular video. Section 4 presents the experimental outcomes, including comparisons with state-of-the-art methods. Section 5 explores the broader implications of the findings and identifies potential directions for future research. In the last section, conclusions are drawn from the above content.

## 2 Related work

### 2.1 Deep learning-based damage detection

Conventional vision-based concrete damage detection methods typically relied on threshold segmentation and edge detection to segment observable defects in images. With the increase in computing power, crack detection methods based on Convolutional Neural Networks (CNNs) have been applied to detect defects in road pavements and infrastructure [10, 11]. Utilizing techniques including convolution, spatial pyramid and skip connections, Ren et al. [12] took deep fully convolutional networks to achieve concrete crack detection. CNN-based Res-Unet architectures have also been deployed to identify common concrete defects [13]. Additionally, many transformer-based deep learning architectures, such as vision transformer [14] and SegFormer [15] have been widely adopted in concrete crack detection [16]. An improved Swin-Transformer-UNet was developed to detect the concrete cracks, incorporating a feature pyramid network for the decoding process [17].

With the trend of training on large-scale data sets, pre-trained models have demonstrated strong generalization capabilities, particularly in tasks beyond their initial training scope [18]. Research has consistently shown that as these models scale up, their performance improves [19]. This trend is especially evident in computer vision and language encoding, two core applications of foundation models. Despite these impressive advancements, substantial technical challenges remain in these domains, largely due to limited training data. The Segment Anything Model (SAM) [20], developed by Meta AI Research, changed conventional segmentation by introducing a promptable visual segmentation task, leveraging the largest segmentation data set in computer vision.

### 2.2 3D reconstruction and rendering

Early approaches to synthesis relied on light fields, initially requiring dense sampling [21, 22] and then allowing unstructured capture [23]. The introduction of structure-from-motion (SfM) [24] opened new possibilities using photo collections to estimate a point cloud during camera calibration. Subsequent advancements in multi-view stereo evolved into sophisticated 3D reconstruction algorithms [25], paving the way for various view synthesis techniques [26–28].

Recent advances in neural rendering algorithms have reduced some artifacts and eliminated the need to cache all data in video random-access memory, outperforming traditional methods in most aspects [29]. Soft3D [30] pioneered view synthesis by maintaining depth uncertainty throughout the 3D reconstruction process. Deep learning methods were developed to generate 3D shapes that closely approximate the ground truth geometry [31, 32]. To address the high computational cost of volumetric ray-marching-based deep learning methods, Neural Radiance Fields method [33] was proposed, offering an optimized neural radiance field and high-quality 3D rendering.

Generating point clouds of objects is straightforward, and the simplest way to reconstruct a 3D model is through point sample rendering [34]. In these approaches, points were extended into circular or elliptical discs to improve 3D rendering quality [35, 36]. Most point-based methods relied on multi-view stereo to initialize the geometry. Neural Point Catacaustic was introduced to calculate point splatting, enabling the rendering of complex curved reflections [37]. Efforts were also made to reduce dependence on multi-view stereo. For example, Zhang et al. utilized initial masks and spherical harmonics to achieve view synthesis in radiance fields [38]. Additionally, 3D Gaussians were employed to represent human bodies [39] and other vision tasks [40] inspiring the development of 3DGS, which achieves real-time rendering without relying on multi-view stereo. The comparisons of the mentioned 3D reconstruction methods are listed in Table 1.

# 3 Methodology

## 3.1 Segment anything model 2

The Meta AI introduced the SAM for promptable segmentation, which was trained on a massive image data set [20]. This extensive training data set endowed SAM with significant potential for enhanced performance and accuracy compared to existing models while also reducing the need for task-specific modeling expertise, computational resources, and custom data annotation [41]. However, SAM operates on single images and lacks the capability to process videos. To address this limitation, Yang et al. [42] combined SAM with XMem [43] to enable video object tracking. Nevertheless, the effectiveness of this approach was constrained by the inherent limitations of both the SAM and XMem algorithms [8].

Following the original SAM, Meta AI proposed SAM 2 as a unified segmentation model capable of handling both image and video tasks. SAM 2 supports a wide range of prompts and retains the core features of its predecessor. Additionally, SAM 2 incorporates a memory module that stores previous object information and interactions. This memory mechanism enables SAM 2 to generate consistent predictions across video frames and refine these predictions using contextual information from previously observed frames. As illustrated in Fig. 1, this enhancement is achieved through an image encoder, memory attention module, memory encoder, and memory bank, which collectively store and transfer prediction and segmentation masks.

When processing a single frame of the video, SAM 2 utilizes the current prompt and the transferred memories to generate segmentation masks. For the entire video stream, the image decoder processes one frame at a time and retrieves the previous memories of the target segmentations. Simultaneously, optional prompts such as points, boxes, or other masks can be input into the mask decoder to refine the segmentation results for the current frame. After segmenting the current frame, the resulting mask is transferred and stored in the memory bank for use in the segmentation of the next frame. For frame $i$, SAM 2 takes the frame $F(i)$, optional prompts $P$, and the previous memories $M$ as inputs, producing the corresponding segmentation mask $S(i)$. This relationship can be expressed as

$$S(i) = \mathrm{SAM2}\left(F(i), P, M\right) \tag{1}$$

**Table 1** Comparison of different 3D reconstruction methods

| Comparison dimension | 3DGS | NeRF | SfM | Photogrammetry |
|---|---|---|---|---|
| Core principle | Explicit 3D Gaussian point clouds + differentiable rasterization | Implicit neural radiance fields (MLP-based density/color) + volume rendering | Sparse keypoint matching to recover camera poses + sparse 3D | Combines SfM + MVS with sensor data (e.g., drone imagery) |
| Input data | Sparse multi-view images or point clouds | Dense multi-view images with camera poses | Unordered multi-view images (no prior poses) | Multi-view images + sensor data (e.g., LiDAR and drone captures) |
| Output | Explicit Gaussian point cloud | Implicit volumetric field (requires ray marching for rendering) | Sparse point cloud + camera poses | Textured mesh/point cloud |
| Rendering speed | Fastest | Slow (0.98–2 FPS for 1024 × 1024 images) | No direct rendering | Slowest |
| Storage requirements | Compact (100 MB–1 GB for single object) | High (1–10 GB) | Low (generate sparse points only) | Extremely high (GB–TB raw data) |
| Data requirements | Low (works with sparse inputs) | High (requires dense image coverage) | Low (relies on feature matching) | High (requires massive raw data) |

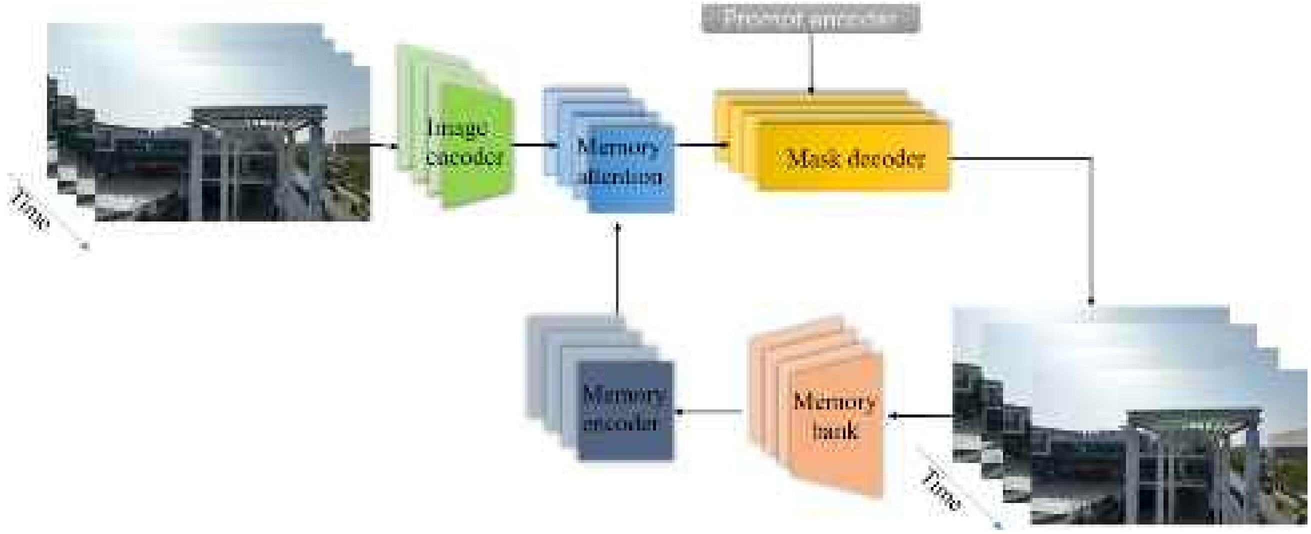


**Fig. 1** Architecture of SAM2

### 3.2 3DGS

Unlike widely used artificial neural networks, the 3DGS technique was developed to render 3D scenes by calculating and splitting 3D Gaussians [9]. Initially, SfM is employed to estimate camera poses and construct the point cloud of objects from images. Subsequently, 3D Gaussians are computed at each point in the cloud. These Gaussians are then refined using Stochastic Gradient Descent (SGD), with their density and quantity adaptively controlled based on gradients and specified criteria. For a point $i$, the 3D Gaussian can be represented as a function $g_i = [\mu_i, S_i, R_i, c_i, o_i]$ including the following parameters:

(1) location $\mu_i$ of the points in the cloud reconstructed from SfM as the centers of the Gaussians;
(2) covariance matrix $\Sigma_i = R_i S_i S_i^\top R_i^\top$ in which $S_i$ is the scaling matrix and $R_i$ is the rotation matrix;
(3) RGB color represented by spherical harmonic coefficients $c_i$;
(4) Opacity $o_i$ which controls the transparency of the Gaussian.

