

# E-3DGS: Event-Based Novel View Rendering of Large-Scale Scenes Using 3D Gaussian Splatting

## Supplementary Material

This supplement provides additional details and insights into the methods and experiments discussed in the main paper. In Sec. I, we elaborate on our frustum-based initialization, explaining the sampling strategy and how it ensures effective Gaussian placement in the scene. Sec. II provides further details on our pose refinement, specifically the use of Gram-Schmidt orthogonalization to maintain valid transformations during optimization. In Sec. III, we analyze the camera pose noise in the E-3DGS-Real dataset and describe the process we use to simulate realistic pose perturbations for the E-3DGS-Synthetic-Hard dataset. Sec. IV outlines the implementation details, including adjustments to the original 3DGS training schedule to improve convergence. Sec. V covers our evaluation, highlighting the measures we take to ensure reliable results, particularly for the ablation studies. Finally, we present a comprehensive comparison in Sec. VI showcasing additional visual results and ablation studies on the E-3DGS-Real, E-3DGS-Synthetic, and E-3DGS-Synthetic-Hard datasets. These experiments expand on the results from the main paper and further demonstrate the effectiveness of our method across different scenarios.

### I. Frustum-Based Initialization

As described in Sec. 4.2 of the main paper, our approach involves initializing a fixed number of Gaussians, denoted as  $N_g$ . If we have  $N_t$  camera poses, we distribute the Gaussians across these poses, resulting in  $N_g/N_t$  Gaussians being initialized for each pose. The initialization process begins by sampling points within the camera frustum in normalized device coordinates (NDC). However, instead of uniformly sampling all three coordinates  $(x, y, z)$  in NDC, we adopt a different strategy for depth (z-axis).

We observe that when depth was sampled directly in NDC, most Gaussians would cluster very close to the near plane ( $z_{\text{near}}$ ), leading to poor scene coverage. To address this, we sample the depth uniformly in camera coordinates between  $z_{\text{near}}$  and  $z_{\text{far}}$ . This ensures a more even distribution of Gaussians across the entire depth range.

Once the depth is sampled in camera coordinates, it is converted into NDC. Next, the  $x$  and  $y$  coordinates are sampled uniformly in NDC. With  $x$ ,  $y$ , and  $z$  values now in NDC, we un-project them back into the world coordinates. This conversion gives us the final positions for the Gaussians in the 3D scene. Next, the entire process is repeated for each camera frustum associated with the given poses  $P_t$ , ensuring a comprehensive initialization across all views. Therefore, the distribution of Gaussians is effectively tied to

the observable scene regions.

### II. Pose Refinement and Gram-Schmidt Orthogonalization

In Sec. 4.5 of the main paper, we introduce our approach to pose refinement, where the refined pose  $P'_t$  is modeled as  $P'_t = P_t^e P_t$ , with  $P_t^e$  being an error correction transform. Rather than directly optimizing  $P_t^e$  as a  $3 \times 3$  matrix, we represent it using two rotation vectors  $r_1$  and  $r_2$  and a translation vector  $T$ , following the method of Hempel et al. [6]. This representation allows us to ensure that  $P_t^e$  remains a valid transformation matrix during optimization.

To maintain the orthogonality of the rotation matrix, we apply Gram-Schmidt orthogonalization to  $r_1$  and  $r_2$  to compute the final rotation matrix  $R = [r'_1, r'_2, r'_3]$ . The process is as follows:

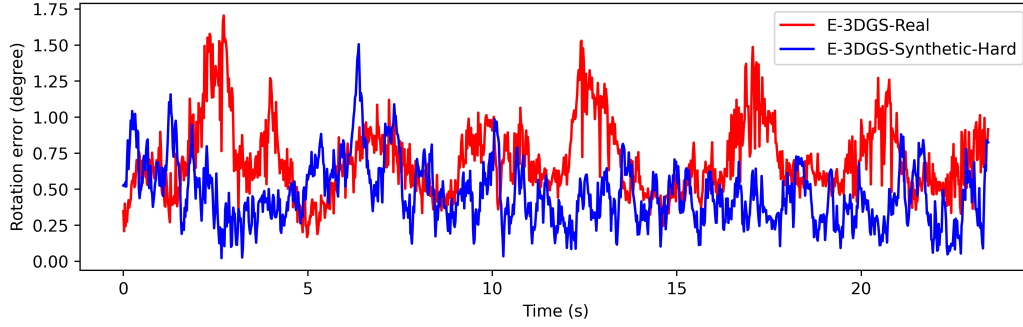
$$\begin{aligned} r'_1 &= \frac{r_1}{\|r_1\|}, \\ r'_2 &= \frac{r_2 - (r'_1 \cdot r_2)r'_1}{\|r_2 - (r'_1 \cdot r_2)r'_1\|}, \\ r'_3 &= r'_1 \times r'_2, \text{ and} \\ P_t^e &= \begin{bmatrix} | & | & | & | \\ r'_1 & r'_2 & r'_3 & T \\ | & | & | & | \\ 0 & 0 & 0 & 1 \end{bmatrix}. \end{aligned} \tag{12}$$

