Remote Sensing Applications: Society and Environment 40 (2025) 101807

Contents lists available at ScienceDirect

Remote Sensing Applications: Society and Environment

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

Gaussian Building Mesh (GBM): Extract a building's 3D mesh with Google Earth and Gaussian Splatting

Kyle Gao [a], Dening Lu [a,*], Hongjie He [b], Liangzhi Li [c], Linlin Xu [a,d], Michael A. Chapman [e], Jonathan Li [a,b]

[a] Department of Systems Design Engineering, University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada
[b] Department of Geography and Environmental Management, University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada
[c] College of Land Engineering, Chang'an University, Xi'an, Shaanxi, 710064, China
[d] Department of Geomatics Engineering, University of Calgary, Calgary, Alberta, T2N 1N4, Canada
[e] Department of Civil Engineering, Toronto Metropolitan University, Toronto, Ontario, M5B 2K3, Canada

ARTICLE INFO

Keywords:
Remote sensing
Gaussian splatting
3D mesh
SAM
3D building model
Google Earth

ABSTRACT

The rapid convergence of computer vision and digital technologies is redefining how buildings are captured, modeled, and managed. In computer vision, recently released open-source pre-trained foundational image segmentation and object detection models allow for geometrically consistent segmentation of objects of interest in multi-view 2D images. Text-based or click-based prompts can be used to segment objects of interest without requiring labeled training datasets, allowing for both user-prompted and automated segmentation. Simultaneously, Gaussian Splatting allows for learning a 3D representation of a scene's geometry and radiance based on 2D images. Combining Google Earth Studio, SAM2+GroundingDINO, 2D Gaussian Splatting, and our improvements in mask refinement based on morphological operations and contour simplification, we created a pipeline to extract the 3D mesh of any building based on its name, address, or geographic coordinates. Our pipeline offers a fast and user-accessible framework for rapid 3D modeling of built environments and structures, enabling downstream applications.

1. Introduction

The extraction of 3D building models from remote sensing images has long been an active research topic, with applications ranging from urban planning, disaster management, environmental monitoring, telecommunications, construction, digital media, and many more. In remote sensing, the standard way to extract 3D information from 2D images is by using photogrammetry, which involves identifying key points in multi-view images of a scene and then triangulating and registering these key points into a cohesive 3D point cloud of the scene. Recent innovations in learning-based 3D rendering approaches, namely Gaussian Splatting, have opened up new possibilities in learning both accurate 3D lighting and 3D geometry from 2D images, attracting much research interest. Additionally, advances in deep learning image processing have greatly improved the capabilities of extracting individual objects from images.

Leveraging Google Earth Studio (Alphabet Inc., 2015-2024), and inspired by GS2Mesh (Wolf et al., 2024), we propose a 3D building mesh extraction pipeline capable of extracting the 3D mesh of a building given its proper name, address, postal code, or

* Corresponding author.
E-mail addresses: y56gao@uwaterloo.ca (K. Gao), d62lu@uwaterloo.ca (D. Lu), hongjie.he@uwaterloo.ca (H. He), liliangzhi@chd.edu.cn (L. Li), lincoln.xu@ucalgary.ca (L. Xu), mchapman@torontomu.ca (M.A. Chapman), junli@uwaterloo.ca (J. Li).

https://doi.org/10.1016/j.rsase.2025.101807
Received 7 August 2025; Received in revised form 1 November 2025; Accepted 19 November 2025
Available online 20 November 2025

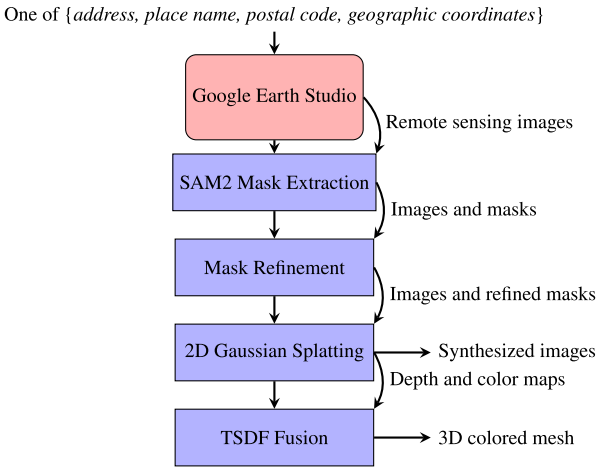

Fig. 1. Flow chart of our pipeline. The output data modality at each step is denoted on the right. Processes and internal modules are in blue. The external module is in red. (For interpretation of the references to color in this figure or legend, the reader is referred to the web version of this article.).

geographical coordinates. When combined with off-the-shelf registration methods, our pipeline enables downstream tasks such as building information models, construction verification, and automated safety assessment.

Our contributions are as follows:

- We introduce a novel meshing pipeline that enables automatic 3D mesh reconstruction of a building directly from semantic inputs such as its name, address, postal code, or geographic coordinates without requiring on-site LiDAR or camera data. This represents a significant step toward text-to −3D geospatial reconstruction, bridging structured location data and 3D modeling.
- We leverage and improve Segment Anything Model-2 (Ravi et al., 2024) and GroundingDINO (Liu et al., 2024b) for geometrically consistent building masking.
- We created a Mask Refinement Module that enhances SAM2 and GroundingDINO performance by combining morphological operations, the Ramer–Douglas–Peucker algorithm, and mask re-prompting, yielding noticeably cleaner and more coherent masks when tested on an independent dataset.
- We improved and modified an unpublished implementation of 2DGS (Huang et al., 2024) to generate 3D colored building meshes via masked TSDF integration, with refinements in depth map filtering, smoothing, and hyperparameter tuning during both training and meshing.

GS2Mesh (Wolf et al., 2024) combines out-of-the-box object segmentation and 3DGS to generate 3D meshes of objects regardless of the background. To our knowledge, it is the only method similar to our own that performs object-based 3D mesh extraction from text prompts or click prompts. This method was the main source of our inspiration. There are many other Gaussian Splatting-based 3D geometry and mesh extraction methods. However, they focus on reconstructing the entire scene as opposed to being able to extract the mesh of designated objects based on user-based text input, clicks, or system-level automation.

2. Related works

2.1. 3D reconstruction from 2D images

Classical 3D reconstruction from 2D images is a well-studied area. Photogrammetry is well-understood in remote sensing and computer vision and is widely used for academic, industrial, and commercial purposes. For this purpose, multiple software suites and code libraries (Schonberger and Frahm, 2016; Snavely, 2008; Adorjan, 2016; Reality, 2024; Solutions, 2024; Esri, 2024) are commercially available or open-sourced, with some specifically designed for remote sensing applications. Many of these not only allow for the extraction of a sparse point cloud from 2D images but also generate a 3D mesh. These 3D meshes are often more desirable than point clouds, as they are more photorealistic and allow for use in 3D modeling and simulations. COLMAP (Schonberger and Frahm, 2016), in particular, is worth highlighting. It is a photogrammetry and Structure-from-Motion (SfM) library widely used as a preprocessing step for modern novel view synthesis and learning-based 3D rendering methods. It is included in most Gaussian Splatting models' pipelines, including our own.

2.2. Modern novel-view synthesis and neural/learning-based rendering

We categorize two broad families of recently developed methods in this section: NeRF and Gaussian Splatting, with a focus on papers that apply these techniques to remote sensing and building model extraction. Given the extensive body of work in this field, it is more comprehensively captured in survey papers (Gao et al., 2022; Chen and Wang, 2024) than in a few paragraphs of literature review.

Both NeRF and Gaussian Splatting methods are fundamentally based on two core concepts: differentiable rendering/rasterization and learning-based 3D representation. Differentiable rendering and rasterization enable the computation of gradients during the creation of 2D images from a 3D representation. This, in turn, allows for the learning and refinement of the 3D representation through loss-function optimization, similar to neural network training.

Neural Radiance Field (NeRF) (Mildenhall et al., 2021), introduced in 2020, has garnered significant attention in this field. In NeRF-based models, the 3D representation consists of a 3D radiance field (directionally dependent color field) and a 3D density field, both represented as Multi-Layer Perceptrons (MLPs). Differentiable volume rendering is employed to generate 2D images by sampling and integrating local 3D color/radiance and density. These radiance and density fields are trained from scratch using a photometric loss function, meaning NeRF models learn a 3D representation of a scene from 2D images. Significant advancements have been made in this area, including (Barron et al., 2021, 2022; Müller et al., 2022), with models like (Wang et al., 2021; Oechsle et al., 2021; Wang et al., 2023) focusing on improving 3D geometry extraction, and others like (Xiangli et al., 2022; Rematas et al., 2022; Tancik et al., 2022; Derksen and Izzo, 2021; Marí et al., 2022) applying NeRF techniques to remote sensing or urban scene capture.

3D Gaussian Splatting (3DGS) (Kerbl et al., 2023), first proposed in 2023, and subsequent Gaussian Splatting models have largely surpassed NeRF-based approaches over the past year. 3DGS uses a large number of 3D Gaussian distributions (also known as primitives in computer graphics) as a learned 3D representation. In addition to standard 3D Gaussian distribution function parameters, each Gaussian primitive also has directionally dependent color and opacity α, which are all trainable parameters. The method uses a differentiable tile-based rasterizer, projecting the 3D Gaussians onto the to-be-rasterized image, and α-blending the projected Gaussians. This allows the 3D Gaussians representing the scene's radiance and geometry to be learned from scratch using a photometric loss function. Compared to NeRF, Gaussian Splatting models are generally much faster to train, have higher view-synthesis quality, but require more memory. Methods such as (Huang et al., 2024; Guédon and Lepetit, 2024; Yu et al., 2024) improved 3D extraction.

While several methods have explored 3D Gaussian Splatting for urban environments, not all are suited for remote sensing. HUGS (Zhou et al., 2024) and HO-Gaussian (Li et al., 2025), for instance, are designed primarily for urban reconstruction from street-level imagery in autonomous driving scenarios, not for aerial or satellite applications. In contrast, Wang et al. (2024) propose a framework that generates and regularizes pseudo-viewpoints to improve training stability in view-sparse remote sensing environments, making it more relevant to aerial reconstruction. Chen et al. (2024) presents a practical drone-based system that performs 3D Gaussian Splatting reconstruction and imports the results into a 3D simulation engine, demonstrating real-world applicability in dynamic scene analysis. Among these, Gao et al. (2025a) is most closely aligned with the present work, as it also employs Google Earth Studio for large-scale 3D reconstruction. However, our previous work (Gao et al., 2025a) remains limited to scene-level point-cloud generation, lacking per-building segmentation. This manuscript extends that line of research by focusing on object-level mesh extraction through open-vocabulary masking and new techniques for improving mask accuracy, enabling finer-grained 3D reconstruction in complex urban environments.