All 3D Gaussians collectively constitute the 3DGS map. Rendering begins by transforming the 3D Gaussians into camera coordinates. Using Max's volume rendering formula, Gaussians are rendered in the sort of their depth from front to back [44]. Max's volume rendering formula can be represented as

$$C(\hat{x}) = \sum_{i \in S} c_i q_i(\hat{x}) \prod_{j=1}^{i-1} \left(1 - q_j(\hat{x})\right) \quad (2)$$

where $C(\hat{x})$ represents the color after rendering for pixel $\hat{x}$ on the camera projection plane, $c_i$ is the spherical harmonic coefficients. All the Gaussians are concerned to calculate the weight due to the occlusion. Derived from the Gaussian kernel [45], the weight function $q_i$ is represented as

$$q_i(\hat{x}) = o_i \frac{1}{|J^{-1}||W^{-1}|} G_{\hat{\Sigma}_i^c}\left(\hat{x} - \hat{\mu}_i\right) \quad (3)$$

In this function, $G_{\hat{\Sigma}_i^c}$ is Gaussian function, $o_i$ is the Opacity, and $\hat{\mu} = [x_1, x_2]$ is obtained by taking the first two mean values of this Gaussian. After eliminating the last row and column of the matrix calculated in Eq. 4, the covariance matrix $\hat{\Sigma}_i^c$ is obtained:

$$\Sigma_i^c = JW\Sigma_i W^\top J^\top \quad (4)$$

where $J$ represents the Jacobian of the projection formula:

$$m(\mu) = K\left(\frac{W\mu}{(W\mu)_z}\right) \quad (5)$$

As illustrated in Fig. 2, the optimization process begins with the SfM point cloud, generating an initial set of 3D Gaussians. The density of these Gaussians is refined and controlled adaptively. The renderer of 3DGS supports real-time navigation across diverse scenes and employs a fast tile-based approach, achieving significantly higher training efficiency compared to other radiance field methods.

### 3.3 Proposed framework

The workflow illustrated in Fig. 3 demonstrates an integrated framework for 3D reconstruction and defect segmentation,

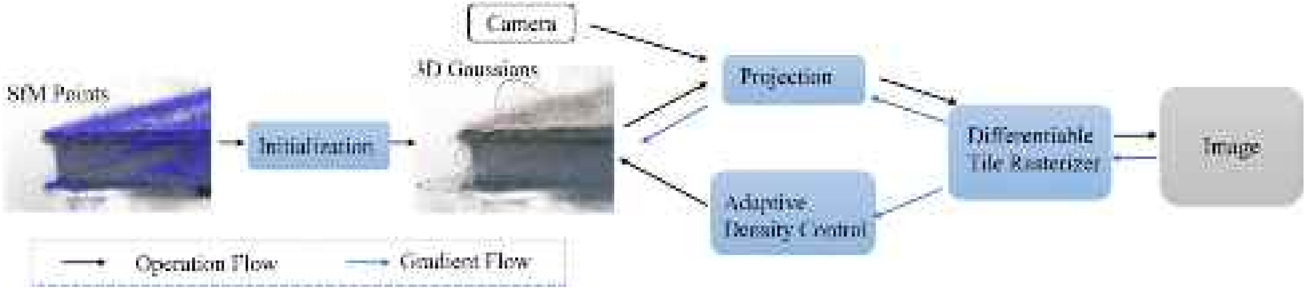


**Fig. 2** Optimization processes of the 3DGS

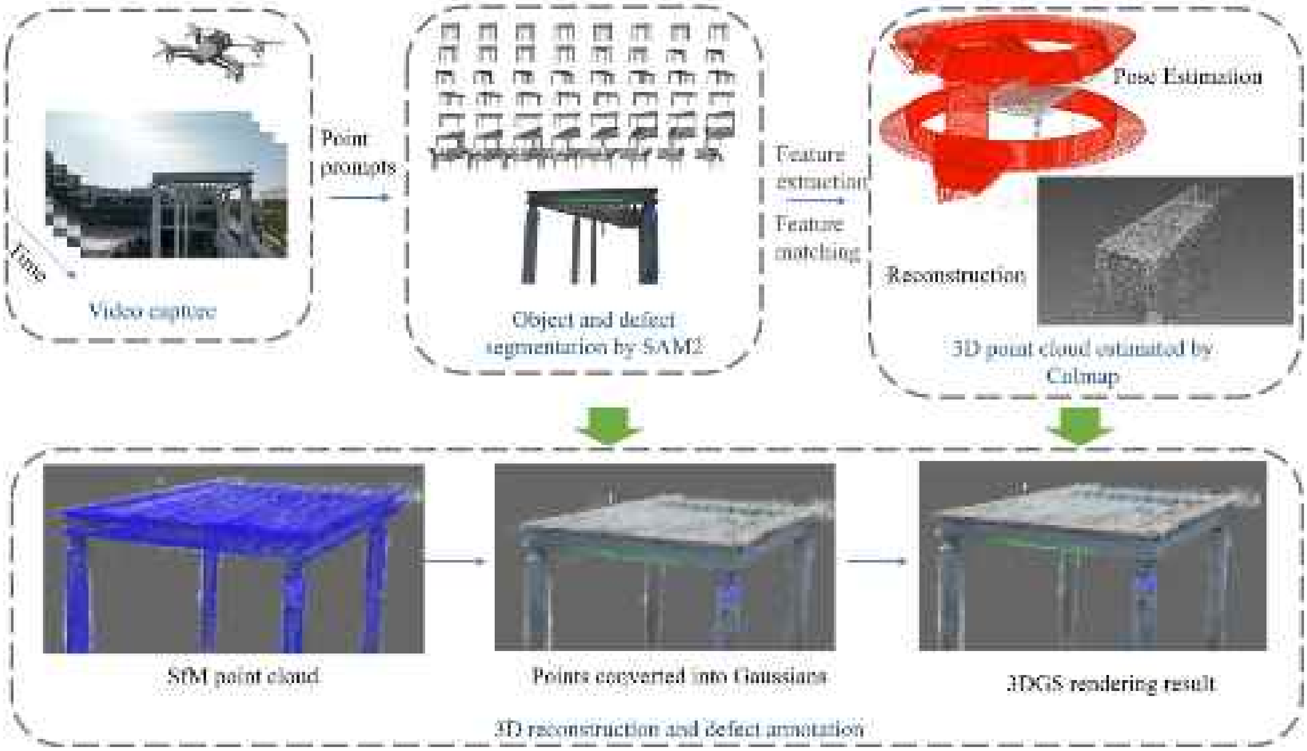

Time
Video capture
Point prompts
Object and defect segmentation by SAM2
Feature extraction
Feature matching
Pose Estimation
Reconstruction
3D point cloud obtained by Colmap
SfM point cloud
Points converted into Gaussians
3DGS rendering result
3D reconstruction and defect annotation

**Fig. 3** Workflow of the proposed framework

combining the capabilities of 3DGS and the SAM 2. The proposed framework is straightforward to implement and enables defect segmentation, tracking, 3D reconstruction, and defect visual highlighting using a platform equipped with a monocular camera.

The framework consists of two main components: defect segmentation and 3D reconstruction. First, SAM 2 is employed to segment the structure and defects in the monocular video. The SAM 2 segmentation process not only annotates defects in the video but also significantly reduces the manual effort required for 3D model segmentation. Next, COLMAP [24] with SfM generates a 3D point cloud of the structure from the labeled video. This point cloud provides the 3D geometry and global location information of both the structure and its defects. The estimated camera poses, 3D points, and processed video data are then used for 3DGS rendering.

