

3D Scene Change Modeling With Consistent Multi-View Aggregation

Supplementary Material

Zirui Zhou¹ Junfeng Ni^{1,2} Shujie Zhang¹ Yixin Chen^{2,†,✉} Siyuan Huang^{2,✉}

[†] Project lead ✉ Corresponding author

¹ Tsinghua University ² State Key Laboratory of General Artificial Intelligence, BIGAI

<https://zr-zhou0o0.github.io/SCaR3D/>

S1. Details on Our Methodology

S1.1. Problem Setup

For each changed object $\mathcal{O} = \{o_1, o_2, \dots, o_n\}$ observed under viewpoint i , the corresponding change mask C_i is defined as the union of individual object masks: $C_i = c_1 \cup c_2 \cup \dots \cup c_n$. To facilitate downstream tasks such as continual scene reconstruction, we explicitly separate \mathcal{C} into two categories: pre-change masks \mathcal{C}_{pre} , which localize the changed objects in the pre-change state, and post-change masks \mathcal{C}_{post} , which annotate them in the post-change state.

We consider five primary types of geometric changes: addition, removal, translation, rotation, and their complex combinations. Other variations that do not alter scene geometry, such as appearance modifications, lighting differences, or shadows, are outside the scope of this paper, and we leave them for future work.

S1.2. COLMAP Registration

Following the image registration strategies in [2, 3], we jointly register the pre-change images \mathcal{I}_{pre} , post-change images \mathcal{I}_{post} , and test images \mathcal{I}_{test} within a single COLMAP process to estimate camera poses in a unified coordinate system. Specifically, we adopt the pinhole camera model in the feature extractor, apply an exhaustive matcher for feature correspondence, and employ the mapper to reconstruct a sparse point cloud.

It is worth noting that changes in scene objects, especially among existing benchmarks, do not hinder the registration process. However, compared to standard dense 360° navigation, our unregularized camera trajectories may lead to erroneous pose estimations for certain frames due to insufficient feature points, even though these frames remain easily recognizable to humans. Addressing such cases remains an open direction for future improvements.

S1.3. 2D Difference

Limitations of 2D Feature Difference

1. **Limited computational resources.** Due to the resolution constraints of foundation model encoders, feature maps must be upsampled to match the image size. This upsampling introduces uncertainty around mask boundaries. The attention mechanism of EFFICIENTSAM and computational constraints also leads to performance degradation in complex scenes.
2. **Object confusion.** Semantic segmentation and vision-language models tend to prioritize object categories over fine-grained appearance. As a result, different instances of the same category (e.g., two books with distinct titles) often yield highly similar features, causing confusion in distinguishing them.
3. **Failure in overlapping or subtle changes.** When objects undergo minor variations, such as slight rotations, or the opening and closing of a box, the unchanged regions are typically not detected as differences. This limitation arises because feature differences are computed at the pixel level, without a holistic consideration of object integrity.

S1.4. 3D Difference

Definition of a Gaussian Being Seen We define the visibility of the i -th Gaussian from the k -th viewpoint according to two criteria:

1. **Image boundary.** Let the projection of the Gaussian center μ_i onto the k -th image plane be p_k^i . If p_k^i falls outside the image boundary, the Gaussian is regarded as not visible in that view.
2. **Occlusion.** To account for occlusions caused by scene geometry, we assume that all pixels in the k -th view are inside the mask, and accumulate per-pixel contributions to the Gaussian using transmission and opacity as weights. Gaussians that receive only negligible accumulated weights are considered occluded.

This definition of visibility is subsequently used to compute n_i^{seen} for voting normalization, and also serves as a criterion in the multi-view pruning procedure.