Here,  $r'_1$  is the normalized version of  $r_1$ , and  $r'_2$  is obtained by subtracting the projection of  $r_2$  onto  $r'_1$  and normalizing the result. The third vector  $r'_3$  is calculated as the cross product of  $r'_1$  and  $r'_2$ , ensuring that the resulting rotation matrix is orthogonal. The final error correction matrix  $P_t^e$  is then constructed using these orthogonal vectors and the translation vector  $T$ .

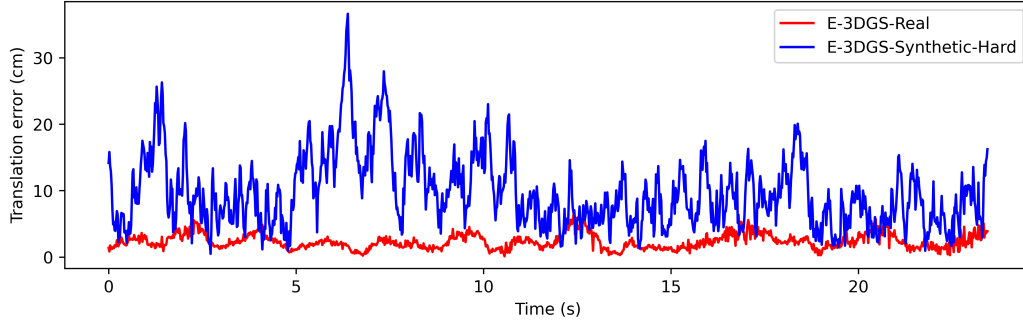
This approach guarantees that the pose refinement remains valid throughout the optimization process, contributing to the stability and accuracy of our method.

### III. Pose Perturbation in E-3DGS-Synthetic-Hard

As described in Sec. 5.2 of the main paper, we provide the E-3DGS-Synthetic-Hard dataset that differs from E-3DGS-Synthetic in two aspects: 1) The camera speed is highly varied and 2) the camera extrinsics exhibit noise similar in characteristics to the noise observed in the real data. To quantify the camera pose noise in the E-3DGS-Real dataset,



(a) Rotation errors for both E-3DGS-Real and E-3DGS-Synthetic-Hard show a similar error distribution.



(b) Larger translation errors are applied to E-3DGS-Synthetic-Hard, compared to those in E-3DGS-Real, to account for the larger scene size and ensure a sufficiently challenging difficulty level for meaningful ablation studies.

Figure I. Comparison of estimated pose errors in the E-3DGS-Real dataset versus the synthetically introduced errors in the E-3DGS-Synthetic-Hard dataset. The synthetic perturbations are generated using an Ornstein–Uhlenbeck process to match the time-correlated nature and variance of the real data.

we compare the refined training camera trajectories with the initial trajectories. Our analysis reveals that these errors are time-correlated. Based on this observation and by examining the scale of these errors, we introduce synthetic perturbations in the E-3DGS-Synthetic dataset using a random walk with decay, specifically the Ornstein–Uhlenbeck process [21], which ensures the perturbations have zero mean while remaining time-correlated.

We calibrate the variance of the synthetic perturbations to match the rotation errors observed in the real data. For translation, we apply a higher level of perturbation, given that the synthetic scenes are significantly larger in scale than the real data. This adjustment ensures that translation errors are proportionally scaled, creating a comparable difficulty level for the ablation studies. The noise patterns are illustrated in Fig. I.

## IV. Implementation Details

Our codebase is based on 3DGS [9]. We train the method for  $6 \cdot 10^4$  instead of  $3 \cdot 10^4$  iterations, allowing the pose refinement to converge. The original paper performs both, densification and opacity resets of the Gaussians until

$1.5 \cdot 10^4$  iterations. In our case, we perform opacity resets until  $3 \cdot 10^4$  and densification until  $5 \cdot 10^4$  iterations. From our analysis—while opacity resets are important to remove floaters—they also hamper the reconstruction quality. Therefore, once the scene is reasonably converged, we stop resetting opacity and only densify the scene to get better reconstruction.