In the 3DGS rendering process, 3D points with initial positions, colors, and sizes are converted into Gaussians, forming the foundation of the scene. The properties of each Gaussian—such as position, color, and covariance (which determines shape and spread)—are refined using Stochastic Gradient Descent. During optimization, these Gaussians are adaptively densified (to capture more detail) or pruned (to eliminate redundancies) based on gradient information and other criteria. The final output of the framework is a

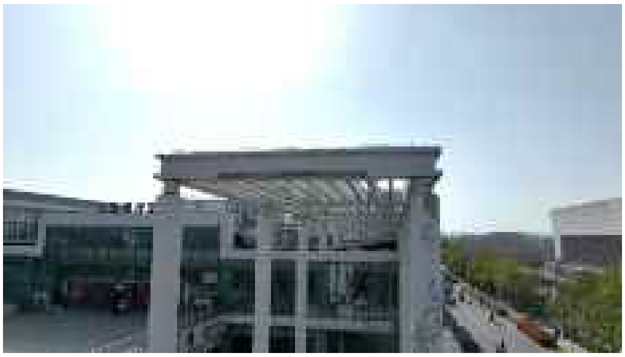


Fig. 4 Test concrete structure

high-fidelity 3D model with detailed surface representations and labeled defects.

In the next section, a validation study is conducted to evaluate the feasibility and accuracy of the proposed framework using a concrete structure as a case study.

## 4 Experimental validation

The test structure is a concrete post-and-beam construction with visible defects, as shown in Fig. 4. A 4K-resolution (4096 × 2160 pixels) video of the test structure was captured using a monocular camera mounted on an Unmanned Aerial Vehicle. The video was encoded at 30 fps with a bit rate of 130 Mbps. To test the robustness of the 3D reconstruction algorithm, the video was captured using average auto-exposure settings, with the metering point intentionally excluded from the main structure. The video frames encompass a wide range of shooting angles, distances, and lighting conditions, ensuring a comprehensive evaluation of the proposed method's applicability. To create a data set for image segmentation validation, one image was extracted every 10 frames from the video footage, resulting in a data set of 325 key frames. The flight trajectory of the UAV used for video recording is illustrated in Fig. 5, with the red rectangular box indicating the camera perspective from which the keyframes were extracted. Since the SfM method was utilized, the order of video frames does not affect the results. It is necessary to ensure sufficient image clarity and multi-angle coverage during capture. For running the segmentation models and 3D reconstruction, a computer platform equipped with an RTX 4090 GPU was utilized for the validation process.

### 4.1 Zero-shot promptable visual segmentation

Promptable segmentation methods, such as SAM and SAM 2, exhibit strong zero-shot transfer performance, enabling them to operate on unseen data without requiring additional training. In the validation, the collected images were used to evaluate the zero-shot generalization capabilities of these segment anything models. No custom images were used for additional training.

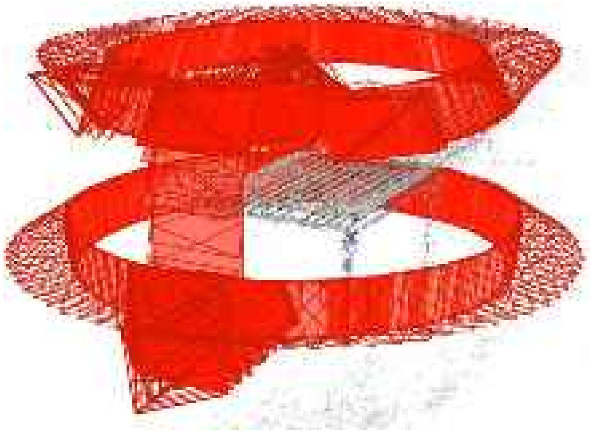


Fig. 5 Flight trajectory of the UAV

#### 4.1.1 Segmentation of single image

The two pre-trained models used in this section were Meta AI's largest models: SAM (sam_vit_huge) and SAM 2 (sam2_hiera_large). Nearly all images in the data set were captured with complex backgrounds, providing a rigorous test of the segmentation capabilities of the segment anything models. The structural segmentation results of the models under complex backgrounds were compared and analyzed.

For example, consider the first image in the data set. Using the same three-point prompts in the image, the following figures illustrate the comparison of segmentation results for the test structure. In Fig. 6a, the three prompts are marked in red on the original image. Compared to the manually labeled ground truth segmentation, SAM 2 demonstrated superior performance over SAM with the same prompts. SAM 2's segmentation exhibited better applicability across most of the unseen images.

As shown in Fig. 6b, the ground truth segmentation of the structure was manually labeled. In Fig. 6c, d, SAM exhibited over-segmentation issues, resulting in less precise segmentation outcomes. In contrast, SAM 2 provided enhanced segmentation accuracy, enabling more precise identification and segmentation of objects in most images.

Next, SAM and SAM 2 were utilized to segment multiple defects in the image. For each defect, a single point prompt was provided to guide the segmentation. The same prompts were used for both models to ensure a fair comparison. Figure 7 illustrates the comparison of image segmentation results for the test structure. In the figures, the segmented regions closely match the ground truth with sharp edges.

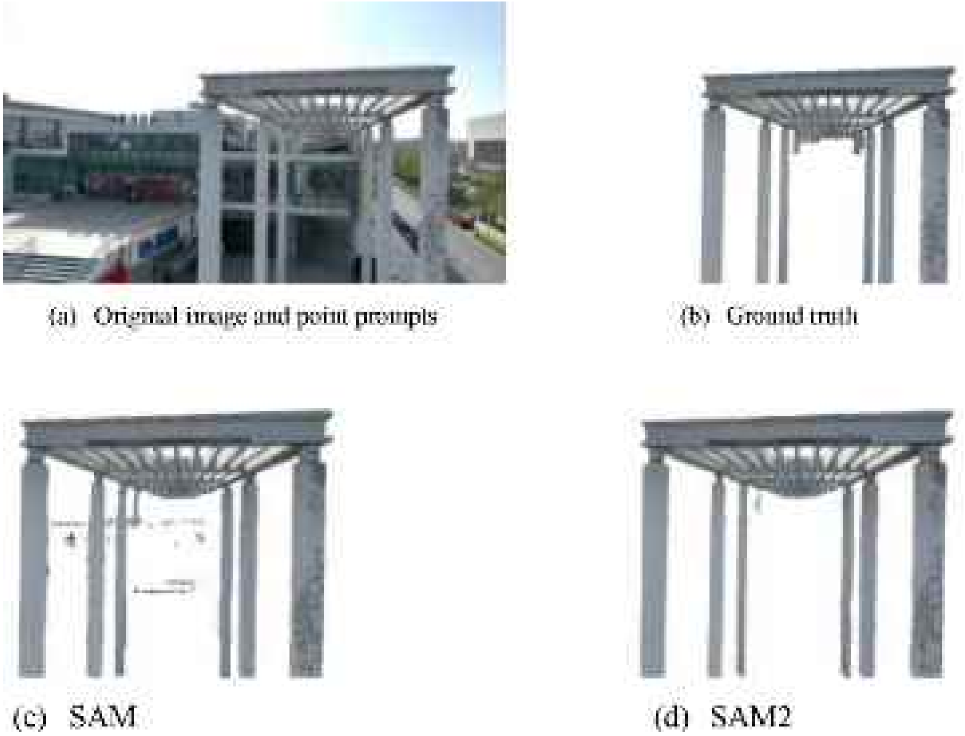


**Fig. 6** Segmentation of the structure

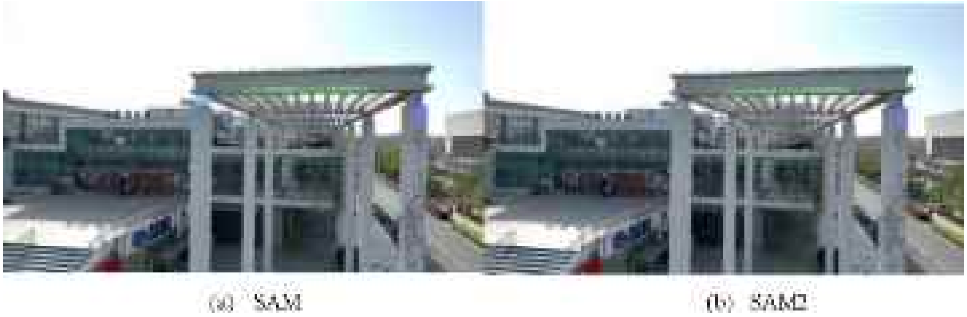


**Fig. 7** Defect segmentation

For the segmentation of tiny defects, SAM 2 demonstrated superior performance, whereas SAM mistakenly included a portion of the pillar component as part of the segmentation target. Across most images in the segmentation test, SAM 2 outperformed SAM.

To evaluate the performance quantitatively, segmentation accuracy was defined as follows:

$$A = \frac{P_r - P_w}{P_g} \times 100\% \tag{6}$$

where $P_r$ is the segmented pixels within the ground truth part, $P_w$ is the segmented pixels out of the ground truth part, $P_g$ is the pixels of the ground truth.