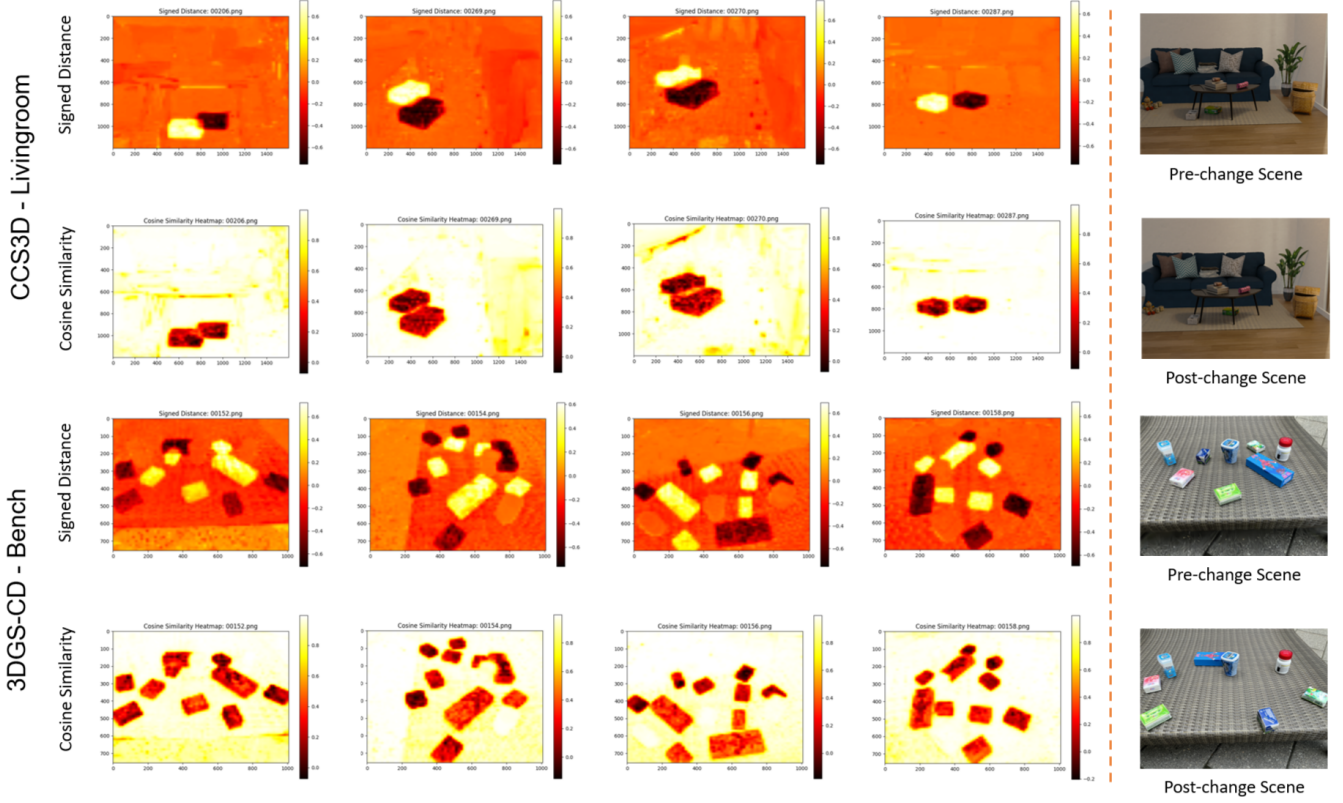


Figure S1. **Comparison between signed distance and cosine similarity.** Both formulations yield comparable performance, indicating that signed distance can serve as an alternative representation to cosine similarity.

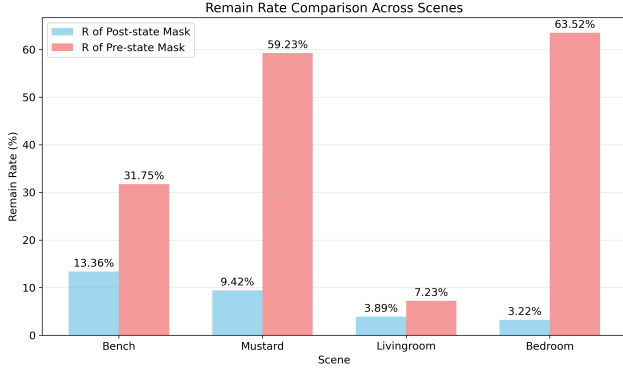


Figure S2. **Remain rate comparison.** Quantitative comparison of the proportion of preserved Gaussians when pruning with pre-change masks versus post-change masks.