However, we note that these methods all focus on scene-wide capture and cannot extract the 3D mesh of individual objects without further processing. Only GS2Mesh (Wolf et al., 2024), being an out-of-the-box pipeline leveraging Segment Anything Model object masking, allows for the extraction of a 3D mesh from user prompting without further training or processing. Unfortunately, according to our preliminary testing, GS2Mesh often fails in remote sensing building extraction scenarios due to issues in the mask generation module and the pretrained deep learning-based stereo depth map module, which motivated our research.

3. Background

3.1. 3D Gaussian splatting

3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) is a view-synthesis technique that enables the learning of a 3D scene's geometry and lighting from multi-view 2D images, which can then be used to rasterize the scene from novel viewpoints. The process begins with an initial point cloud, often generated using COLMAP (Schonberger and Frahm, 2016) Structure-from-Motion (SfM). For each point in this cloud, 3DGS initializes a Gaussian primitive that encodes learnable parameters such as mean, covariance, opacity, and local lighting in the red, green, and blue channels, represented as spherical harmonic coefficients.

To render an image of the scene, a differentiable tile-based rasterizer is employed, projecting the Gaussian primitives into 2D on the image plane. These projected Gaussians are alpha-blended to generate the final image. During training, the learning of the Gaussian Splatting parameters is guided by comparing the rasterized image for a given camera pose to the ground truth training image. The difference in pixel values and overall image quality is used to optimize the parameters of the Gaussians.

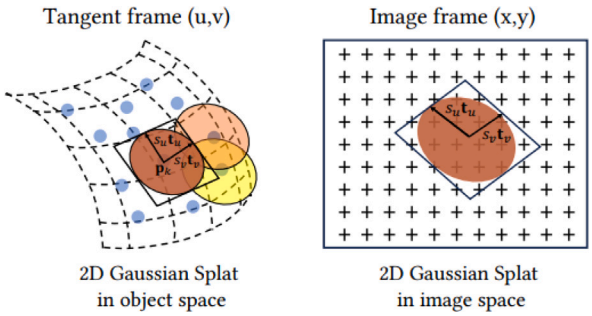

Fig. 2. 2D Gaussian Splatting uses surfaced aligned 2D Gaussian primitives embedded in 3D to represent the 3D scene. 2D Gaussian is represented by it's 3D position $\mathbf{p_k}$, it's scale s_u, s_v, and it's orientation $\mathbf{t_u}, \mathbf{t_v}$. Huang et al. (2024).

3.2. 2D Gaussian splatting

2D Gaussian Splatting (2DGS) (Huang et al., 2024) enhances the standard 3D Gaussian Splatting (3DGS), improving the reproduction of 3D surface geometry. While GS2Mesh slightly outperforms 2DGS on the DTU benchmark, the two methods are fundamentally different. GS2Mesh is primarily a mesh extraction pipeline that utilizes vanilla 3DGS during the 3D reconstruction phase. In contrast, 2DGS is a significant improvement on 3DGS itself, altering the nature of the Gaussian splats. As a result, 2DGS can be used to replace 3DGS in many pipelines, as we have done.

The key innovation in 2D Gaussian Splatting is the representation of the scene using 2D-oriented planar Gaussians instead of 3D Gaussians. Like standard 3DGS, 2DGS employs Gaussian primitives that store spherical harmonic coefficients for each color channel, local transparency α, and 3D location $\mathbf{p}_k$. However, unlike 3D Gaussian primitives, 2D Gaussian primitives have two scalar values to represent variance (s_u, s_v) and two tangent vectors $(\mathbf{t}_u, \mathbf{t}_v)$ whose cross product results in the normal vector that defines orientation (see Fig. 2). Depth maps can be accurately rendered using the projected depth value. For more details, we refer readers to the original paper (Huang et al., 2024).

3.3. Segment anything model (version 2) and grounding DINO

Segment Anything Model (SAM) (Kirillov et al., 2023) is an out-of-the-box image segmentation model pretrained on a massive billion-image dataset. It is capable of segmenting most objects given point-based or bounding box priors, without requiring further training from the user. Users can provide point-click or bounding box prompts to identify the object(s) of interest, and SAM will return segmentation masks and associated scores. However, a key limitation of SAM is that, when applied to a scene viewed from multiple images, the individual masks generated are not necessarily consistent with each other. This inconsistency limits SAM's effectiveness in segmenting video data, which requires temporal consistency, and multi-view data, which requires 3D consistency.

SAM2 (Ravi et al., 2024), released in August 2024, addresses this issue by introducing consistent video segmentation that maintains 3D and temporal consistency through the use of memory attention. Since we aim to extract the mesh of an individual building rather than an entire neighborhood, segmentation masks are crucial.

GroundingDINO (Liu et al., 2024b) is a pretrained open-set object detector capable of extracting object bounding boxes in images from natural language prompts without requiring additional training from the user. GroundingDINO can be combined with SAM/SAM2 to enable text-based object segmentation. The process involves first generating a bounding box from the text description and then using the bounding box to prompt SAM/SAM2. This combination of GroundingDINO and SAM/SAM2 is known as the Grounded-SAM pipeline (Ren et al., 2024), which is available as an open-source library.

3.4. GS2Mesh

GS2Mesh (Wolf et al., 2024) is a Gaussian Splatting-based 3D reconstruction pipeline, outperforming concurrent and competing methods such as SuGAR (Guédon and Lepetit, 2024), 2DGS (Huang et al., 2024), and GOF (Yu et al., 2024) on the DTU dataset benchmark (Sølund et al., 2016).

- GS2Mesh learns and stores the scene in a standard 3DGS model.
- The trained 3DGS model is then used to generate a stereo pair for each input image. Each stereo pair is used to generate a depth image.
- Grounded-SAM is used to generate multi-view masks to mask out the background for mesh extraction.
- A pre-trained depth from stereo model DLNR is used to generate depth maps for each stereo pair.
- The entire ensemble of depth images is integrated into a mesh using the Truncated Signed Distance Function fusion (TSDF) algorithm (Curless and Levoy, 1996) with the Marching-Cubes algorithm (Lorensen and Cline, 1998).

More specifically, a standard 3DGS model is trained from the input images. The 3DGS model is then used to generate a stereo pair for each training image's camera pose. For each pair, the left image is generated with the same camera pose as the training image, and the right image is generated with a small shift $[b, 0, 0]$ to the right. Since Gaussian Splatting models perform best near training poses, this method ensures visual high quality in the generated stereo image pair.

For these stereo image pairs, the Segment Anything Model 2 (SAM2) is used to generate segmentation masks for the objects. In the GS2Mesh paper, which was published before SAM2, the authors addressed the 3D consistency issue by projecting the initial mask onto other frames, sampling new points within the projected mask as SAM prompts, and creating a new SAM mask from these prompts for each frame. The GS2Mesh codebase has since been updated to use SAM2 for 3D geometry-consistent mask generation.

From these stereo pairs, DLNR (Zhao et al., 2023), pretrained on the Middlebury dataset (Scharstein and Szeliski, 2002), is used for depth extraction from stereo images. To improve the quality of the reconstructions, multiple masks are applied to the stereo model's output to filter out regions visible to only one camera and to discard depth estimates outside the valid range.

Following depth extraction, a standard Open3D (Zhou et al., 2018) implementation of the TSDF algorithm initializes and populates a voxel grid with the scene geometry. The voxel representation of the scene is populated with the signed distance to the nearest scene surface, integrated from the depth images generated by DLNR. The marching cubes algorithm then assigns each cube's vertices in the voxel grid to be inside or outside the nearest surface based on the previously calculated TSDF values. Based on the 8-vertex configuration, a local surface is meshed for each cube, which is repeated across the entire voxel grid. This process generates a mesh from the voxel representation.

3.5. Evaluation metrics

For 2D view synthesis visual quality assessment, we use the commonly accepted Peak Signal-to-Noise Ratio (PSNR) (Netravali, 2013), 2D Structural Similarity Index Measure (SSIM) (Wang et al., 2004), and Learned Perceptual Image Patch Similarity (LPIPS) (Zhang et al., 2018). These are full-reference metrics that compare an assessed image with a ground truth image. PSNR and SSIM are higher when the assessed image and the ground truth are similar. SSIM achieves a maximum value of 1 when the two images are identical. LPIPS, on the other hand, is lower when the two images are similar, with a minimum of 0 when the two images are identical.

For 3D mesh quality assessment, we use the 3D Structural Similarity Index Measure (3D-SSIM) (Zeng and Wang, 2012), comparing a 360-degree rendering video of the mesh with a 360-degree ground truth video created by segmenting the building from its background in the Google Earth training images. We note that there are other full-reference 3D geometry and visual quality metrics that compare 3D models to other 3D models. However, we lack ground truth 3D models for the buildings we meshed and only have access to ground truth 2D images. This is the main reason for using the video-based 3D-SSIM for mesh quality comparison. We provide both the average 3D-SSIM across the entire video and the minimum 3D-SSIM across video frames.

4. Methodology

Leveraging Google Earth Studio and inspired by GS2Mesh, we created a 3D building mesh extraction pipeline capable of extracting the 3D mesh of a building given its location name, address, postal code, or geographic coordinates. We performed our experiments on a single Nvidia GTX 3080 with Pytorch 2.01 compiled with CUDA 11.8. As shown in Fig. 1, our pipeline consists of the following steps:

1. **Multi-view remote-sensing image collection**: We leverage Google Earth Studio to collect multi-view images of a building of interest using its name, address, or postal code or coordinates.
2. **Automated building mask extraction**: We automatically extract multi-view consistent building masks for the building of interest in each image using system-level prompts.
3. **Building mask refinement**: We refine SAM2 masks using morphological dilation, which slightly extends the mask outwards and fills in holes. We then simplify mask contours using the Ramer–Peuker–Douglas algorithm. Optionally, we also allow for user-based re-prompting to correct possible errors from the automated process.
4. **Gaussian Splatting**: We train a 2DGS+ (Ye and GitHub-Contributors, 2024) model to learn building geometry and radiance.
5. **Mesh extraction**: We perform masked Truncated Signed Distance Function Fusion on smoothed depth maps to extract a 3D building mesh. We perform mesh refinement and mesh simplification using Open3D triangular mesh post-processing functionalities.