In the test data set, the segmentation accuracy for all images was calculated using the proposed accuracy function. The same point prompts were manually provided for each image. For the defects listed in the table, the number of prompts corresponds to the number of prompts per defect. The segmentation accuracy of SAM and SAM 2 is summarized in Table 2.

#### 4.1.2 Segmentation of image series

To evaluate the performance of promptable video segmentation, SAM 2 was compared with previous works on structure segmentation and defect segmentation. These evaluations were conducted on a zero-shot video data set using a three-click prompt in the key frame for structure segmentation. For defect segmentation, a single point prompt was provided for each defect in the key frame. Figure 8 illustrates the defect segmentation results in the key frame after applying the prompts using SAM 2.

After providing the prompts in the key frames, SAM 2 automatically tracked and segmented the targets across all frames in the video. Five defects were successfully labeled

**Table 2** Segmentation accuracy of SAM and SAM 2

| Prompt | SAM | | | SAM2 | | |
|---|---|---|---|---|---|---|
| | One-point prompt (%) | Two-point prompts (%) | Three-point prompts (%) | One-point prompt (%) | Two-point prompts (%) | Three-point prompts (%) |
| Structure segmentation accuracy | 62.6 | 78.3 | 82.6 | 69.7 | 91.8 | 93.1 |
| Defect segmentation accuracy | 76.4 | 73.1 | 78.7 | 82.5 | 83.1 | 87.3 |

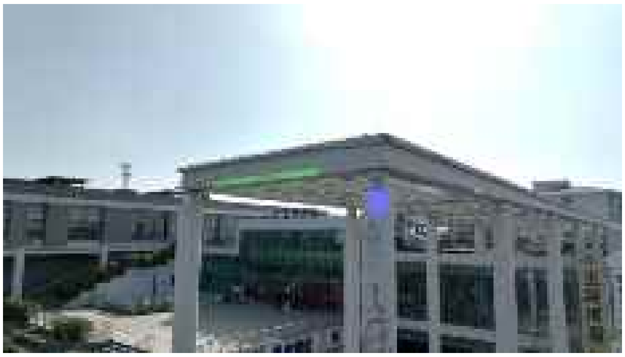


**Fig. 8** Defects segmentation in the key frame

and segmented, as shown in Fig. 9a, c. However, in Fig. 9b, d, SAM 2 lost track of the targets, and the defects were not consistently tracked. This issue occurred, because the camera movement caused the target defects to move out of view, leading to tracking failures in some instances.

To further evaluate the performance of SAM 2, two methods combined SAM with XMem++ [46] or Cutie [47] were tested as baselines for video segmentation tasks. XMem++ and Cutie are advanced technologies for tracking and segmenting objects in videos. SAM was used to provide prompted segmentation masks to XMem++ and Cutie for further video processing. In the first method, XMem++ utilized masks as input on one or multiple frames for video segmentation. In the second method, Cutie was modified to accept SAM segmentation masks as inputs on multiple frames.

The standard $\mathcal{J}$ and $\mathcal{F}$ metric [48] was took to evaluate the segmentation accuracy. The metric includes region similarity ($\mathcal{J}$) and boundary accuracy ($\mathcal{F}$), as proposed in DAVIS 2016[49].Given a segmentation mask $M$ and the

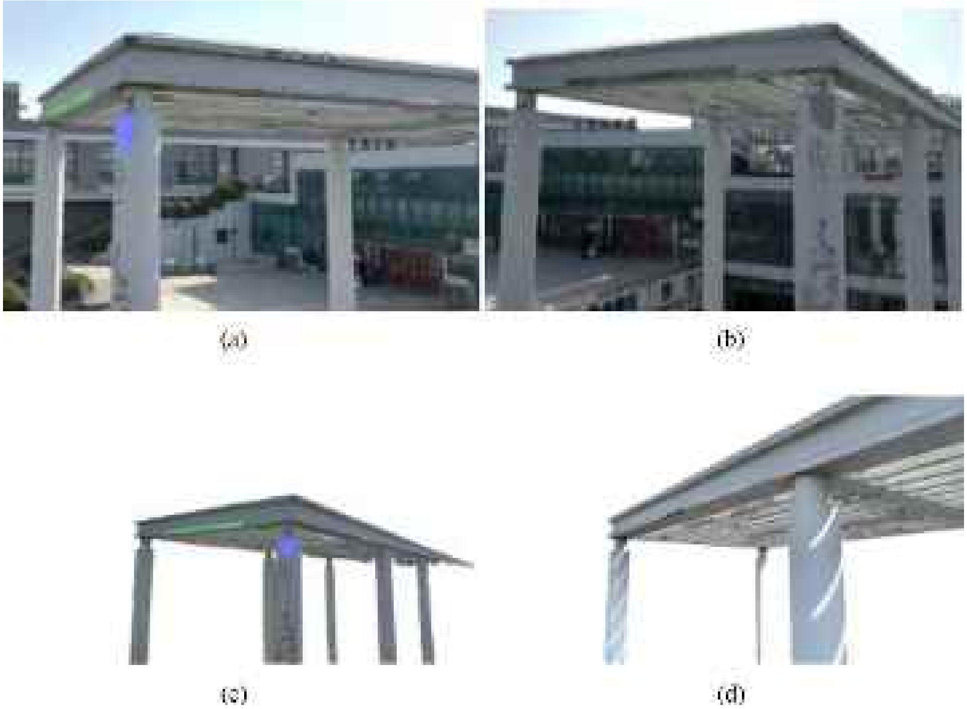


**Fig. 9** Key frame segmentation with point prompts

corresponding ground truth mask $G$, so that region similarity $\mathcal{J}$ was defined as

$$\mathcal{J} = \frac{|M \cap G|}{|M \cup G|} \tag{7}$$

where $|M \cap G|$ represents the intersection of the segmentation mask and the ground truth, and $|M \cup G|$ denotes their union.

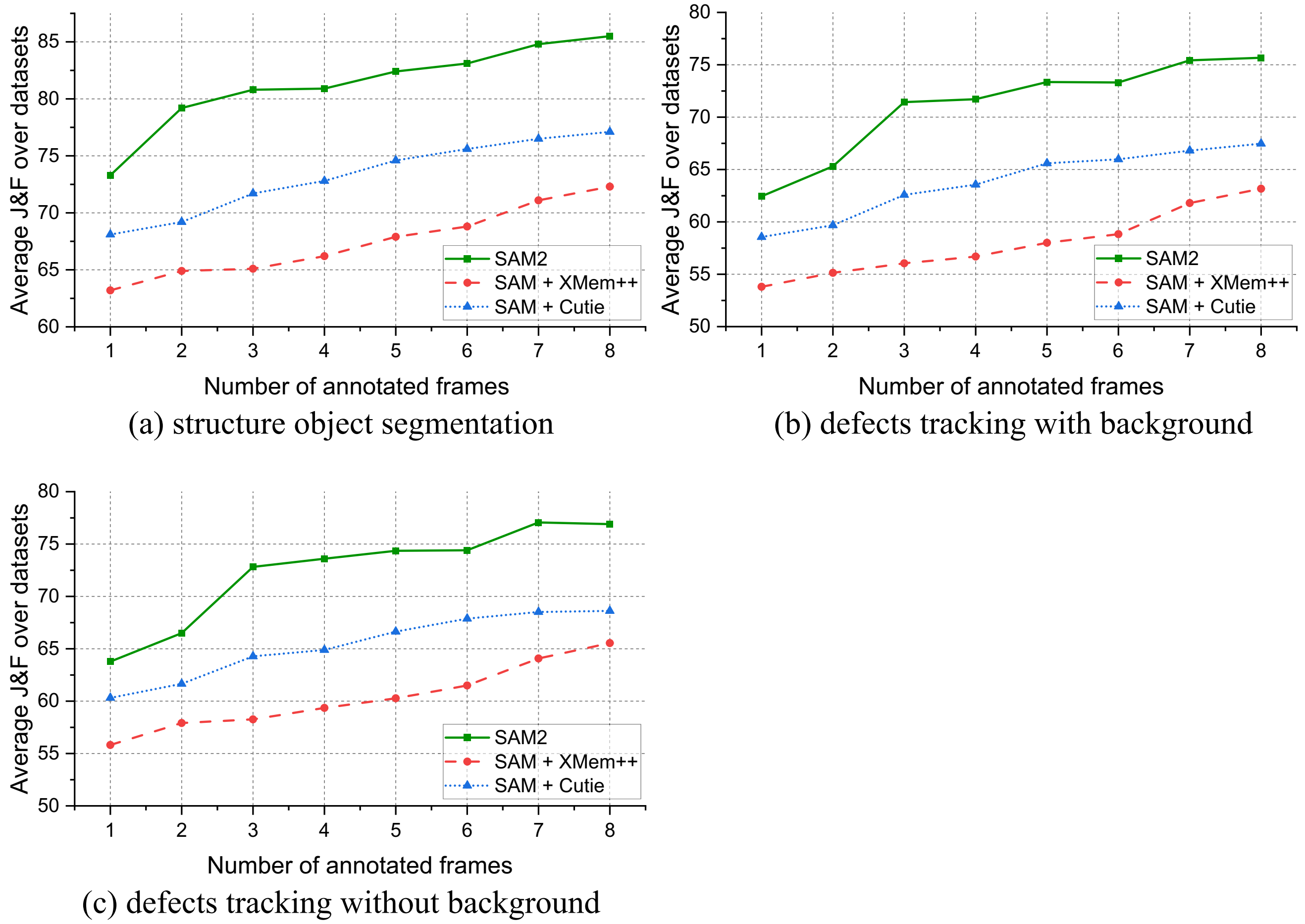
For contour accuracy, the closed contours of the segmentation mask and the ground truth are defined as $c(M)$ and $c(G)$, respectively. Based on this, the contour-based precision ($Pc$) and recall ($Rc$) between the contour points of $c(M)$ and $c(G)$ can be calculated. The contour accuracy $\mathcal{F}$ is defined as