Multi-view Pruning The pruning procedure is summarized in Algorithm 1.

Process of Segmentation Validation We validate the 3D difference maps using segmentation results to obtain the final masks. For each view, the 3D difference map is first projected onto the image plane. We then identify all segmenta-

tion masks that intersect with the projected differences. To generate candidate masks, we overlay a grid on the image and use grid nodes falling inside the masks as prompt points for EfficientSAM. Since the segmentation quality of EfficientSAM is sensitive to the density of prompt points, we adopt both sparse and dense grids to construct the prompt set for each scene. We further observe that batching prompt points or processing images in patches significantly improves segmentation accuracy, albeit with increased computation time. Finally, for each connected component in the 3D difference map, we select the segmentation mask with the highest IoU as the final validated mask C .

S2. Details on CCS3D Dataset

Each scene in the **CCS3D** dataset consists of seven folders: train-pre, train-post, test-pre (pre-change scene rendered from test views), test-post (post-change scene rendered from test views), gt_pre, gt_post, and sparse point cloud reconstruction results. All images are rendered at a resolution of 1600×1200 pixels. For the Livingroom, Desk, and Bookcase scenes, 100 pre-change training images are provided, while the Bedroom scene contains 200 images to better capture its com-

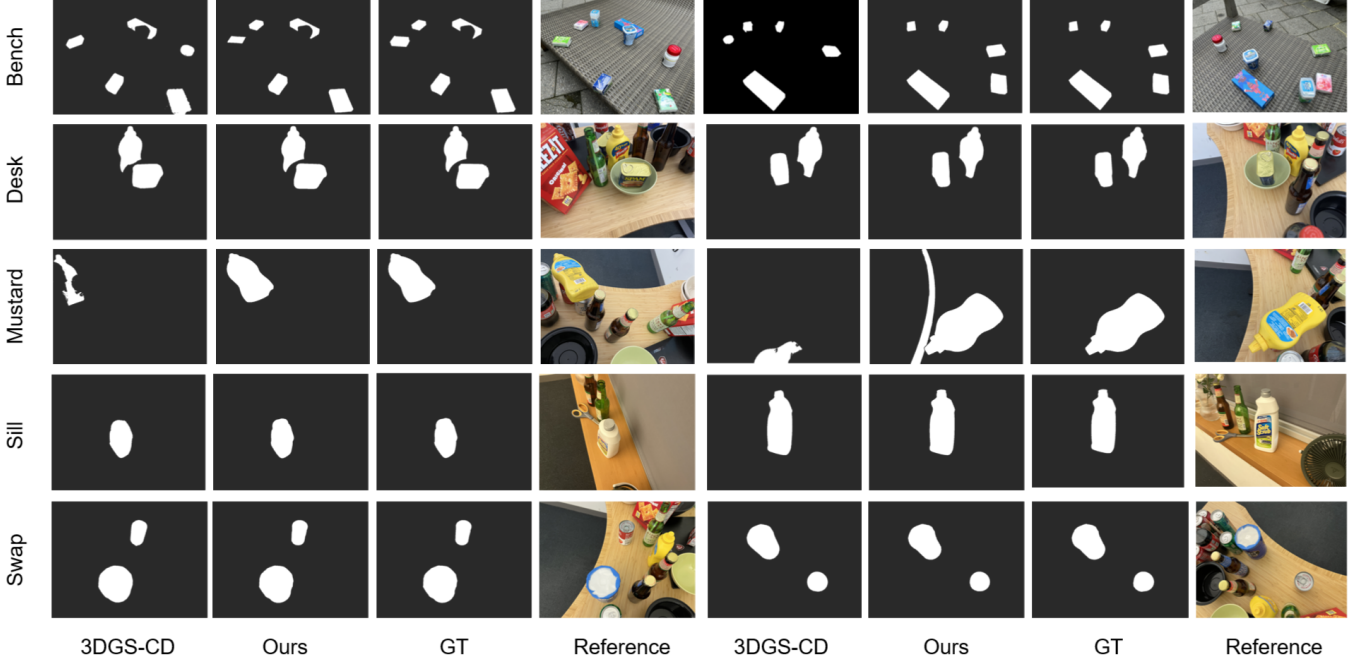


Figure S3. **Qualitative results on 3DGS-CD.** Visual comparison of pre-change and post-change mask pruning on the 3DGS-CD benchmark. Each row corresponds to one scene.

Algorithm 1 Multi-view Consistent Pruning

```

1: Input: Gaussians  $\mathcal{G} = \{\mu_i\}$ , post-change masks  $\{M^k\}_{k=1}^{n_{\text{post}}}$ , threshold  $\tau$ 
2: Output: Pruned set of Gaussians  $\mathcal{G}'$ 
3: for each Gaussian  $g_i \in \mathcal{G}$  do