4.1. Google Earth Studio

Google Earth Studio (Alphabet Inc., 2015-2024) is a web-based animation tool. With access to Google's vast collection of 2D and 3D Earth data, ranging from large geological formations to individual buildings, Google Earth Studio provides a simple and efficient way to collect off-nadir images. While traditionally used for visualization and cinematography, we repurpose this platform as the data acquisition module in our frame for generating controlled, high-fidelity multi-view imagery. This approach enables the creation of synthetic yet geographically accurate datasets optimized for the training and evaluation of 3D Gaussian Splatting (3DGS) models (Gao et al., 2025a). Google Earth Studio allows for the specification of a target of interest in terms of longitude and latitude coordinates, address, postal code, or location name. After selecting the target of interest, Google Earth Studio enables the

Fig. 3. Example of Google Earth Studio images. **Top Left** Buckingham Palace (high-resolution, consistent geometry) **Top right**: CN Tower (high-resolution, consistent geometry). **Bottom-left**: Whitehouse (low-resolution, consistent geometry) **Bottom-right**: hotel in Hambantota–Sri Lanka (low-resolution, distorted geometry).

specification of a camera path for which images of the target location are rendered. By selecting a circular camera path orbiting above and pointing towards the target of interest, Google Earth Studio allows for the extraction of a multi-view dataset of the building with 360-degree coverage, well-suited for 3D reconstruction.

Through this module, we tested two complementary sets of scenes: 13 high-quality scenes featuring detailed buildings selected for meshing, and 5 lower-resolution scenes used to examine model robustness under degraded input conditions (see Table 1). The White House is present in both sets; its reduced resolution reflects the Washington D.C. restricted airspace, which constrains aerial imaging. Each scene comprises 31 frames sampled evenly along a full circular orbit around the structure, ensuring consistent spatial coverage and angular diversity. Building identification can be performed and were tested through multiple methods: address, postal code, geographic coordinates, and named search. Camera tilt angles were discretized to the nearest half-degree, providing a fine-grained control over viewing geometry. A tilt of 0° corresponds to nadir (downward) imaging, while 90° represents a horizontal view parallel to the ground.

Upon inspection of the low-resolution and low-coverage scenes, we observed that, except for the White House scene, the Google Earth Studio multi-view images did not yield coherent multi-view geometry. For the McGill Arctic Research Station, Summit Research Station, Alomar Observatory, and the hotel in Hambantota, Sri Lanka, COLMAP failed to preprocess the scene data, and Gaussian Splatting training did not produce coherent results. The Google Earth Studio images for these scenes exhibited highly distorted geometry, as shown in Fig. 3, causing COLMAP initialization to fail, which in turn led to the failure of any tested Gaussian Splatting modules. In contrast, the White House scene, although low in resolution, maintained coherent geometry, allowing the pipeline to reconstruct its mesh successfully. These limitations are discussed further in Section 6. For these reasons, the remaining experiments were conducted using the 13 buildings for which reconstruction was successful.

4.2. Mask extraction and mask refinement

SAM2 provides a rough segmentation mask of the building. This process can be automated by providing system-level prompts to identify the building of interest. We noticed the following problems while inspecting the SAM2 segmentation masks and the 3D mesh extraction that resulted from directly using those masks.

1. **Poor geometry and holes**: We noticed that SAM2 had trouble with some buildings whose roof or wall colors might be confused with the background, resulting in incomplete masks with holes.
2. **Noise pixels and false positives**: SAM2 masks sometimes produced small false-positive pixels away from the object of interest. This occurred at times with multiple identical buildings in close proximity to each other.
3. **Poor mask boundary at building base**: SAM2 produced masks with poor geometry at the buildings' base, often resulting in incomplete masking and jagged mask boundaries. This resulted in 3D reconstructions with jagged bottoms, greatly affecting visual quality.

Table 1
Camera altitude and tilt for Google Earth Studio scenes' camera flight path. Low-resolution/low-view-coverage scenes are colored in gray.

Location	Camera altitude (m)	Camera tilt (°)
ICON	680	63.5
Canada Parliament	119	63.5
CN Tower	1870	47.5
Laurel Heights	713	57.0
Perimeter Institute	882	25.5
Dana Porter Library	526	57.0
Townhouse	527	45.0
Coloseum	364	65.6
Buckingham	350	63.4
Westminster	344	63.0
Pyramid	1211	51.3
Bath Chancellory	531	50.2
Whitehouse	310	43.6
McGill Arctic Research Station	363	42.4
Summit Arctic Research Station	3548	50.2
Alomar Observatory	704	63.0
Hambantota, Sri Lanka	217	55.0

4. **Wrong building identification**: When using automated system prompts in scenes where the building of interest is not a key landmark (e.g., when multiple identical buildings are together), SAM2 at times segmented the wrong building.
5. **Incomplete geometry**: When using automated system-level prompting, at times, not the entire building of interest was segmented. Moreover, for complex buildings, SAM2 would at times lose part of the building in long image sequences.

To address these issues, we developed a mask refinement pipeline. We optionally morphologically erode the mask first, removing noise. Then, we grow the mask using morphological dilation to fill in any holes and expand incomplete masks. We then extract the mask's contour using Canny's algorithm and straighten/simplify the contour using the Ramer–Douglas–Peucker algorithm. The contour is then filled into a refined mask, addressing these issues. Our SAM2 image sequence-based building segmentation (mask generation) process is as follows.

4.2.1. Automated mask extraction

By using the Grounded-SAM pipeline with SAM2, we extract object masks without further training with natural language text. We use the system prompt "the central landmark or building of interest" to automatically extract the multi-view segmentation masks. Additionally, if a uniquely identifying proper noun (e.g., "The Parliament of Canada") was used to extract the multi-view images of the building from Google Earth Studio, we also automatically pass this to the Grounded-SAM pipeline as an additional system prompt.

4.2.2. User re-prompting

To solve the aforementioned issue of incomplete geometry, we modified the masking pipeline to add optional user re-prompting. We allow users to identify frames with errors and re-prompt the image at those frames. SAM2 re-propagates the user's new prompt throughout all frames, effectively solving these mask inconsistencies. The need for reprompting depends on scene complexity. Across the tested scenes, reprompting was required in approximately 40% of cases. In each of these scenes which need reprompting, either one or two frames (5% of frames) were reprompted (see Fig. 4).

4.2.3. Morphological dilation and erosion

The morphological dilation $\oplus$ of an image I by a filter F is defined by

$$I \oplus F = \bigcup_{p \in F} I_p, \tag{1}$$

where I_p denotes the image I shifted by pixel $p = (p_x, p_y)$. Roughly speaking, this grows the image by sliding the filter along the pixels of the original image. This operation can be performed iteratively to further extend the image. At each step, the output image of the previous iteration is dilated again using the same filter.

Before the morphological dilation, we add an optional morphological erosion step (with the same parameters), which is the inverse operation of morphological dilation. That is, morphological erosion shrinks the mask according to the filter. This step serves two purposes. Firstly, depending on the outline of the building, we may not want the dilated mask to extend past the building's true boundaries. Performing the erosion-dilation steps allows the mask to be roughly as tight-fitting as the initial SAM2 mask, while still morphologically filling in holes in the mask. Secondly, at times, SAM2 masks introduce small false-positive pixels away from the central object in the form of noise pixels. This step allows the pipeline to erode away this type of noise.

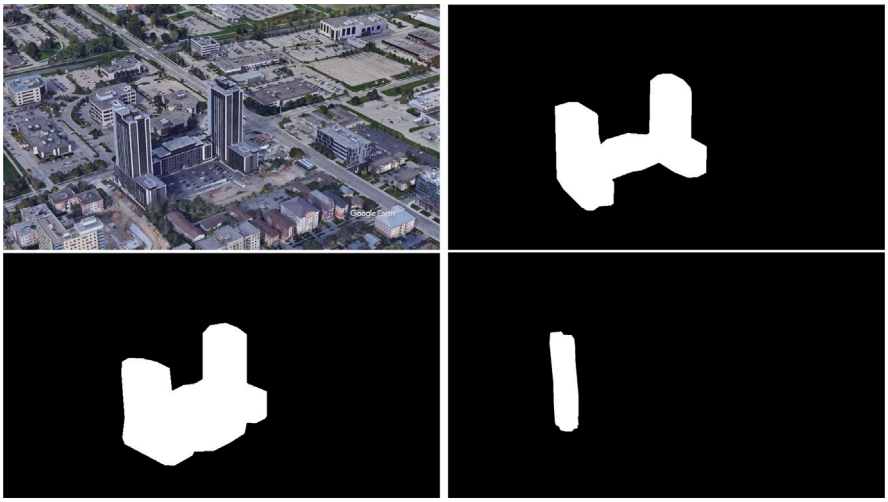

Fig. 4. Example of SAM-2 mask inconsistencies. **Top Left**: 2D image of the building of interest. **Top right**: SAM-2 mask of the same frame. **Bottom-left**: SAM-2 mask in a later frame after some rotation, with the parking lot masked in erroneously. **Bottom-right**: SAM-2 mask in a later frame, with the second tower missing.

4.2.4. Canny edge extraction

Canny edge extraction is a well-known classical algorithm for extracting lines using the image's gradient. The Canny algorithm is well-known in computer vision. As such, we refer readers to the original paper for details (Canny, 1986).