$$\mathcal{F} = \frac{2P_cR_c}{P_c + R_c} \tag{8}$$

As shown in Fig. 10, the average $\mathcal{J}$ and $\mathcal{F}$ accuracy of the video segmentations was calculated across cases with interacted frames ranging from 1 to 8. SAM 2 demonstrated superior performance compared to SAM with XMem + +and Cutie for both structural segmentation and defect segmentation. SAM 2 achieved more accurate video segmentation with fewer prompts while maintaining the ability to refine results during the segmentation process. Overall, SAM 2 exhibited better performance and produced higher quality results with fewer prompted frames.

Figure 10a shows the average $\mathcal{J}$ and $\mathcal{F}$ values for structure object segmentation across the data set. Figure 10b, c presents the average $\mathcal{J}$ and $\mathcal{F}$ values for defect tracking in the image data sets with and without background, respectively.

Based on Meta AI's research and the validation conducted in this section, SAM 2 may lose focus on target objects when the shooting position changes too rapidly or the scene is overly crowded [8]. To improve segmentation accuracy, additional prompts can be added to frames throughout the video to mitigate this issue. During the defect tracking tests, SAM 2 lost track of targets multiple times across video frames. Although additional manual prompts were provided for frames in the middle of the video, the



(a) structure object segmentation

(b) defects tracking with background

(c) defects tracking without background

**Fig. 10** $\mathcal{J}$ and $\mathcal{F}$ accuracy of the video object tracking defect tracking results were not flawless. This performance limitation may be attributed to rapid changes in shooting perspective and the presence of nearby objects with similar appearances. Despite some mis-segmentations in certain frames, the processed videos remained sufficiently accurate for 3D reconstruction. The validation of 3D reconstruction and rendering is presented in the next section.

## 4.2 3D reconstruction

The video processed in the previous section was used for 3D reconstruction and rendering. The validation included separate reconstructions of the original video and the processed video. This comparison aimed to analyze the influence of scene background on 3D model reconstruction.

### 4.2.1 The 3D reconstruction from the original video

The first step of 3D reconstruction was recovering a 3D point cloud from images using SfM technique which can be implemented using libraries such as COLMAP [50, 51]. When using the original video frames, COLMAP extracted key points and their descriptors from each image and then matched these features between images to identify corresponding points across different views. Part of the input images is shown in Fig. 11, and the recovered 3D point cloud is illustrated in Fig. 12.

The point cloud was converted into 3D Gaussians which encapsulated position information, color information, and a covariance matrix that describes the shape of the point cloud distribution. This conversion prepared the point cloud data for rasterization. From the SfM data, the position and color of each Gaussian were obtained. Training was then performed for each Gaussian function to estimate the optimal positions, colors, covariance, and transparency.

Stochastic Gradient Descent was employed for the training process. All currently differentiable Gaussian functions were used for the rendering. The training loss was determined by measuring the discrepancy between the rendered output and the ground truth data. Subsequently, the parameters of each Gaussian were optimized according to the calculated loss values. The density of points in the relevant Gaussian was dynamically controlled based on the scene requirements. If the alpha value of a Gaussian was too low, it was removed to ensure that the remaining Gaussians better fit fine details and eliminate unnecessary elements.

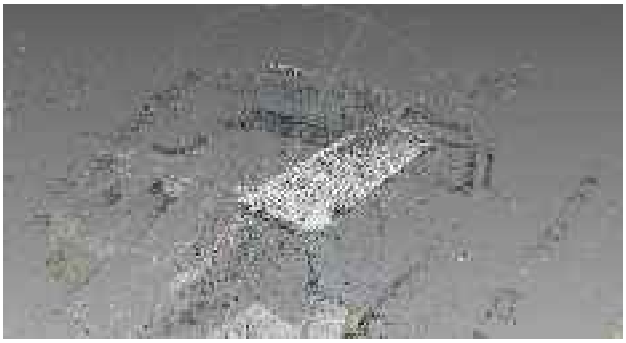
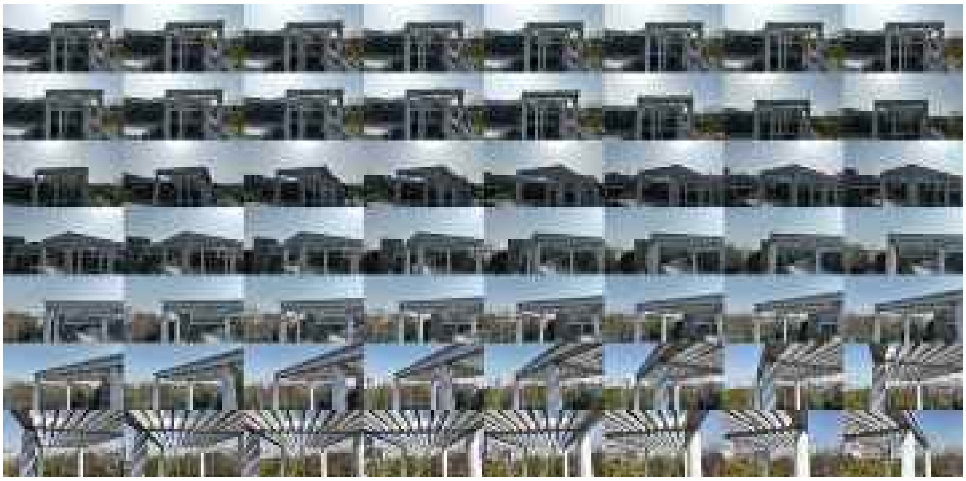


**Fig. 12** 3D point cloud recovered from the original video image series

Figure 13a shows the rendering result of the 3DGS. In the initial step, COLMAP detected a set of local appearance feature descriptors at specific locations. These features remained invariant under variations in lighting and geometry. COLMAP extracted feature points and automatically matched them between adjacent images. These feature points corresponded to different depths of field. Based on the feature point cloud, the final 3DGS rendering result was a panoramic depth scene. To segment the target object from the panoramic scene, 3D Gaussian editors or 3D promptable segmentation methods[52, 53] could be used for manual intervention. Figure 13b shows the processed 3D model after manual modification.

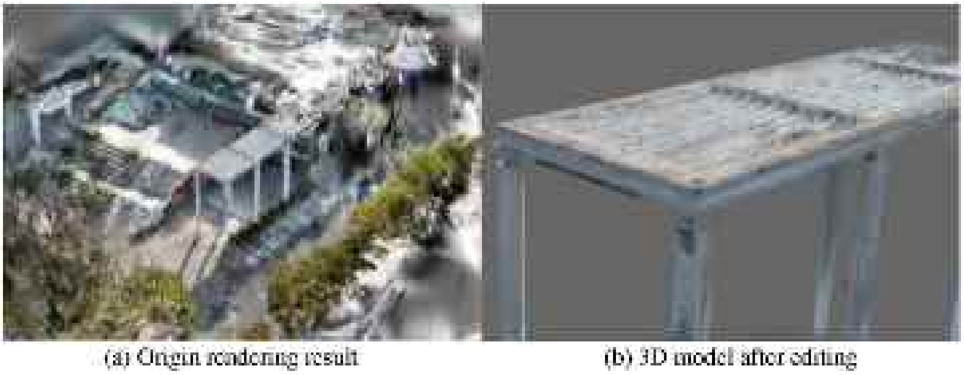
**Fig. 11** Input images for 3D reconstruction



#### 4.2.2 The 3D reconstruction from the segmented video

Segmenting the target object after 3DGS rendering can require significant manual effort and waste rendering processing resources. As demonstrated in Sect. 4.1, SAM 2 can effectively segment the target structure from the video, which can then be used for 3D reconstruction and rendering. Examples of input images segmented by SAM 2 are shown in Fig. 14. The point cloud in Fig. 15a was generated using COLMAP from the SAM 2-processed images, as described in Sect. 4.1. Most of the point cloud consists of feature points from the target structure. Using the SfM point cloud data, the 3DGS model was rendered, resulting in the final 3D model shown in Fig. 15b.