Furthermore, 3DGS uses the fixed threshold value  $2 \cdot 10^{-4}$  to decide whether a Gaussian should be split up during the densification. We start the optimization with the same value, however, we linearly decrease it to  $4 \cdot 10^{-5}$  over  $4 \cdot 10^4$  iterations. First, this allows our method to refine the poses with larger Gaussians, providing more support, and second, reduce the threshold in later stages to obtain a more detailed reconstruction. We initialize  $N_g = 5 \cdot 10^4$  Gaussians in all our trainings.

In the experiments with pose refinement, we restrict the number of spherical harmonics to one, as it allows for better pose refinement [8, 15]. For the experiments with perfect poses, we follow the original 3DGS approach and use three spherical harmonics. In all experiments, except those conducted with the EventNeRF dataset [25], we consistently use  $N_{\max}=10^6$  events for the window size. As sequences of

the latter are very short and do not contain enough events for such large windows, we use  $N_{\max}=10^5$  for them. Training the full method takes one to two hours with a single NVIDIA GeForce RTX 3090, depending on the scene size.

## V. Further Evaluation Details (Ablations)

To ensure the reliability of the results, all ablation studies are conducted four times, with evaluation metrics averaged to provide more accurate insights and minimize the effects of coincidence. For the E-3DGS-Synthetic-Hard dataset, where the camera poses are perturbed, direct evaluation is not feasible due to slight misalignments between the learned 3D scene and the ground truth. To correct this, we first freeze the Gaussians and then refine the test poses with a small learning rate to ensure proper convergence. This alignment process allows the test views to match the ground truth accurately, enabling precise evaluation.

## VI. Additional Comparisons and Ablations

In this section, we expand on the main paper experiments by showing additional results on E-3DGS-Real, E-3DGS-Synthetic, and E-3DGS-Synthetic-Hard datasets. Fig. II demonstrates the performance of E-3DGS in comparison to Deblur-GS [28], E2VID [23]+3DGS [9] and EventNeRF [25] on the E-3DGS-Real dataset. These baselines exhibit severe artifacts such as blur, floaters and noise. In the same figure, we also demonstrate the impact of the key components of our method. Removing  $L_{\text{iso}}$  leads to increased amounts of floaters and other artifacts. As the captured camera poses contain noise, pose refinement (PR) is crucial to achieve accurate results. Hence, without it, the model cannot produce accurate predictions, resulting in severe artifacts and blurriness. However, the model without the adaptive windows (AW) shows similar performance to the full model. That is likely due to the overall uniformity of the camera speeds in the used dataset, which diminishes the potential impact of adaptive event windows.

In Fig. III, we compare E-3DGS against EventNeRF [25] and E2VID [23]+3DGS [9] on E-3DGS-Synthetic dataset. Both baselines perform poorly: While E2VID+3DGS captures the edges and the general structure, it struggles with color representation, and EventNeRF reconstruction is much noisier and blurrier compared to our method. In contrast, our E-3DGS outperforms them, showing clear and sharp novel views with accurate color representation. Some issues are still observable but are mostly in less supervised areas, *e.g.*, on the roof in ScienceLab or Subway scenes.

Lastly, Fig. IV visualizes results of the ablation study on the E-3DGS-Synthetic-Hard dataset. In comparison to E-3DGS-Synthetic, this dataset has artificially added camera extrinsics noise, which we describe in Sec III, and drastically increased camera speed variation (Sec. 5.2). While

these changes make obtaining high reconstruction quality more difficult, our full method still works well, outperforming all ablated models. As on the E-3DGS-Real, removing  $L_{\text{iso}}$  results in severe artifacts (*e.g.*, in the first view of Company or in the second view of Subway). E-3DGS-Synthetic-Hard dataset has camera pose noise, and, hence, using pose refinement (PR) is important, as removing it results in blurriness and artifacts. Removing the adaptive event windows (AW) leads to deterioration; *e.g.*, the method without AW exhibits artifacts on the sofa in the first view of the Company sequence that are absent in the results of the full method. It is also noteworthy that while all ablated models struggle with the second view of the Subway sequence, the full method, nevertheless, achieves a better result: The structure is clearer and more recognizable with fewer artifacts.