4.2.5. Ramer–Douglas–Peucker contour refinement

The Ramer–Douglas–Peucker algorithm (RDP) (Ramer, 1972) is an iterative contour simplification algorithm that recursively reduces the number of points in the contour. The algorithm is as follows:

Given:

A curve defined by a sequence of ordered points:

$$P = \{P_1, P_2, \ldots, P_n\}, \quad \text{where } P_i = (x_i, y_i) \tag{2}$$

and a threshold parameter:

$$\epsilon > 0 \tag{3}$$

Algorithm:

1. Start with endpoints P_1 and P_n. Define the line segment: $\overline{P_1 P_n}$
2. For each point $P_i = (x_i, y_i)$ with $i \in [2, n-1]$, compute the perpendicular distance d_i to the line $\overline{P_1 P_n}$:
$$d_i = \frac{|(x_n - x_1)(y_1 - y_i) - (x_1 - x_i)(y_n - y_1)|}{\sqrt{(x_n - x_1)^2 + (y_n - y_1)^2}} \tag{4}$$
3. Determine the maximum distance:
$$d_{\max} = \max_{i=2,\ldots,n-1} d_i \tag{5}$$
4. If $d_{\max} > \epsilon$, let P_k be the point where the maximum occurs. Recursively apply the algorithm to:
$$\{P_1, P_2, \ldots, P_k\} \quad \text{and} \quad \{P_k, P_{k+1}, \ldots, P_n\} \tag{6}$$
5. If $d_{\max} \leq \epsilon$, approximate the set by $\overline{P_1 P_n}$ and keep only P_1 and P_n.

Output: A reduced set $P' \subseteq P$, forming a piecewise linear approximation within tolerance ϵ.

We used $\epsilon = \frac{p}{500}$, as the ratio of the perimeter p of the contour being simplified, allowing longer perimeter contours to retain more complexity. The resulting building contours became linear interpolations of approximately 20–40 points. We performed a linear parameter search for ϵ, and found that a reasonable range of values $\frac{p}{200} \sim \frac{p}{800}$ worked well. We then filled the contour to create the refined mask, which addresses the aforementioned issues. We tested both our own implementation and an OpenCV implementation and found no discernible difference.

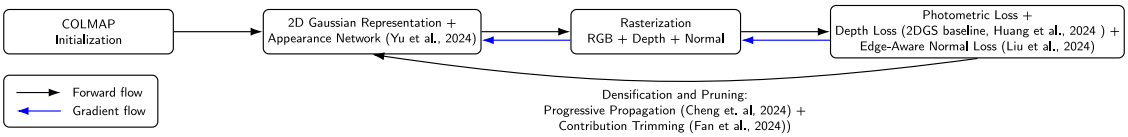

Fig. 5. Flowchart of 2DGS+ as our Gaussian Splatting Module, highlighting additions to 2DGS.

4.3. 2DGS+

We compared 2DGS, SuGAR, and GOF as potential Gaussian Splatting frameworks. We selected 2DGS because it produced superior mesh reconstruction results compared to SuGAR and performance comparable to GOF, while offering greater efficiency and ease of use. We confirmed this in our ablation study in Table 4. After developing initial extensions to 2DGS, we adopted 2DGS+[1] as our foundation, as it integrated similar enhancements and broader improvements from the research community.

2DGS+ is a state-of-the-art implementation of Gaussian Splatting built on the foundations of 2DGS. This implementation, which we dubbed 2DGS+ (Ye and GitHub-Contributors, 2024), is an unpublished fork of the 2DGS repository and combines many recent advances in Gaussian Splatting, improving 2DGS with ideas from AbsGS (Ye et al., 2024), PixelGS (Zhang et al., 2024), TrimGS (Fan et al., 2024), AtomGS (Liu et al., 2024a), GaussianPro (Cheng et al., 2024), and Taming-3DGS (Mallick et al., 2024). We do not make use of all these additions. The useful additions to the original 2DGS, as tested in our ablation study, are shown in Fig. 5.

The **Appearance Network** is a small 4-layer convolutional neural network (CNN) used to decouple per-image appearance differences. This was introduced in GOF (Yu et al., 2024) and VastGaussian (Lin et al., 2024). An appearance embedding l_i is learned per training image i, which is used to account for per-image photometric differences that are strictly non-geometric (e.g., lighting change). The embedding is concatenated to each pixel of the down-sampled training image i fed into the appearance CNN, which produces a multiplicative correction for the rendered image; i.e. this correction helps account for lighting/appearance-based differences between the training image and the image produced by Gaussian Splatting. This CNN is optimized along with the Gaussian Splatting model.

Progressive propagation, introduced in GaussianPro (Cheng et al., 2024) propagates accurate geometry from well-modeled regions to under-modeled ones, allowing new Gaussians to be created. Each pixel defines a local 3D plane parameterized by its normal $\mathbf{n}$ and distance d from the camera origin, computed as

$$d = z\,\mathbf{n}^{\top} K^{-1}\mathbf{p}_e, \tag{7}$$

where z is the pixel depth, K is the intrinsic matrix, and $\mathbf{p}_e$ is the homogeneous pixel coordinate. Plane parameters are propagated to neighboring pixels via patch matching and refined using multi-view consistency; pixels with large depth discrepancies are converted into new 3D Gaussians for further optimization.

The **Contribution Trimming** from TrimGS (Fan et al., 2024) is defined by trimming Gaussians based on their Contribution Scores. The Contribution Score of a Gaussian to the kth image is defined as the sum over all pixels p of alpha-blending weights, defined as

$$\mathbf{C}_k = \sum_{p \in \mathcal{P}_k} \alpha_{i(p)} \prod_{j=1}^{i(p)-1} (1 - \alpha_j). \tag{8}$$

where $\alpha_i(p)$ is the opacity of the ith Gaussian contributing to the pixel p, and $\prod_{j=1}^{i(p)-1}(1-\alpha_j)$ is the transmittance term up to the ith Gaussian. This contribution score is then normalized and averaged over the top-5 views for which the Gaussian contributes the most. Low contribution Gaussians are removed every 1000 iterations during training.

The **Edge-Aware Normal Loss** from AtomGS (Liu et al., 2024a) enforces smooth geometry while preserving sharp details. It combines the curvature magnitude of surface normals $|\nabla N|$ with an edge map $|\nabla I|$ derived from image gradients. A weighting function $\omega(x) = (x-1)^q$, with even $q > 0$, reduces smoothing near high-frequency edges and increases it in flat regions. The loss is defined as

$$\mathcal{L}_{\text{normal}} = \frac{1}{HW} \sum_{i=1}^{H} \sum_{j=1}^{W} |\nabla N| \otimes \omega(|\nabla I|), \tag{9}$$

summing over the image height and width H, W. $\otimes$ denotes the Hadamard product. The Edge-Aware Normal Loss encourages normal consistency in smooth areas while preserving geometric discontinuities along object boundaries.

In our experiments, we found that the Gaussian Splatting representations generated from 2DGS+ have fewer floaters (free-floating Gaussians in mid-air), are smoother, and produce better depth maps, resulting in significantly higher-quality meshes during TSDF fusion meshing. We tuned the hyperparameters for 3D mesh extraction quality with a linear hyperparameter search, one parameter at a time. We used the progressive training modality from GaussianPro, densifying the Gaussians until the 25,000th iteration. We used $\lambda_{depth} = 0.2$ and $\lambda_{normal} = 0.1$ as loss weights associated to the depth loss and the normal loss, respectively (see Fig. 5).

[1] 2DGS+ can be found at https://github.com/hugoycj/2d-gaussian-splatting-great-again, and follows the standard Gaussian Splatting license from Inria and MPII.

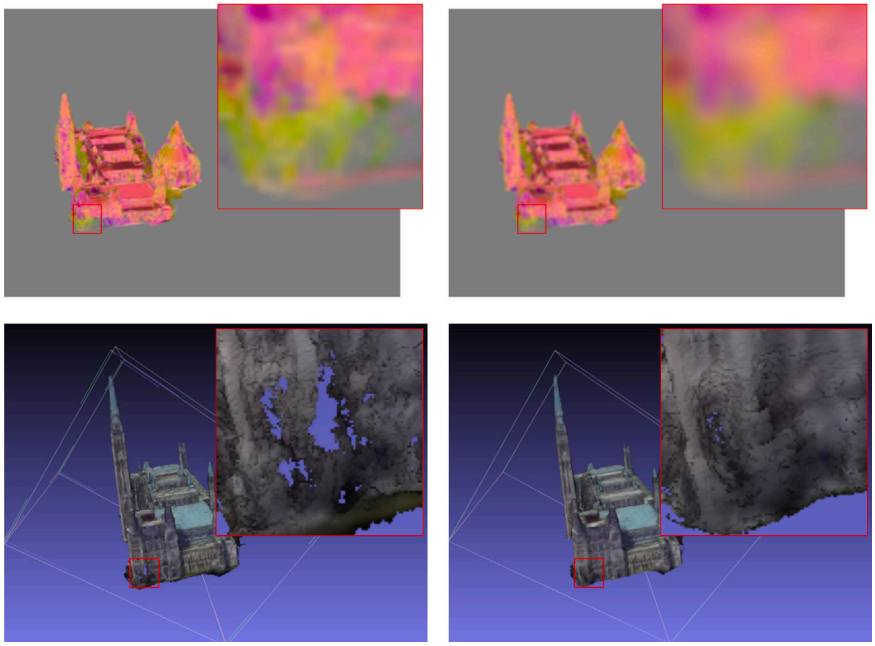

Fig. 6. Example of depth blurring. **Top-left**: Raw depth map. **Top-right**: Smoothed depth map. **Bottom-left**: 3D Mesh from raw depth map. **Bottom-right**: 3D Mesh from the smoothed depth map.

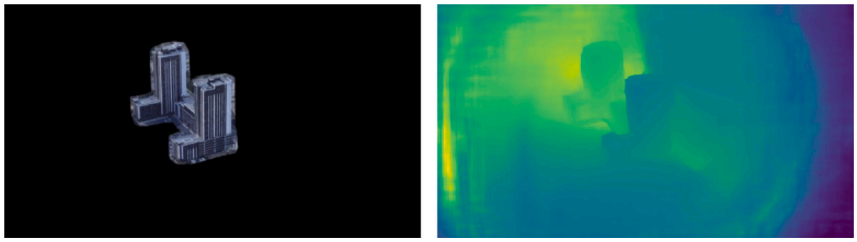

Fig. 7. Example of DLNR failure case. **Left**: 2D image of the building of interest with background masked out. **Right**: DLNR stereo depth reconstruction from a building with background masked out.

4.4. TSDF fusion and mesh extraction

Like GS2Mesh and 2DGS, we used the Open3D implementation of the TSDF Fusion algorithm to convert depth maps into colored 3D meshes. We adapted the 2DGS mesh extraction module, adding a depth map smoothing step and tuning the hyperparameters. We used the 2DGS+ implementation (Huang et al., 2024) of 2DGS with its bounded mesh extraction module.