The workload for feature point extraction and 3DGS rendering was reduced due to the elimination of the background


(a) Origin rendering result

(b) 3D model after editing

**Fig. 13** Rendering result of the 3DGS

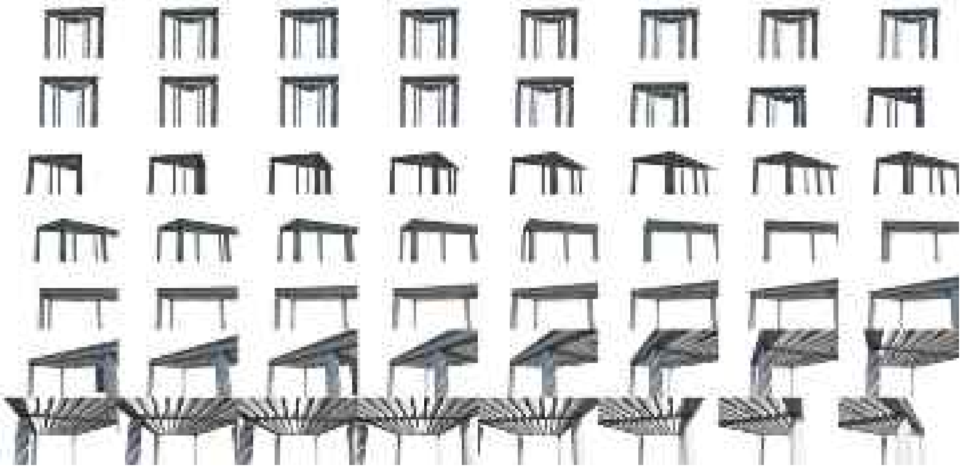
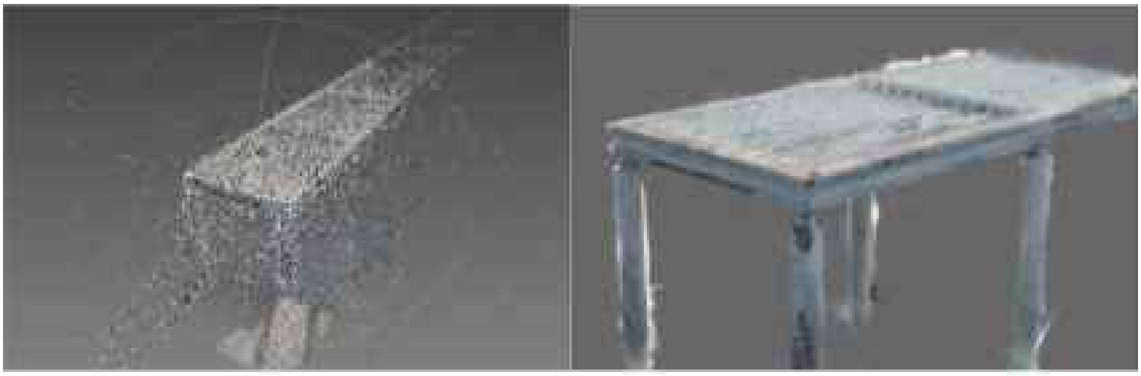


**Fig. 14** Input images segmented by SAM2 for 3D reconstruction


(a) Point cloud

(b)3DGS rendering

**Fig. 15** 3D reconstruction result

scene in the images. While achieving nearly identical rendering results, the proposed process reduced the rendering file size by 80%, rendering memory consumption by 50%, and rendering time by 30% in this test. However, white Gaussian splatting occurred at the edges of the object. This phenomenon is attributed to background area filling after segmentation.

### 4.3 3D reconstruction with defect label

As presented in Sect. 4.1.2, defects were labeled in every image. Two data sets—with and without background—were used for 3D reconstruction and rendering. To facilitate clearer observation, the 3D model reconstructed from images with a background scene was manually edited to remove unnecessary objects. Figure 16a illustrates the 3D model reconstructed from the video with a background, while Fig. 16b shows the 3D model reconstructed from the video without a background. The 3D reconstruction and label of defects appeared visually similar in both results. However, due to the white background in the processed images, the 3D model reconstructed by the proposed framework exhibited white edges, which were noticeable from certain viewing angles.

To ensure consistent comparisons when calculating error metrics, the peak signal-to-noise ratio (PSNR), structural similarity index measure (SSIM), and learned perceptual image patch similarity (LPIPS) which were the most frequently cited in the literatures were employed to measure accuracy.

The PSNR is an error metric derived from Mean Squared Error (MSE), which calculates the average squared difference between the original and rendered pixel values. The formula for PSNR is

$$\mathrm{PSNR} = 10 \cdot \log_{10}\left(\frac{\mathrm{MAX}_G^2}{\mathrm{MSE}}\right) \tag{9}$$

where $\mathrm{MAX}_G^2$ represents the maximum pixel value in the ground truth image. A higher PSNR value indicates that the rendered image is more similar to the original image. However, small visual distortions that are easily noticeable to humans may still result in a high PSNR score. Therefore, perceptual metrics such as the *SSIM* are often used in conjunction with PSNR to provide a more human-centered evaluation of image quality.

The SSIM is another metric that describes the similarity between two images [54]. It focuses on the structural similarity of the images, aligning more closely with human visual perception than simpler metrics like PSNR. SSIM is widely adopted in image and video processing tasks as a method to evaluate rendering quality compared to the ground truth.

SSIM primarily considers three key features of images: luminance ($L$), contrast($C$), and structure($S$). Between image $x$ and image $y$, the functions of the three features are as follows:

$$L(x,y) = \frac{2\mu_x\mu_y + C_1}{u_x^2 + \mu_y^2 + C_1} \tag{10}$$

$$C(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \tag{11}$$

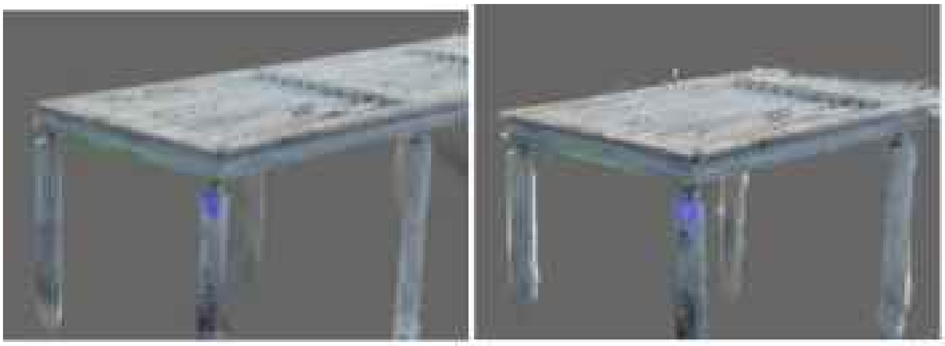
$$S(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_1} \tag{12}$$

where $\mu_x$ and $\mu_y$ are the mean luminance values of image $x$ and image $y$, $\sigma_x$ and $\sigma_y$ are their variances, and $\sigma_{xy}$ is the covariance, which measures how the structures of images $x$ and $y$ align. $C_1$, $C_2$ and $C_3$ are constants to keep the functions stable.

These three components constitute SSIM index:

$$\mathrm{SSIM}(x,y) = [L(x,y)]^\alpha \cdot [C(x,y)]^\beta \cdot [S(x,y)]^\gamma \tag{13}$$

where, $\alpha$, $\beta$ and $\gamma$ are weight factors, typically set to 1, indicating equal importance for all three components. SSIM value is in the range of 0–1, and higher values indicating that the generated image is closer to the ground truth.



(a) 3D model reconstructed from whole images (b) 3D model reconstructed from images without background

**Fig. 16** 3D reconstruction results

The LPIPS between two images $x$ and $y$ is defined as follows:

$$\mathrm{LPIPS}(x, y) = w \parallel f(x) - f(y) \parallel_2^2 \tag{14}$$

where $w$ represents learned weights, and $f(x)$ and $f(y)$ are feature maps extracted from images $x$ and $y$, respectively. LPIPS applies learned weights to the feature differences, with representing the Euclidean distance between the feature maps.

To further evaluate the performance of 3DGS, comparisons were made with now available Nerf-based approaches, including Instant-NGP [55], Defacto [56] and Mip-NeRF [57]. The 3D reconstruction was implemented using the Nerfstudio [56] platform with the same data sets. Mip-NeRF improves anti-aliasing by modeling pixel footprints with cone sampling instead of single rays, ensuring crisp multi-scale renderings at the cost of slow training. In contrast, Instant-NGP prioritizes speed via hash grids and tiny neural networks, achieving fast training. 3DGS can achieve fast rendering using explicit Gaussian splats that are parallelized on GPU, avoiding neural network computations, and its training is slower, because it dynamically optimizes Gaussian positions, shapes, and densities through iterative gradient updates and scene refinement. Defacto is a combination of many published methods proposed by Nerfstudio platform.