4:    $n_i^{\text{seen}} \leftarrow 0, n_i^{\text{out}} \leftarrow 0$ 
5:   for each view  $k = 1, \dots, n_{\text{post}}$  do
6:     Project  $\mu_i$  to pixel  $p_k^i \leftarrow \pi_k(\mu_i)$ 
7:     if  $p_k^i$  inside image bounds then
8:        $n_i^{\text{seen}} \leftarrow n_i^{\text{seen}} + 1$ 
9:       if  $M^k(p_k^i) = 0$  then  $\triangleright$  outside 2D mask
10:         $n_i^{\text{out}} \leftarrow n_i^{\text{out}} + 1$ 
11:       end if
12:     end if
13:   end for
14:   if  $n_i^{\text{out}} > \tau \cdot n_i^{\text{seen}}$  then
15:     prune  $g_i$ 
16:   else
17:     keep  $g_i$  in  $\mathcal{G}'$ 
18:   end if
19: end for
20: return  $\mathcal{G}'$ 

```

plexity. The `train-post` folder contains 4–5 images per scene. Rendering is performed using the Cycles engine with one glossy bounce and one diffuse bounce, and all images

are captured under consistent lighting conditions.

The dataset contains four types of scene changes: addition, removal, translation, and rotation. We manually ensure that all changes are visible in both the `train-post` and test sets. Camera trajectories follow a coarse-to-fine strategy: the first pass captures an overview of the entire scene, followed by detailed views of objects and corners.

S3. Experimental Details

S3.1. Implementation Details

We train the pre-change Gaussians \mathcal{G}_{pre} for 30,000 iterations using the Adam optimizer with an initial position learning rate of 1.6×10^{-4} . All images are resized to 1200×1600 , and camera poses are estimated with COLMAP. Experiments are conducted on a single NVIDIA RTX 3090 GPU using PyTorch 2.4.1 with CUDA 12.8. The overall change detection process takes approximately 30 seconds per scene.

S3.2. Inference Time Comparisons

Our method achieves the highest detection accuracy while maintaining relatively fast inference. For efficiency evaluation, we only compute the change detection module on the Livingroom scene in the **CCS3D** dataset.