In early experiments, we noticed the presence of non-smoothness in the depth maps of certain complex buildings, which resulted in poor-quality mesh reconstructions. In these cases, applying a simple Gaussian blur to the depth maps before TSDF fusion greatly improved the mesh reconstruction quality.

After mesh extraction, like GS2Mesh and 2DGS, we used Open3D to clean the mesh by computing mesh clusters, removing unreferenced vertices, and degenerate triangles. Unlike GS2Mesh and 2DGS, we discarded all but the largest mesh cluster.

5. Results

Although there are multiple Gaussian Splatting-based methods for 3D mesh extraction of entire scenes, we found GS2Mesh to be the only method comparable in its application: mesh extraction for single objects from text-based or click-based prompts with optional automation. Additionally, we used it as a starting point to develop our framework. As such, we focus our comparisons against it.

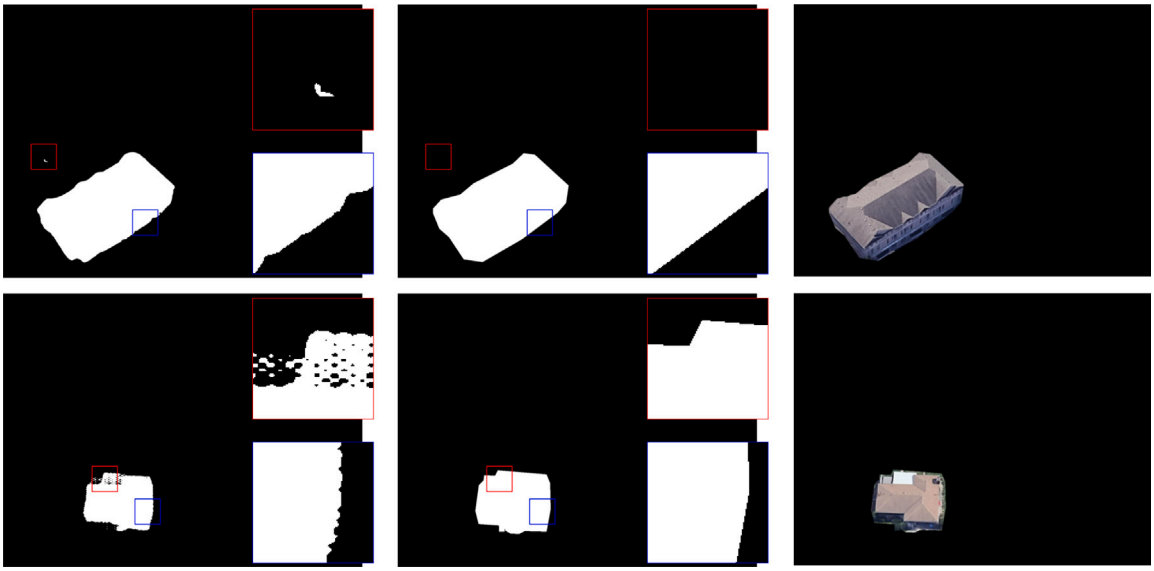

Fig. 8. Example of mask refinement with noise removal **(top)** and hole filling **(bottom)**: **Left**: Raw SAM-2 Mask. **Middle**: Refined Mask. **Right**: masked Image. Regions of interest have been highlighted. Red indicates a region with false positives or false negative pixels, and blue indicates a region where the contour was refined and straightened. (For interpretation of the references to color in this figure or legend, the reader is referred to the web version of this article.)

5.1. Preliminary experiments

Our preliminary experiments on 3D building mesh extraction from remote sensing images showed that the GS2Mesh 3D reconstruction pipeline, which uses DLNR (Zhao et al., 2023), a pretrained neural-network-based stereo depth reconstruction and depth map fusion, only matched the quality of 2DGS depth map fusion in the best-case scenario. More often than not, it struggled in the 3D reconstruction of buildings due to two key issues: poor mask quality and poor stereo depth estimation. DLNR occasionally struggled with stereo depth reconstruction, likely because it was not trained on a remote sensing image dataset and had difficulty interpreting large-scale aerial scenes. This issue becomes particularly pronounced when stereo depth extraction is applied after masking, as the neural network loses the context of the background. In Fig. 7, we observe that DLNR fails completely when performing depth extraction on a building with the background masked out. This motivated us to seek alternative solutions for the 3DGS-DLNR pipeline components in GS2Mesh.

Additional preliminary experiments with 2DGS+ showed that depth maps of certain complex buildings exhibited non-smoothness, resulting in poor-quality 3D meshes. We addressed this issue with the simple solution of smoothing the depth map using a Gaussian filter. We use the OpenCV implementation, which applies two separate one-dimensional Gaussian filter operations. The one-dimensional Gaussian filter kernel is defined as:

$$G(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}} \tag{10}$$

where σ is the standard deviation controlling the spread of the blur. The filtered output is obtained through convolution:

$$f'(x) = (f * G)(x) = \sum_{i=-k}^{k} f(x-i)\, G(i). \tag{11}$$

Here, k defines the kernel radius. We tested various values of σ and k by visually inspecting hole closure in the resulting mesh for all our scenes, and chose $\sigma = 4$ and $k = 15$ as the default for GBM. For two-dimensional images, OpenCV uses the separability property of the Gaussian function, applying two 1D convolutions, one along each axis:

$$I'(x, y) = (G_x * (G_y * I))(x, y) \tag{12}$$

where G_x and G_y are 1D Gaussian kernels in the horizontal and vertical directions, respectively. This approach reduces the computational complexity from $O(n^2)$ to $O(2n)$ while preserving the same smoothing effect. An example is shown in Fig. 6. After applying the filter, the holes in the zoomed-in region were almost completely filled. This particular issue arose due to poor visual coverage of the zoomed-in region in the initial training data.

Table 2
Comparison of 2D novel view synthesis metrics between Gaussian Splatting modules. For each method, the best scene for each score is highlighted in red, the second best in yellow, and the worst is in gray. The best average score is bolded.

Building	2DGS+ (in Our Pipeline)			3DGS (in GS2Mesh)		
	SSIM ↑	PSNR ↑	LPIPS ↓	SSIM ↑	PSNR ↑	LPIPS ↓
Icon	0.9944	37.61	0.0100	0.9886	37.23	0.0162
Canada Parliament	0.9894	35.11	0.0156	0.9321	29.75	0.1076
CN Tower	0.9977	40.01	0.0039	0.9946	42.63	0.0092
Laurel Heights	0.9948	38.13	0.0089	0.9895	38.04	0.0127
Perimeter Institute	0.9950	39.40	0.0080	0.9935	43.93	0.0149
Dana Porter Library	0.9746	32.25	0.0330	0.9923	38.40	0.0109
Townhouse	0.9946	36.82	0.0087	0.9919	39.66	0.0118
Buckingham	0.9919	32.69	0.0118	0.9712	34.88	0.0456
Whitehouse	0.9980	39.99	0.0032	0.9841	35.47	0.0351
Westminster	0.9873	33.81	0.0201	0.9666	33.30	0.0909
Pyramid	0.9974	40.49	0.0038	0.9859	39.55	0.0285
Bath Chancellory	0.9982	42.75	0.0027	0.9811	35.26	0.0310
Colosseum	0.9818	33.39	0.0149	0.9712	33.27	0.0176
Average	**0.9919**	**37.11**	**0.0111**	0.9800	37.03	0.0300

Table 3
Comparison of 3D-SSIM scores between our pipeline and GS2Mesh. For each method, the best scene is highlighted in red, the second best in yellow, and the worst is in gray. The best average score is bolded.

Building	GBM (Ours) 3D-SSIM ↑		GS2Mesh	
	Average (31 frames)	Minimum	Average (31 frames)	Minimum
Icon	0.9287	0.9158	0.8355	0.8000
Canada Parliament	0.8535	0.8458	0.1462	0.0703
CN Tower	0.9841	0.9771	0.9397	0.9282
Laurel Heights	0.9566	0.9529	0.9427	0.9412
Perimeter Institute	0.9747	0.9726	0.9475	0.9431
Data Porter Library	0.8214	0.8078	0.7054	0.6862
Townhouse	0.9780	0.9749	0.9785	0.9761
Buckingham	0.9138	0.9097	0.8361	0.8166
Whitehouse	0.9100	0.9069	0.4302	0.3616
Westminster	0.8694	0.8662	0.8786	0.8613
Pyramid	0.9559	0.9485	0.7839	0.6899
Bath Chancellory	0.9728	0.9712	0.8068	0.7610
Colosseum	0.9075	0.8930	0.9017	0.8213
Average	**0.9251**	**0.9186**	0.7794	0.7428

5.2. Mask refinement

SAM2 masks sometimes included false-positive pixels away from the object of interest. This issue is visible in the left-most subfigure of Fig. 8 (top row). This is likely caused by false-positive segmentation due to the presence of a neighboring building's rooftop with a similar shape and color in the original full image. Our mask refinement algorithm successfully removed the false-positive pixels and refined the mask contour.

SAM2 also occasionally produced masks with holes or false negatives. We observed that this occurred more frequently when part of the building exhibited significantly different coloration from the rest of the structure. An example is shown in Fig. 8 (bottom row), where the garage roof has a different color. Our mask refinement algorithm was able to complete the mask and refine the mask contour.

5.3. View synthesis

Compared to GS2Mesh, we replaced the Gaussian Splatting module in the pipeline, transitioning from the original implementation of 3DGS to 2DGS+. As such, we compare the training set novel view synthesis results of 3DGS and 2DGS+ across scenes. We did not use the standard MipNeRF-360 convention of a training/testing split (leaving out one out of every eight images). Instead, we trained on the entirety of the Google Earth Studio footage, as we wanted smooth coverage of the 360° camera rotation for 3D reconstruction purposes. The novel view synthesis scores on the training set are provided in Table 2.

For 2DGS+ in our pipeline, we trained the model on the masked building without the background. The model's synthesized images were then compared to images of the building with the background masked out. On the other hand, in the GS2Mesh pipeline, the 3DGS training occurs before masking and is conducted on the entire image, including the background. Its 3DGS module generates images with the entire background, and the results are compared with unmasked ground truth images.