The results are summarized in Table 3. All time values in the table were obtained from local runs on an RTX 4090 GPU. For clarity, the data set consisting of original images with defect labels is referred to as Data Set 1, while the data set with background removed and defect labels is referred to as Data Set 2.

The table demonstrates that 3DGS performed well on both data sets. Mip-NeRF is the slowest and Instant-NGP is the fastest in the validation. The Mip-NeRF method required an average training time of 15 h, compared to 3DGS's 12–15 min. 3DGS achieves the highest PSNR and SSIM across two data sets, indicating superior reconstruction accuracy and structural fidelity compared to Instant-NGP, Defacto, and Mip-NeRF. 3DGS's LPIPS scores (0.037/0.035) confirm better perceptual alignment with ground truth than other methods, which suffer from blur. While 3DGS trains slightly slower than Instant-NGP (15m/12m vs. 10m/11m), it has the best fidelity. Besides, 3DGS is 60 × faster than Mip-NeRF (15h/14h) and renders 150/127 FPS—orders of magnitude faster than NeRF-based methods (0.1–2.6 FPS). This highlights 3DGS's optimal balance of quality, speed, and practicality. Overall, 3DGS achieved comparable quality to Mip-NeRF and Defacto while reducing time consumption.

## 5 Discussion

### 5.1 Merits of the proposed framework

This paper presents an integrated framework combining SAM 2 and 3DGS for efficient segmentation of structural defects and 3D reconstruction. The proposed framework demonstrates improvements in segmentation precision and rendering quality through a three-step process: (1) utilizing SAM 2 for background elimination and defect segmentation in videos; (2) Extracting and matching features from the processed video frames and employing SfM for initial pose estimation and 3D point cloud generation;(3) Converting points into 3D Gaussians representation and refining these Gaussians to achieve adaptive densification. These refinements contribute to enhancing defect detection and surface representation accuracy, as outlined below:

(1) Improvements in defect tracking and segmentation

The proposed framework leverages SAM 2 to achieve precise defect tracking and segmentation. SAM 2's built-in memory mechanism, which stores information about each defect across frames, enables consistent segmentation of multiple defects throughout the video. Even in cases of view interruption or viewpoint changes, SAM 2 demonstrates robust performance.

(2) Enhancements in 3D rendering efficiency

Using a fast tile-based renderer for Gaussians, the proposed framework optimizes computational workload, achieving competitive training speeds compared to radiance field approaches. 3DGS supports real-time navigation across diverse scenes, making it suitable for large-scale inspections,

**Table 3** Quantitative evaluation of 3DGS

| | Data Set 1 | | | | | Data Set 2 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | PSNR | SSIM | LPIPS | Training time | FPS | PSNR | SSIM | LPIPS | Training time | FPS |
| Instant-NGP | 19.661 | 0.758 | 0.229 | 10m14s | 2.225 | 21.419 | 0.878 | 0.132 | 11m37s | 0.816 |
| Defacto | 17.330 | 0.568 | 0.243 | 18m47s | 2.587 | 19.064 | 0.790 | 0.169 | 17m55s | 2.636 |
| Mip-NeRF | 28.362 | 0.913 | 0.103 | 15 h | 0.110 | 27.712 | 0.823 | 0.117 | 14 h | 0.110 |
| 3DGS | 28.696 | 0.967 | 0.037 | 15m23s | 150.117 | 28.661 | 0.967 | 0.035 | 12m18s | 127.367 |

where rapid rendering and accuracy are critical. This framework provides substantial benefits in managing the reconstruction pipeline, balancing speed with high detail quality for practical applications.

(3) Improvements in defect 3D localization

The proposed framework integrates SAM 2's segmentation capabilities with 3DGS, enabling precise surface defect localization in 3D models. By leveraging SAM 2's promptable mask decoder, this framework provides reliable multi-object tracking and segmentation across video frames. The combination labels and locates defects on both 2D images and 3D models, offering a novel method for 3D defect localization.

### 5.2 Limitations and future works

While this paper presents encouraging results through the combination of SAM 2 and 3DGS for defect segmentation and 3D reconstruction, several limitations warrant further research:

(1) Limited depth perception

While effective for surface-level segmentation, the current framework does not capture internal contours or depths of defects, such as cracks. Addressing this limitation might involve integrating infrared thermography or alternative imaging techniques with Simultaneous Localization and Mapping to capture sub-surface details.

(2) Tracking challenges in complex scenes

SAM 2 may struggle to maintain object tracking through shot changes, highly occluded scenes, and longer video sequences, leading to occasional misidentifications or missed detections. While these tracking limitations have minimal impact on routine inspections, they could affect the analysis of critical structural components. In such cases, the proposed framework may serve as an initial assessment tool, while more advanced inspection frameworks with multi-sensor integration could be applied for safety–critical evaluations.

(3) Incomplete point clouds in challenging conditions

Due to factors, such as low-texture surfaces, occlusions, and suboptimal lighting, the 3D reconstruction process may produce incomplete point clouds with missing sections. This issue could be mitigated in practical applications by tracking sparse points in real time, which would help identify regions requiring additional scans, particularly in texture-poor areas. Future studies should aim to enhance reconstruction completeness in these challenging regions to improve the method's robustness for inspections.

(4) Defect recognition and segmentation across diverse concrete structures

While the proposed framework demonstrates fine performance on flat or moderately curved surfaces with limited weathering, its generalizability to highly complex curved surfaces with irregular geometries or severely degraded concrete remains unverified. Highly curved regions may require adaptive Gaussian density adjustments beyond current implementation, while heavily weathered textures could degrade segmentation and reconstruction fidelity. Future work should validate the method on a broader diversity of structures, including extreme curvature and material degradation scenarios.

In the future, this framework could be deployed on Unmanned Aerial Vehicles or robotic platforms equipped with path-planning algorithms to achieve fully autonomous inspections. This would further enhance the applicability and versatility of the method for structural analysis and defect detection.

## 6 Conclusions

By integrating 3DGS with SAM2 method, a novel framework is presented to segment and locate defects and 3D reconstruct. Leveraging SAM2 for precise object segmentation, irrelevant background elements are effectively masked, allowing the focus to remain on the structural surfaces of interest. This segmentation serves as the foundation for the proposed framework, enabling the localization and enhanced visibility of defects the combination of these techniques facilitates the identification of fine structural irregularities while maintaining high rendering quality.

Two primary advancements were achieved through this method: (1) enhanced defect segmentation was achieved through SAM2's ability to generalize masks in zero-shot tasks while maintaining temporal consistency in video-based segmentation. (2) The implementation of 3DGS for high-fidelity rendering, which optimizes both the visual representation of rendered surfaces and the precision of defect localization in 3D. This method ensures not only the detection of defects at close range but also the localization and recognition of defects in complex scenes, resulting in more reliable defect recognition.

Furthermore, experimental comparisons demonstrate that this method outperforms other 3D reconstruction techniques in both efficiency and defect detection accuracy. The proposed framework's ability to locate structural defects is particularly valuable for applications in fields, such as architectural preservation, industrial inspection, and cultural heritage conservation. By combining advanced segmentation and rendering techniques, this paper offers a new perspective on the integration of deep learning-driven segmentation with 3D rendering for defect detection in 3D reconstructions. Ultimately, this research demonstrates the potential for accurate 3D reconstruction and defect localization, contributing valuable insights to the fields of digital reconstruction.

**Acknowledgements** Financial support to complete this study was provided in part by the National Natural Science Foundation of China (Grant No. 52278301) and the Hefei University Talent Research Fund Project (Grant No. 24RC23).

**Funding** National Natural Science Foundation of China, 52278301, Zuocai Wang, Hefei University Talent Research Fund Project, 24RC23, DAYOU DUAN.

**Data availability** The data used to support the findings of this study will be available from the corresponding author upon reasonable request.