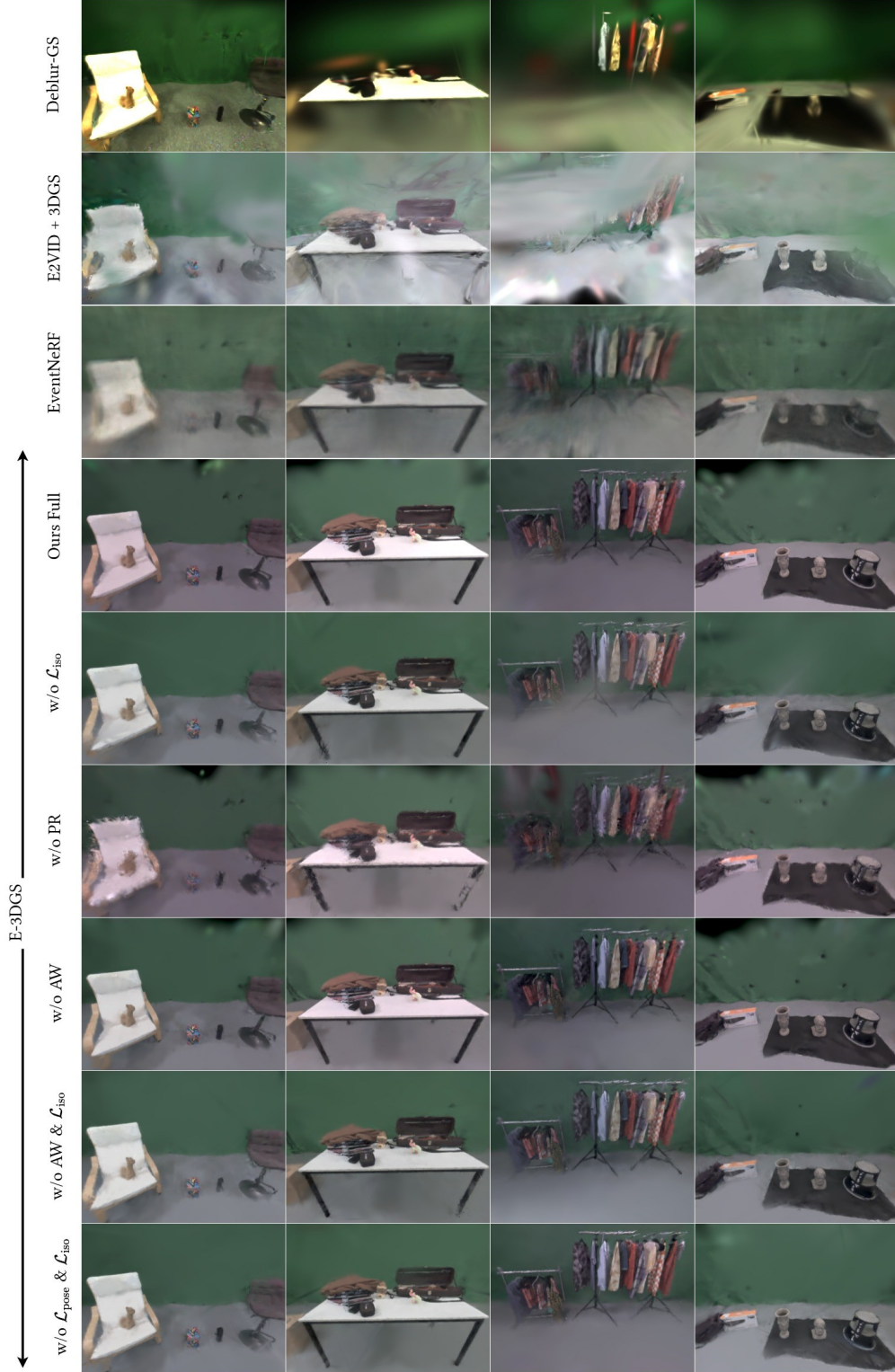


Figure II. Comparison of E-3DGS against the baselines and ablation study on E-3DGS-Real. As observed in the main paper, Deblur-GS, E2VID + 3DGS, and EventNeRF exhibit issues such as blurring, floaters, and noise. Notably, the ablation study highlights the impact of removing key components. Removing  $\mathcal{L}_{iso}$  leads to an increase in floaters and artifacts. In contrast, the experiment without adaptive event windows (AW) shows little difference in performance. This is likely due to the relatively consistent camera speeds in this dataset, which reduce the potential benefits of the adaptive event window.