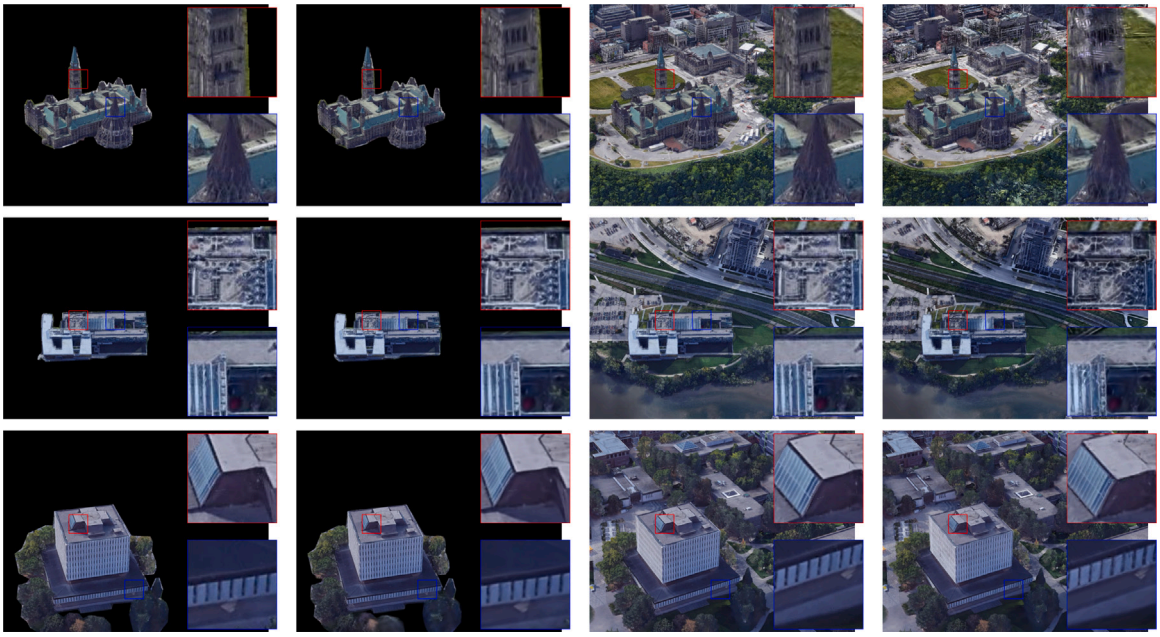

Fig. 9. Visualization of 2D novel view synthesis results. **Left to Right: 2DGS Ground Truth; 2DGS Sythesized Image; 3DGS Ground Truth; 3DGS Synthesized Image**. We note that for all cases, including those not shown, the results were nearly visually indistinguishable from ground truth by human eyes, except for 3DGS trained on the Canada Parliament scene (top right).

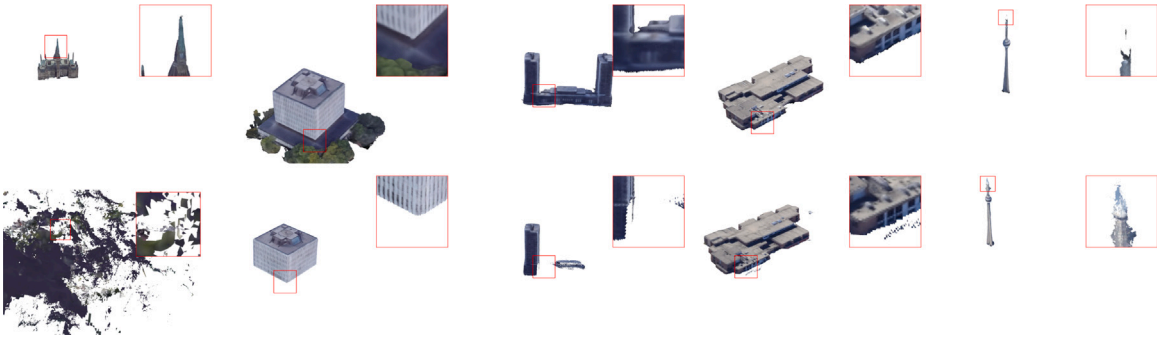

Fig. 10. Visualization of 3D colored mesh results, showcasing imperfections in both methods. **Left to Right**: Canada Parliament; Dana Porter Library; ICON; Laurel Heights; CN Tower. **Top row** are our results, **Bottom row** are GS2Mesh results.

On average, we found that 2DGS+ produced higher PSNR, SSIM, and LPIPS scores. Nonetheless, both models achieved novel view synthesis results that were nearly indistinguishable from ground truth images, with the exception of 3DGS struggling on the Canada Parliament scene, as shown in the top row of Fig. 9 (see the central tower highlighted in the top-right zoom-in). This is also reflected in Table 2, where the 3DGS scores for the Canada Parliament scene are significantly lower than for other scenes (see Fig. 10).

5.4. 3D building mesh extraction

As shown in Table 3, our building extraction results are superior to GS2Mesh, except the Townhouse scene, where both methods scored similarly, within ~0.0005 3D-SSIM. The results are somewhat skewed by the Canada Parliament scene (first column of Fig. 10), where GS2Mesh 3D mesh reconstruction failed due to the significant presence of floaters during the training of its 3DGS module. This failure caused its DLNR depth reconstruction module to produce completely inaccurate depth maps. The poor mesh quality resulted in extremely low 3D-SSIM scores, as reflected in Table 3. The resulting mesh was completely unrecognizable. The poor performance of the 3DGS module of GS2Mesh for this scene is also reflected in its PSNR score in Table 2, which is approximately 10 dB lower than for other scenes. GS2Mesh also partially failed on the ICON scene and Dana Porter Library scene, where the lack of user re-prompting caused the masking to be inconsistent across the initial training images, resulting in missing geometric

Table 4
Ablation study on different Gaussian Splatting modules for view synthesis and 3D meshing in our pipeline. Testset scores color highlighting: best (red), second-best (yellow).

Additions	Train			Test			3DSSIM↑	
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	Mean	Min
2DGS (baseline)	34.0	.987	.021	28.5	.954	.045	.876	.865
SuGaR	35.1	.986	.016	27.5	.943	.042	.693	.650
GoF	35.4	.989	.014	29.0	.958	.038	.890	.859
2DGS+Trim	36.4	.992	.011	28.5	.955	.042	.882	.877
2DGS+Trim+Prog	36.7	.992	.013	28.4	.952	.043	.881	.876
2DGS+Trim+Prog+Norm	36.8	.992	.013	28.3	.951	.044	.884	.870
2DGS+Trim+Prog+EANorm*	36.6	.992	.013	28.4	.952	.042	.891	.887
2DGS+Trim+Prog+EANorm+Appearance	36.7	.992	.013	28.5	.952	.043	.884	.879
2DGS+Trim+Prog+EANorm+Appearance+60k	38.2	.994	.010	28.2	.949	.045	.867	.863
2DGS+Trim+Fast+Norm	36.8	.993	.011	28.7	.955	.041	.882	.877

features and incomplete meshing. Our method produced less noisy meshes overall and cleaner mesh boundaries. This improvement is reflected in the 3D-SSIM scores and can be seen in the fourth column of Fig. 10. However, both GS2Mesh and our method struggled with sharp structural elements, such as the top of the CN Tower, as shown in the fifth column of Fig. 10.

5.5. Ablation studies

The Gaussian splatting ablation study is performed on the Westminster Palace scene, comparing GOF (Yu et al., 2024), SuGaR (Guédon and Lepetit, 2024), as well as the improvements in 2DGS+ compared 2DGS as a baseline. A mip-NeRF (Barron et al., 2021) style training/testing split was used. Both view synthesis and 3D mesh quality assessments are used. As additions to 2DGS, Trim-Gaussian's (Fan et al., 2024) and PixelGS's (Zhang et al., 2024) contribution-based Gaussian pruning is denoted as Trim. Progressive propagation from GaussianPro (Cheng et al., 2024) is denoted as Prog. The standard 2DGS normal-based supervision is denoted as Norm. The Edge-Aware Normal Supervision from AtomGS (Liu et al., 2024a) is denoted EANorm. The Appearance Network from GOF (Yu et al., 2024) is denoted as Appearance. Fast-SSIM from Taming-GS (Mallick et al., 2024) is denoted as Fast. Doubled training iteration is denoted as 60k. As shown in Table 4, despite resulting in lower 2D view synthesis scores than 2DGS+Trim+Fast+Norm and GOF, the 2DGS+Trim+Prog+EANorm module resulted in the highest 3DSSIM score. As such, we use it as our Gaussian splatting backbone and dub it "2DGS+". The training view synthesis scores were consistently higher than testing view scores, indicating a mild overfitting effect. This effect is often observed for Gaussian Splatting training with low training views. As such, we forgo view synthesis scores as a selection criterion for Gaussian splatting modules.

The multi-view Google Earth Studio images do not include ground truth masks. To evaluate the components of our Mask Refinement Module, we used the WHU Building Dataset (Ji et al., 2018), a 2D orthoimage dataset with ground truth building-based semantic segmentation annotations. Because our method is designed for single-object segmentation, we manually selected 50 patches from the validation set containing only a few buildings. We then implemented single-object Intersection over Union (IoU) and Dice score calculations. For a predicted mask P, we searched for the best-matching ground truth component G^* as a connected region in the ground truth mask. The IoU and Dice scores are then given by

$$\text{IoU} = \frac{|P \cap G^*|}{|P \cup G^*|.} \tag{13}$$

$$\text{Dice} = \frac{2|P \cap G^*|}{|P| + |G^*|.} \tag{14}$$

We compared the mean Intersection over Union (mIoU) and mean Dice (mDice) scores for each component of the Mask Refinement Module: baseline, erosion only, dilation only, erosion followed by dilation (ED), Ramer–Douglas–Peucker (RDP), and all operations combined (ALL), as shown in Table 5. An example scene is shown in Fig. 11. Both the quantitative results and visual inspection indicate that RDP contributes the most to mask refinement. However, when combined with morphological operations, the mask quality is further improved.