## Declarations

**Conflicts of interest** The authors declare that they have no conflicts of interest.

## References

1. Zhu W, Zhang H, Eastwood J, Qi X, Jia J, Cao Y (2023) Concrete crack detection using lightweight attention feature fusion single shot multibox detector. Knowl-Based Syst 261:110216
2. Yang X, Li H, Yu Y, Luo X, Huang T, Yang X (2018) Automatic pixel-level crack detection and measurement using fully convolutional network. Computer-Aided Civ Infrastr Eng 33(12):1090–1109
3. Kim B, Cho S (2019) Image-based concrete crack assessment using mask and region-based convolutional neural network. Struct Control Health Monit 26(8):e2381
4. Deng L, Sun T, Yang L, Cao R (2023) Binocular video-based 3D reconstruction and length quantification of cracks in concrete structures. Autom Constr 148:104743
5. Ai D, Jiang G, Lam S-K, He P, Li C (2023) Computer vision framework for crack detection of civil infrastructure—a review. Eng Appl Artif Intell 117:105478
6. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444
7. Sun H, Song L, Yu Z (2023) A deep learning-based bridge damage detection and localization method. Mech Syst Signal Process 193:110277
8. Ravi N et al (2024) SAM 2: segment anything in images and videos. arXiv:2408.00714
9. Kerbl B, Kopanas G, Leimkühler T, Drettakis G (2023) 3D Gaussian splatting for real-time radiance field rendering. ACM Trans Graph 42(4)
10. Xiong B et al (2024) DefNet: A multi-scale dual-encoding fusion network aggregating transformer and CNN for crack segmentation. Constr Build Mater 448:138206
11. Liang J, Gu X, Jiang D, Zhang Q (2024) CNN-based network with multi-scale context feature and attention mechanism for automatic pavement crack segmentation. Autom Constr 164:105482
12. Ren Y et al (2020) Image-based concrete crack detection in tunnels using deep fully convolutional networks. Constr Build Mater 234:117367
13. Han X, Cheng Q, Chen Q, Chen L, Liu P (2024) Deep learning-based multi-category disease semantic image segmentation detection for concrete structures using the Res-Unet model. J Civ Struct Health Monit: 1–12
14. Dosovitskiy A et al (2020) An image is worth 16x16 words: transformers for image recognition at scale. arXiv:2010.11929
15. Xie E, Wang W, Yu Z, Anandkumar A, Alvarez JM, Luo P (2021) SegFormer: simple and efficient design for semantic segmentation with transformers. Adv Neural Inf Process Syst 34:12077–12090
16. Wang Z, Leng Z, Zhang Z (2024) A weakly-supervised transformer-based hybrid network with multi-attention for pavement crack detection. Constr Build Mater 411:134134
17. Zhang H, Ma L, Yuan Z, Liu H (2024) Enhanced concrete crack detection and proactive safety warning based on I-ST-UNet model. Autom Constr 166:105612
18. Iglovikov V, Shvets A (2018) Ternausnet: U-net with vgg11 encoder pre-trained on imagenet for image segmentation. arXiv:1801.05746
19. Narayanan D et al (2021) Efficient large-scale language model training on gpu clusters using megatron-lm. In: Proceedings of the international conference for high performance computing, networking, storage and analysis, pp 1–15
20. Kirillov A et al (2023) Segment anything. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 4015–4026
21. Hanrahan MLP (1996) Light field rendering. SIGGRAPH96, computer graphics proceeding
22. Szeliski R, Gortler S, Grzeszczuk R, Cohen MF (1996) The lumigraph. In: Proceedings of the 23rd annual conference on computer graphics and interactive techniques (SIGGRAPH 1996), pp 43–54
23. Buehler C, Bosse M, McMillan L, Gortler S, Cohen M (2001) Unstructured lumigraph rendering. In: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, pp 425–432
24. Schönberger JL, Frahm JM (2016) Structure-from-motion revisited. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR), 27–30 June 2016, pp 4104–4113. https://doi.org/10.1109/CVPR.2016.445
25. Goesele M, Snavely N, Curless B, Hoppe H, Seitz SM (2007) Multi-view stereo for community photo collections. In: 2007 IEEE 11th international conference on computer vision. IEEE, pp 1–8
26. Eisemann M et al (2008) Floating textures. In: Computer graphics forum, vol 27, no 2. Wiley Online Library, pp 409–418
27. Hedman P, Philip J, Price T, Frahm J-M, Drettakis G, Brostow G (2018) Deep blending for free-viewpoint image-based rendering. ACM Trans Graph (ToG) 37(6):1–15
28. Kopanas G, Philip J, Leimkühler T, Drettakis G (2021) Point-based neural rendering with per-view optimization. Comput Graph Forum 40(4):29–43
29. Tewari A et al (2022) Advances in neural rendering. In: Computer graphics forum, vol 41, no 2. Wiley Online Library, pp. 703–735
30. Penner E, Zhang L (2017) Soft 3d reconstruction for view synthesis. ACM Trans Graph (TOG) 36(6):1–11
31. Henzler P, Mitra NJ, Ritschel T (2019) Escaping plato's cave: 3D shape from adversarial rendering. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 9984–9993
32. Sitzmann V, Thies J, Heide F, Nießner M, Wetzstein G, Zollhofer M (2019) Deepvoxels: learning persistent 3D feature embeddings. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 2437–2446
33. Mildenhall B, Srinivasan PP, Tancik M, Barron JT, Ramamoorthi R, Ng R (2021) Nerf: representing scenes as neural radiance fields for view synthesis. Commun ACM 65(1):99–106

34. Grossman J, Dally WJ (1998) Point sample rendering by JP Grossman. Signature
35. Botsch M, Hornung A, Zwicker M, Kobbelt L (2005) High-quality surface splatting on today's GPUs. In: Proceedings eurographics/IEEE VGTC symposium point-based graphics. IEEE, pp 17–141
36. Ren L, Pfister H, Zwicker M (2002) Object space EWA surface splatting: a hardware accelerated approach to high quality point rendering. Comput Graph Forum 21(3):461–470
37. Kopanas G, Leimkühler T, Rainer G, Jambon C, Drettakis G (2022) Neural point catacaustics for novel-view synthesis of reflections. ACM Trans Graph (TOG) 41(6):1–15
38. Zhang Q, Baek S-H, Rusinkiewicz S, Heide F (2022) Differentiable point-based radiance fields for efficient view synthesis. In: SIGGRAPH Asia 2022 conference papers, pp 1–12
39. Rhodin H, Robertini N, Richardt C, Seidel H-P, Theobalt C (2015) A versatile scene model with differentiable visibility applied to generative pose estimation. In: Proceedings of the IEEE international conference on computer vision, pp 765–773
40. Angtian Wang PW, Sun J, Kortylewski A, Yuille A (2023) Voge: a differentiable volume renderer using gaussian ellipsoids for analysis-by-synthesis. In: The eleventh international conference on learning representations
41. Ahmadi M, Lonbar AG, Sharifi A, Beris AT, Nouri M, Javidi AS (2023) Application of segment anything model for civil infrastructure defect assessment. arXiv:2304.12600
42. Yang J, Gao M, Li Z, Gao S, Wang F, Zheng F (2023) Track anything: segment anything meets videos. arXiv:2304.11968
43. Cheng HK, Schwing AG (2022) Xmem: long-term video object segmentation with an Atkinson-Shiffrin memory model. In: European conference on computer vision. Springer, pp 640–658
44. Max N (1995) Optical models for direct volume rendering. IEEE Trans Visual Comput Graphics 1(2):99–108
45. Zwicker M, Pfister H, Van Baar J, Gross M (2001) EWA volume splatting. In: Proceedings visualization, VIS'01. IEEE, pp 29–538
46. Bekuzarov M, Bermudez A, Lee J-Y, Li H (2023) Xmem++: production-level video segmentation from few annotated frames. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 635–644
47. Cheng HK, Oh SW, Price B, Lee J-Y, Schwing A (2024) Putting the object back into video object segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 3151–3161
48. Pont-Tuset J, Perazzi F, Caelles S, Arbeláez P, Sorkine-Hornung A, Van Gool L (2017) The 2017 Davis challenge on video object segmentation. arXiv:1704.00675
49. Perazzi F, Pont-Tuset J, McWilliams B, Van Gool L, Gross M, Sorkine-Hornung A (2016) A benchmark dataset and evaluation methodology for video object segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 724–732
50. Schönberger JL, Zheng E, Frahm J-M, Pollefeys M (2016) Pixelwise view selection for unstructured multi-view stereo. Computer Vision–ECCV 2016: 14th European conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part III 14. Springer, pp 501–518
51. Schonberger JL, Frahm J-M (2016) Structure-from-motion revisited. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4104–4113
52. Choi S, Song H, Kim J, Kim T, Do H (2024) Click-Gaussian: interactive segmentation to any 3D Gaussians. arXiv:2407.11793
53. Cen J et al (2023) Segment any 3D Gaussians. arXiv:2312.00860
54. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP (2004) Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process 13(4):600–612
55. Müller T, Evans A, Schied C, Keller A (2022) Instant neural graphics primitives with a multiresolution hash encoding. ACM Trans Graph (TOG) 41(4):1–15
56. Tancik M et al (2023) Nerfstudio: a modular framework for neural radiance field development. In: ACM SIGGRAPH 2023 conference proceedings, pp 1–12
57. Barron JT, Mildenhall B, Tancik M, Hedman P, Martin-Brualla R, Srinivasan PP (2021) Mip-nerf: a multiscale representation for anti-aliasing neural radiance fields. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 5855–5864

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.