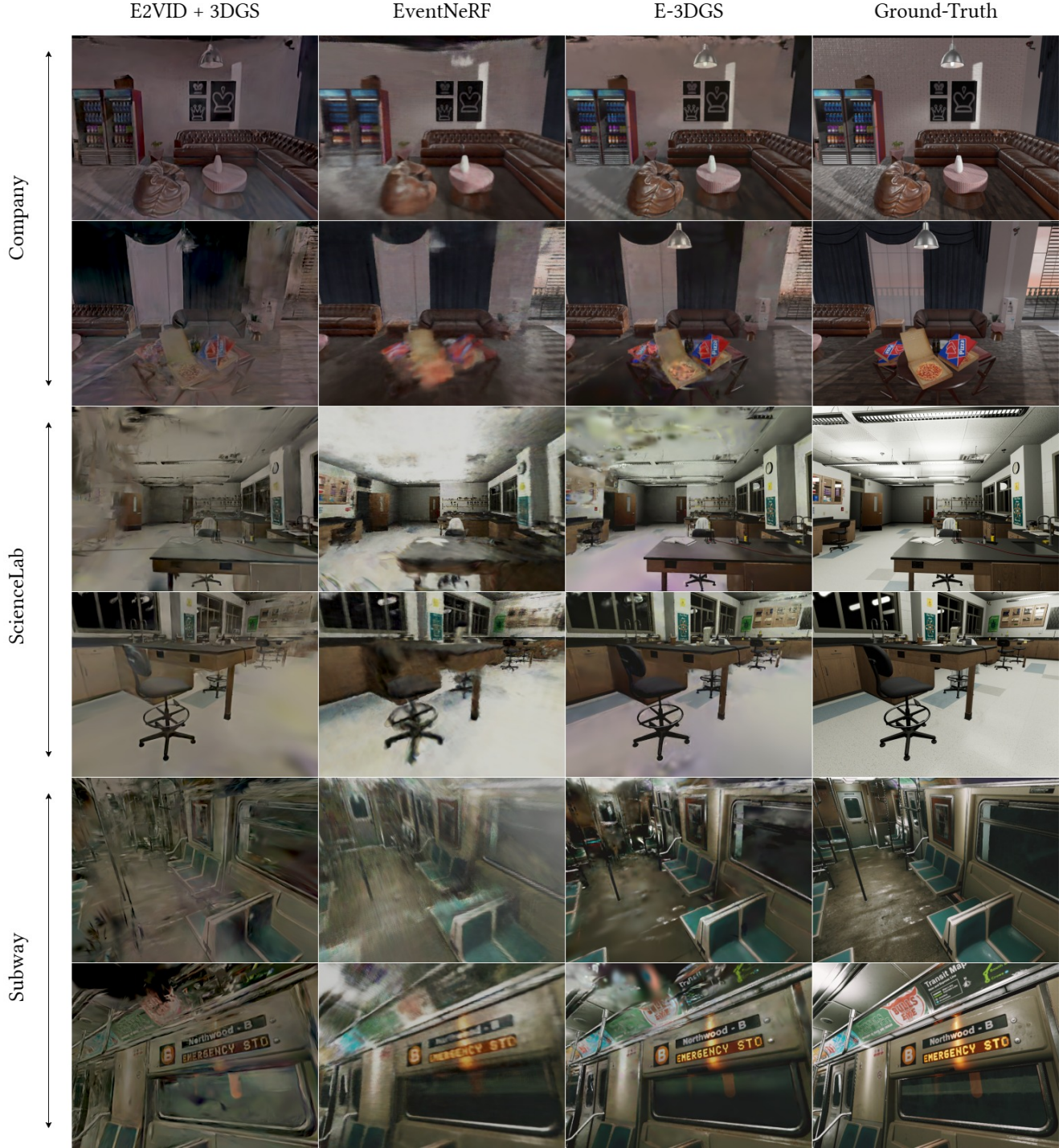


Figure III. Comparison of E-3DGS vs. baselines on the E-3DGS-Synthetic dataset. As observed in the main paper, E2VID + 3DGS struggles with poor color reconstruction but captures edges and structure reasonably well. EventNeRF suffers from noise and a lack of sharpness. In contrast, our method delivers clear details and accurate colors, with issues mainly confined to less observed areas, such as the roof. Best viewed with zoom.