6. Discussions and future research direction

In terms of experiments, for the Dana Porter Library scene (second column of Fig. 10), without user re-prompting, the base of the building was masked inconsistently due to the presence of occluding trees, which resulted in messy GS2Mesh reconstruction. Instead, for GS2Mesh training on this scene, we found that prompting for the large cubical structure without the first floor and the trees resulted in better reconstruction. Our pipeline was able to properly capture the building base along with the surrounding trees. For the ICON scene (third column of Fig. 10), without re-prompting, the mask for the second tower and part of the connected structure were missing in some frames, resulting in an incomplete reconstruction for GS2Mesh. We note that both methods struggled with sharp, needle-like building structure elements, such as the top of the CN Tower. This was likely due to a combination of the difficulty in masking needle-like objects and the challenge of producing depth maps for these objects.

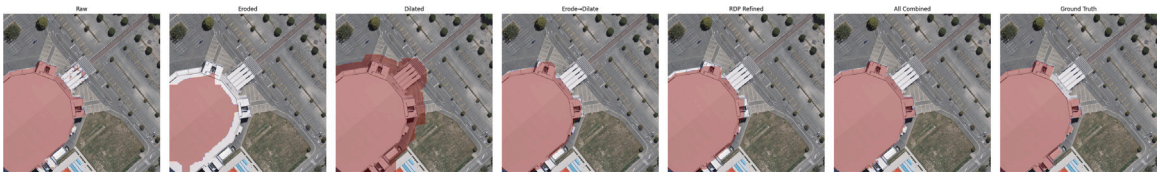

Fig. 11. Ablation study of the Mask Refinement Module on the WHU Building Dataset (Ji et al., 2018), visualized on a representative patch. The mask (pink) is overlaid on top of the aerial image. **Left to right**: Raw SAM Mask, Erosion, Dilation, Erosion → Dilation, Ramer–Peuker–Douglas boundary refinement, combined Mask Refinement, and ground truth. (For interpretation of the references to color in this figure or legend, the reader is referred to the web version of this article.)

Table 5
Ablation study for mask refinement. Color highlighting: best (red), second-best (yellow), and baseline (gray).

Mask ablations	mIoU	mDice
SAM Masks (baseline)	0.7626	0.8518
Erosion	0.4573	0.6068
Dilation	0.5607	0.7019
ED	0.7531	0.8469
RDP	0.8106	0.8858
ALL	0.8298	0.8951

The GS2Mesh failure on the Canada Parliament scene was unexpected. We believe the failure occurred in the 3DGS module, as we noticed a significant amount of floaters (Gaussians floating in empty space). We did not attempt further fine-tuning of the 3DGS training. These floaters were absent during the 2DGS+ stage in our pipeline, even without additional fine-tuning. Future research could enhance the reconstruction of intricate architectural elements, such as the sharp spires observed in the Canada Parliament scene. This can be achieved by supporting the TSDF mesh reconstruction with an additional depth estimation component, either through a monocular depth estimation model or by integrating point-cloud-based depth estimation into the TSDF framework.

Ethical challenges in automated urban modeling can arise from issues of privacy, consent, representation, and legal restrictions. High-resolution 3D reconstructions risk exposing private areas or sensitive sites, raising surveillance and privacy concerns. For Google Earth Studio, local data is collected without clear consent from local inhabitants, further complicating privacy concerns. Detailed geolocation data, which can be retrieved from the pipeline, can also pose security risks if used maliciously. Moreover, geographic biases in Google Earth-based data may lead to unequal representation, favoring well-documented regions while neglecting others. Finally, cultural and legal sensitivities must be respected, ensuring compliance with local restrictions and protecting culturally significant or legally restricted areas.

Further improvements can be made to the mesh boundary at the building base. Although our method's mesh boundaries improved, we still noticed irregularities and poor-quality meshing at the base of buildings. We suspect that this issue, along with the problem of meshing sharp structural elements, can be addressed by further improving building masks, a potential extension to this research. Our improvements to SAM2 masking can also be used independently of the meshing pipeline in other applications of SAM2/GroundedSAM. This topic could constitute a separate line of research apart from Gaussian Splatting-based mesh extraction.

Assessing 3D visual quality remains challenging, particularly in the absence of a reliable ground-truth model. The 3D SSIM metric has clear limitations, as it was designed for video perceptual quality and is highly sensitive to minor frame misalignments that do not affect perceived realism. Future research in 3D quality assessment should emphasize perceptually informed approaches that integrate geometric analysis with learned visual similarity models to better capture reconstruction fidelity. Moreover, in our experiments, direct 3D geometric evaluation proved impractical due to the lack of dependable reference meshes. Available datasets revealed two main obstacles. First, segmentation ambiguity made it unclear which structural regions should be included in the reference model, as illustrated by the Colosseum data from Brizi et al. (2024). Such inconsistency caused metrics like Chamfer and Hausdorff Distance to measure segmentation differences rather than true geometric accuracy. Second, discrepancies in scale, alignment, and orientation between our outputs and reference data produced further distortions. Because of these limitations, we postpone full-reference 3D geometric evaluation to future work.

Google Earth Studio offers the advantage of generating dense multi-view images along user-specified poses, making it well-suited for Gaussian Splatting. However, despite being free, access to Google Earth Studio requires approval, and its license limits certain applications. Drone-based platforms equipped with camera sensors present a strong alternative. Among open-source multi-view aerial datasets, SpaceNeT MVOI (Weir et al., 2019) and WHU-TLC (Gao et al., 2021) are reasonable options. Yet, as raw datasets, they lack geolocation-based building scene retrieval and provide fewer input image views, limitations that could be addressed through further framework development.

Future work could leverage real-time or near-real-time data streams from Google Earth for additional imagery, or from the Google Maps Platform for cloud-based GIS data integration, building upon (Gao et al., 2025b) as our ongoing research. This would

enable adaptive 3D modeling that reflects evolving environmental and urban conditions, advancing toward practical digital twin applications. Another promising direction is the incorporation of dynamic elements such as moving vehicles and vegetation into the reconstruction pipeline, enabling temporally consistent and lifelike 4D scene representations.

Our study revealed that the quality and feasibility of 3D reconstruction depends strongly on the coherence of multi-view geometry in Google Earth Studio data and less so on image resolution. We found two types of low-resolution scenes, those with coherent geometry and those with severe distortions. Our model, like other Gaussian Splatting frameworks, failed in cases where COLMAP could not recover accurate camera poses or initialize point clouds under standard models. We suspect that these failures stem from Google Earth Studio's internal rectification process, which works well for low-resolution scenes with sufficient flight coverage, such as the White House, but fails in regions with sparse or uneven coverage, such as Arctic research stations or underdeveloped areas. This exposes a key limitation of our approach: it assumes consistent geometric relationships and accurate intrinsic calibration across views, which may not hold in inconsistently rectified or composited imagery. The entire Gaussian Splatting module fails when these assumption fails. This disadvantage is shared across all Gaussian Splatting models that depend on COLMAP initialization. Future work should focus on improving robustness to imperfect multi-view inputs by developing adaptive camera models or integrating learned priors that can infer geometric consistency even when data coverage is incomplete or distorted.

As another future research direction, we plan to pursue object-level hyperspectral Gaussian Splatting-based reconstruction, combining hyperspectral super-resolution models such as (Li et al., 2023, 2024, 2023). This approach will enable high-fidelity spectral-spatial reconstruction of complex 3D objects, enhancing material discrimination, vegetation health analysis, and urban surface characterization. Such advancements could support applications in remote sensing, precision agriculture, and environmental monitoring, where both geometric accuracy and spectral detail are critical.

7. Conclusion

The ability to generate high-quality 3D building models from minimal user input addresses a key need in digital building monitoring workflows. We developed a robust pipeline capable of automatically extracting a colored 3D mesh of a building using its address, postal code, geographic coordinates, or location name directly. Our pipeline does not require image, video, or point cloud data from the user, and instead extracts the data from Google Earth Studio using a building's name, address, or geocoding information. We benchmarked our method against GS2Mesh, which, to the best of our knowledge, is the only comparable segmentation-based Gaussian Splatting meshing algorithm publicly available. Our results demonstrate that our approach produces significantly higher-quality meshes and is significantly less prone to failure, allowing for automated extraction of the 3D mesh of any user-designated buildings given Google Earth image coverage. These capabilities open the door to scalable applications in digital twin creation, construction monitoring and automated inspection, disaster monitoring, and many other downstream tasks.

CRediT authorship contribution statement

Kyle Gao: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Dening Lu:** Writing – review & editing, Investigation. **Hongjie He:** Writing – review & editing, Investigation. **Liangzhi Li:** Writing – review & editing, Investigation. **Linlin Xu:** Writing – review & editing, Supervision, Conceptualization. **Michael A. Chapman:** Writing – review & editing, Supervision, Conceptualization. **Jonathan Li:** Writing – review & editing, Supervision, Funding acquisition, 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.

Acknowledgments

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery Grant (No. RGPIN-2022-03741, No. RGPIN-2019-06744). We would like to thank the authors of 2DGS (Huang et al., 2024) and GS2Mesh (Wolf et al., 2024), and the GitHub contributors of 2DGS+ (Ye and GitHub-Contributors, 2024) for making their codes publicly available.

Data availability

Data will be made available on request.

References

Adorjan, M., 2016. Opensfm: A Collaborative Structure-From-Motion System (Ph.D. thesis). Wien.

Alphabet Inc., 2015-2024. Google earth studio. URL: https://www.google.com/earth/studio/.

Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.P., 2021. Mip-NeRF: A multiscale representation for anti-aliasing neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5855–5864.

Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P., 2022. Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5470–5479.

Brizi, L., Giacomini, E., Di Giammarino, L., Ferrari, S., Salem, O., De Rebotti, L., Grisetti, G., 2024. VRB: A vision benchmark in rome. In: 2024 IEEE International Conference on Robotics and Automation. ICRA, IEEE, pp. 15868–15874.

Canny, J., 1986. A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. (6), 679–698.

Chen, M., Lal, D., Yu, Z., Xu, J., Feng, A., You, S., Nurunnabi, A., Shi, Y., 2024. Large-scale 3D Terrain Reconstruction Using 3D Gaussian Splatting for Visualization and Simulation. Int. Arch. Photogramm. Remote. Sens. Spatial Inf. Sci. 48, 49–54.

Chen, G., Wang, W., 2024. A survey on 3d gaussian splatting. arXiv preprint arXiv:2401.03890.