Among the baselines, 2D change detection methods such as pixel difference and feature difference are the fastest but yield fragmented and unreliable masks. The learning-based

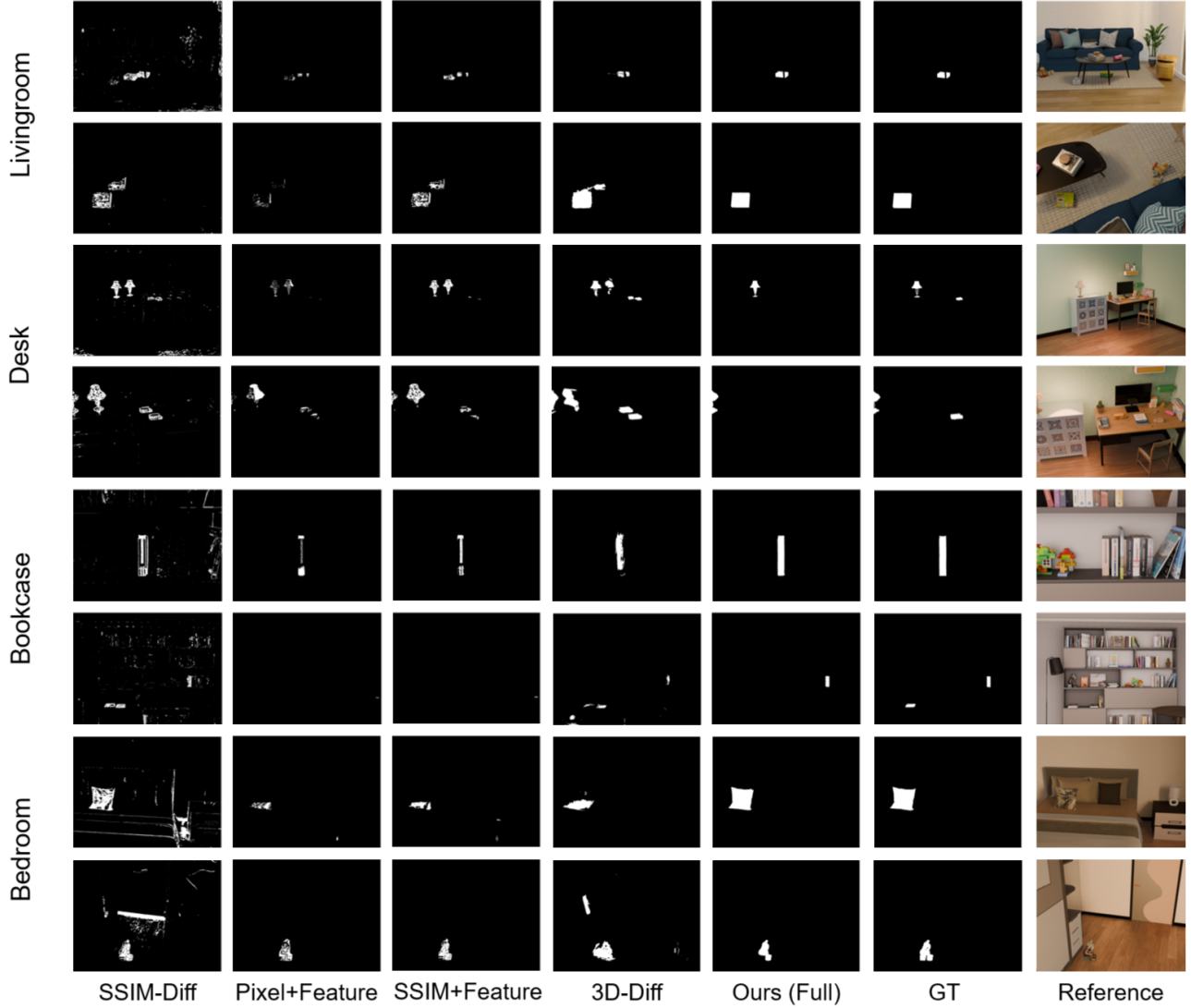


Figure S4. **Ablation study.** Qualitative evaluation of different choices of 2D differences and 3D differences.