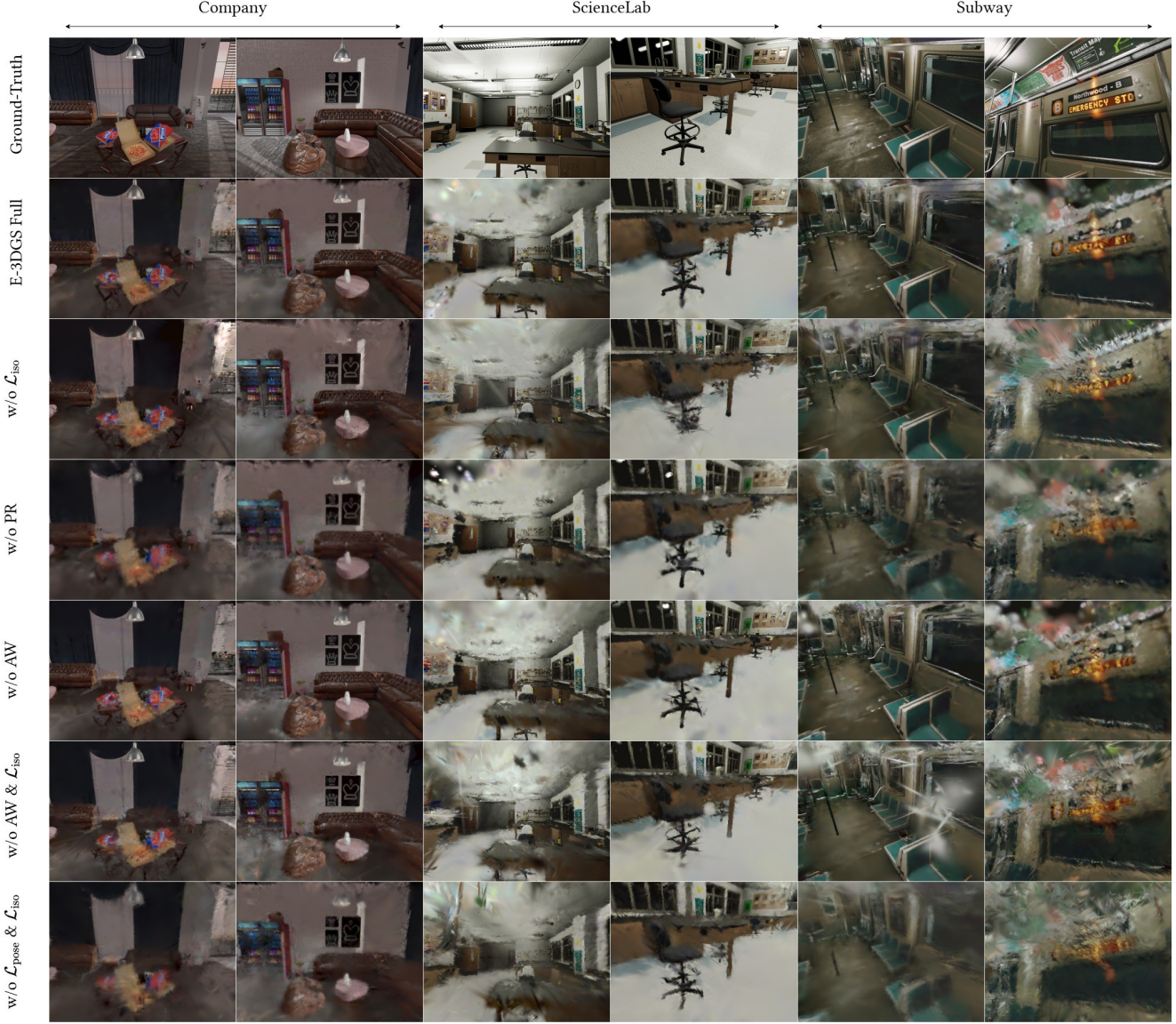


Figure IV. Ablation study of E-3DGS on the E-3DGS-Synthetic-Hard dataset. The increased difficulty of this dataset leads to overall performance deterioration compared to E-3DGS-Synthetic, but our full method still performs well. The version without the adaptive event window (AW) is closest to the full method but shows more artifacts. For example, in the first column of the Company sequence, the sofa shows some artifacts in the AW-removed version that are absent in the full method. Similar minor artifacts are visible elsewhere. The second column of the Subway sequence is interesting, as all versions struggle with reconstructing it. Even so, the full method demonstrates a better structure and fewer artifacts than the others.