Cheng, K., Long, X., Yang, K., Yao, Y., Yin, W., Ma, Y., Wang, W., Chen, X., 2024. Gaussianpro: 3D Gaussian splatting with progressive propagation. In: International Conference on Machine Learning.

Curless, B., Levoy, M., 1996. A volumetric method for building complex models from range images. In: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques. pp. 303–312.

Derksen, D., Izzo, D., 2021. Shadow neural radiance fields for multi-view satellite photogrammetry. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1152–1161.

Esri, 2024. ArcGIS Software Suite. Redlands, CA, USA, Available: https://www.esri.com/arcgis.

Fan, L., Yang, Y., Li, M., Li, H., Zhang, Z., 2024. Trim 3D Gaussian splatting for accurate geometry representation. arXiv preprint arXiv:2406.07499.

Gao, K., Gao, Y., He, H., Lu, D., Xu, L., Li, J., 2022. NERF: Neural radiance field in 3d vision, a comprehensive review. arXiv preprint arXiv:2210.00379.

Gao, J., Liu, J., Ji, S., 2021. Rational polynomial camera model warping for deep learning based satellite multi-view stereo matching. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6148–6157.

Gao, K., Lu, D., He, H., Xu, L., Li, J., Gong, Z., 2025a. Enhanced 3D urban scene reconstruction and point cloud densification using Gaussian splatting and google earth imagery. IEEE Trans. Geosci. Remote Sens..

Gao, K., Lu, D., Li, L., Chen, N., He, H., Xu, L., Li, J., 2025b. Digital buildings analysis: 3D modeling, GIS integration, and visual descriptions using Gaussian splatting, ChatGPT/deepseek, and google maps platform. IEEE Geosci. Remote. Sens. Lett..

Guédon, A., Lepetit, V., 2024. SuGaR: Surface-aligned Gaussian splatting for efficient 3D mesh reconstruction and high-quality mesh rendering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5354–5363.

Huang, B., Yu, Z., Chen, A., Geiger, A., Gao, S., 2024. 2D gaussian splatting for geometrically accurate radiance fields. In: ACM SIGGRAPH 2024 Conference Papers. pp. 1–11.

Ji, S., Wei, S., Lu, M., 2018. Fully convolutional networks for multisource building extraction from an open aerial and satellite imagery data set. IEEE Trans. Geosci. Remote Sens. 57 (1), 574–586.

Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G., 2023. 3D Gaussian splatting for real-time radiance field rendering. ACM Trans. Graph. 42 (4), 1–14.

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.-Y., et al., 2023. Segment anything. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 4015–4026.

Li, Z., Zhang, Y., Wu, C., Zhu, J., Zhang, L., 2025. Ho-Gaussian: Hybrid optimization of 3d Gaussian splatting for urban scenes. In: European Conference on Computer Vision. Springer, pp. 19–36.

Li, J., Zheng, K., Gao, L., Ni, L., Huang, M., Chanussot, J., 2024. Model-informed multistage unsupervised network for hyperspectral image super-resolution. IEEE Trans. Geosci. Remote Sens. 62, 1–17.

Li, J., Zheng, K., Li, Z., Gao, L., Jia, X., 2023. X-shaped interactive autoencoders with cross-modality mutual learning for unsupervised hyperspectral image super-resolution. IEEE Trans. Geosci. Remote Sens. 61, 1–17.

Lin, J., Li, Z., Tang, X., Liu, J., Liu, S., Liu, J., Lu, Y., Wu, X., Xu, S., Yan, Y., et al., 2024. VastGaussian: Vast 3D Gaussians for large scene reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5166–5175.

Liu, R., Xu, R., Hu, Y., Chen, M., Feng, A., 2024a. AtomGS: Atomizing Gaussian splatting for high-fidelity radiance field. arXiv preprint arXiv:2405.12369.

Liu, S., Zeng, Z., Ren, T., Li, F., Zhang, H., Yang, J., Jiang, Q., Li, C., Yang, J., Su, H., et al., 2024b. Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection. In: European Conference on Computer Vision. Springer, pp. 38–55.

Lorensen, W.E., Cline, H.E., 1998. Marching cubes: A high resolution 3D surface construction algorithm. In: Seminal Graphics: Pioneering Efforts that Shaped the Field. pp. 347–353.

Mallick, S.S., Goel, R., Kerbl, B., Steinberger, M., Carrasco, F.V., De La Torre, F., 2024. Taming 3DGS: High-quality radiance fields with limited resources. In: SIGGRAPH Asia 2024 Conference Papers. pp. 1–11.

Marí, R., Facciolo, G., Ehret, T., 2022. Sat-NeRF: Learning multi-view satellite photogrammetry with transient objects and shadow modeling using rpc cameras. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1311–1321.

Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R., 2021. NeRF: Representing scenes as neural radiance fields for view synthesis. Commun. ACM 65 (1), 99–106.

Müller, T., Evans, A., Schied, C., Keller, A., 2022. Instant neural graphics primitives with a multiresolution hash encoding. ACM Trans. Graph. 41 (4), 1–15.

Netravali, A.N., 2013. Digital Pictures: Representation, Compression, and Standards. Springer.

Oechsle, M., Peng, S., Geiger, A., 2021. UNISURF: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5589–5599.

Ramer, U., 1972. An iterative procedure for the polygonal approximation of plane curves. Comput. Graph. Image Process. 1 (3), 244–256.

Ravi, N., Gabeur, V., Hu, Y.-T., Hu, R., Ryali, C., Ma, T., Khedr, H., Rädle, R., Rolland, C., Gustafson, L., et al., 2024. SAM 2: Segment anything in images and videos. arXiv preprint arXiv:2408.00714.

Reality, C., 2024. RealityCapture Software. Bratislava, Slovakia, Available: https://www.capturingreality.com.

Rematas, K., Liu, A., Srinivasan, P.P., Barron, J.T., Tagliasacchi, A., Funkhouser, T., Ferrari, V., 2022. Urban radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12932–12942.

Ren, T., Liu, S., Zeng, A., Lin, J., Li, K., Cao, H., Chen, J., Huang, X., Chen, Y., Yan, F., et al., 2024. Grounded SAM: Assembling open-world models for diverse visual tasks. arXiv preprint arXiv:2401.14159.

Scharstein, D., Szeliski, R., 2002. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. Int. J. Comput. Vis. 47, 7–42.

Schonberger, J.L., Frahm, J.-M., 2016. Structure-from-motion revisited. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4104–4113.

Snavely, N., 2008. Bundler: Structure from motion (sfm) for unordered image collections. URL: http://phototour.cs.washington.edu/bundler/.

Sølund, T., Buch, A.G., Krüger, N., Aanæs, H., 2016. A large-scale 3D object recognition dataset. In: 2016 Fourth International Conference on 3D Vision. 3DV, IEEE, pp. 73–82.

Solutions, H.G., 2024. ENVI Software Suite. Boulder, CO, USA, Available: https://www.l3harrisgeospatial.com/Software-Technology/ENVI.

Tancik, M., Casser, V., Yan, X., Pradhan, S., Mildenhall, B., Srinivasan, P.P., Barron, J.T., Kretzschmar, H., 2022. Block-NeRF: Scalable large scene neural view synthesis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8248–8258.

Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P., 2004. Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. 13 (4), 600–612.

Wang, Y., Han, Q., Habermann, M., Daniilidis, K., Theobalt, C., Liu, L., 2023. NeuS2: Fast learning of neural implicit surfaces for multi-view reconstruction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 3295–3306.

Wang, P., Liu, L., Liu, Y., Theobalt, C., Komura, T., Wang, W., 2021. NeuS: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. arXiv preprint arXiv:2106.10689.

Wang, Y., Tang, X., Ma, J., Zhang, X., Zhu, C., Liu, F., Jiao, L., 2024. Pseudo-viewpoint regularized 3D Gaussian splatting for remote sensing few-shot novel view synthesis. In: IGARSS 2024-2024 IEEE International Geoscience and Remote Sensing Symposium. IEEE, pp. 8660–8663.

Weir, N., Lindenbaum, D., Bastidas, A., Etten, A.V., McPherson, S., Shermeyer, J., Kumar, V., Tang, H., 2019. Spacenet MVOI: A multi-view overhead imagery dataset. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 992–1001.

Wolf, Y., Bracha, A., Kimmel, R., 2024. GS2Mesh: Surface reconstruction from Gaussian splatting via novel stereo views. In: European Conference on Computer Vision. Springer, pp. 207–224.

Xiangli, Y., Xu, L., Pan, X., Zhao, N., Rao, A., Theobalt, C., Dai, B., Lin, D., 2022. BungeeNeRF: Progressive neural radiance field for extreme multi-scale scene rendering. In: Proceedings of the European Conference on Computer Vision. pp. 106–122.

Ye, C., GitHub-Contributors, 2024. 2d-gaussian-splatting-great-again. GitHub repository, URL: https://github.com/hugoycj/2d-gaussian-splatting-great-again/tree/main.

Ye, Z., Li, W., Liu, S., Qiao, P., Dou, Y., 2024. Absgs: Recovering fine details in 3d gaussian splatting. In: Proceedings of the 32nd ACM International Conference on Multimedia. pp. 1053–1061.

Yu, Z., Sattler, T., Geiger, A., 2024. Gaussian opacity fields: Efficient adaptive surface reconstruction in unbounded scenes. ACM Trans. Graph. 43 (6), 1–13.

Zeng, K., Wang, Z., 2012. 3D-SSIM for video quality assessment. In: 2012 19th IEEE International Conference on Image Processing. IEEE, pp. 621–624.

Zhang, Z., Hu, W., Lao, Y., He, T., Zhao, H., 2024. Pixel-GS: Density control with pixel-aware gradient for 3d gaussian splatting. In: European Conference on Computer Vision. Springer, pp. 326–342.

Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O., 2018. The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 586–595.

Zhao, H., Zhou, H., Zhang, Y., Chen, J., Yang, Y., Zhao, Y., 2023. High-frequency stereo matching network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1327–1336.

Zhou, Q.-Y., Park, J., Koltun, V., 2018. Open3D: A modern library for 3D data processing. arXiv preprint arXiv:1801.09847.

Zhou, H., Shao, J., Xu, L., Bai, D., Qiu, W., Liu, B., Wang, Y., Geiger, A., Liao, Y., 2024. HUGS: Holistic urban 3D scene understanding via Gaussian splatting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21336–21345.