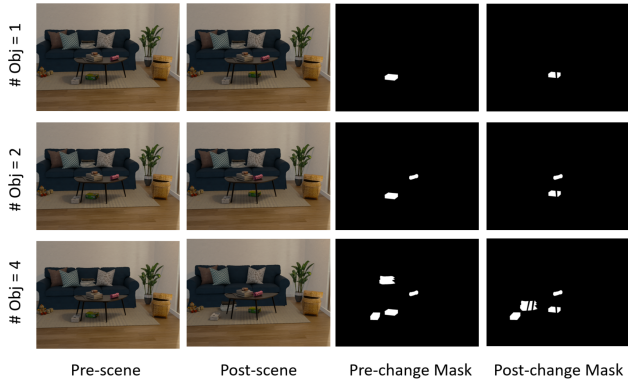


Figure S5. **Visualization of the controlled dataset.** Example from the Livingroom scene showing a translation case involving objects 1, 2, and 4.

method MV3DCD [1] requires significantly longer inference time. Similarly, 3DGS-CD involves additional time overhead due to its pose estimation step.

S3.3. Controlled Dataset

We also release a controlled evaluation subset of **CCS3D**, in which the number of changed objects in the Livingroom scene ranges from 1 to 4, as illustrated in Fig. S5.

S4. Additional Experimental Results

S4.1. Signed Distance

To further evaluate the effectiveness of signed distance, we conduct experiments comparing cosine similarity with signed distance, as shown in Fig. S1. The signed dis-

tance robustly distinguishes between object additions and removals.

S4.2. Remain Rate

In the Change Direction Distinction step, we define a remain rate R to separate pre-change and post-change masks. We evaluate the remain rate R across both synthetic datasets (Livingroom and Bedroom in the **CCS3D** dataset) and real-world datasets (Bench and Mustard in the 3DGS-CD dataset), as shown in Fig. S2.

S4.3. Experiments on CCS3D dataset

Extended evaluation results on the **CCS3D** dataset are reported in Tab. S1. In addition to the F1-score and IoU presented in the main text, we further include precision and recall for a more comprehensive assessment.

S4.4. Experiments on 3DGS-CD dataset

Qualitative results on the 3DGS-CD dataset are shown in Fig. S3, demonstrating the generalization ability of our method to real-world data.

S4.5. Ablation Study

Additional qualitative ablation results are presented in Fig. S4.

References

- [1] Chamuditha Jayanga Galappaththige, Jason Lai, Lloyd Windrim, Donald Dansereau, Niko Sunderhauf, and Dimity Miller. Multi-view pose-agnostic change localization with zero labels. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pages 11600–11610, 2025.
- [2] Rui Huang, Binbin Jiang, Qingyi Zhao, William Wang, Yuxiang Zhang, and Qing Guo. C-nerf: Representing scene changes as directional consistency difference-based nerf. *arXiv preprint arXiv:2312.02751*, 2023.
- [3] Ziqi Lu, Jianbo Ye, and John Leonard. 3dgs-cd: 3d gaussian splatting-based change detection for physical object rearrangement. *IEEE Robotics and Automation Letters*, 2025.

Table S1. **Extended quantitative change detection results on the CCS3D dataset.** In addition to F1-score and IoU reported in the main text, we also provide precision and recall for completeness.

Method	Livingroom		Desk		Bookcase		Bedroom		Average	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Pixel-Diff	0.384	0.295	0.348	0.481	0.235	0.537	0.244	0.517	0.303	0.458
Feature-Diff	0.380	0.475	0.375	0.683	0.344	0.305	0.786	0.679	0.436	0.538
CL-Splat	0.783	0.830	0.409	0.937	0.307	0.294	0.501	0.615	0.500	0.669
MV3DCD	0.449	0.629	0.243	0.374	0.339	0.769	0.446	0.886	0.369	0.664
3DGS-CD	0.876	0.925	0.622	0.464	0.438	0.542	0.089	0.490	0.506	0.605
Ours	0.951	0.960	0.963	0.487	0.494	0.379	0.937	0.892	0.836	0.680