Ghost-FWL: A Large-Scale Full-Waveform LiDAR Dataset for Ghost Detection and Removal (Supplementary Material)

Contents

A. Overview of Supplementary Material

This supplementary material provides additional details and further experimental results that complement the content presented in the main paper. Please also refer to the **supplementary video** at https://keio-csg.github.io/Ghost-FWL/, which presents the SLAM results of the comparative methods and our proposed method. For clarity, we use red to denote references corresponding to the main paper, and blue to denote those corresponding to these supplementary materials.

Table 1. Ghost-FWL statistics by scene.

Scene	Frames	Location
001	2500	Indoor
002	2500	Indoor
003	2749	Indoor
004	1853	Indoor
005	2500	Outdoor
006	2445	Outdoor
007	2461	Outdoor
008	2300	Outdoor
009	2354	Outdoor
010	2750	Outdoor
TOTAL	**24412**	Indoor 4 / Outdoor 6

B. Ghost-FWL Dataset

This section describes the details of the Ghost-FWL dataset. Fig. 1 shows an overview of all scenes in the Ghost-FWL dataset. Table 1 shows the number of frames for each scene.

C. Fundamentals of LiDAR and Full-Waveform Signals

LiDAR measures the three-dimensional structure of the environment by emitting laser pulses and computing the time-of-flight (ToF) of their returns. In standard commercial LiDAR systems, the raw received signal is internally processed to detect only the dominant return peaks, and the sensor outputs a set of 3D points, commonly referred to as a point cloud. This point-cloud representation is compact, easy to handle in downstream perception pipelines, and greatly reduces data bandwidth. However, this conversion discards most of the physical information originally present in the raw waveform, including material-dependent reflectance behavior, multi-path components, and the detailed temporal shape of each peak.

In contrast, a Full-Waveform (FW) LiDAR records the complete temporal intensity profile of the returned signal for each beam direction. Because the full waveform preserves the full time evolution of the reflected light, it retains rich physical cues such as variations induced by material properties, surface geometry, incidence angle, and multi-

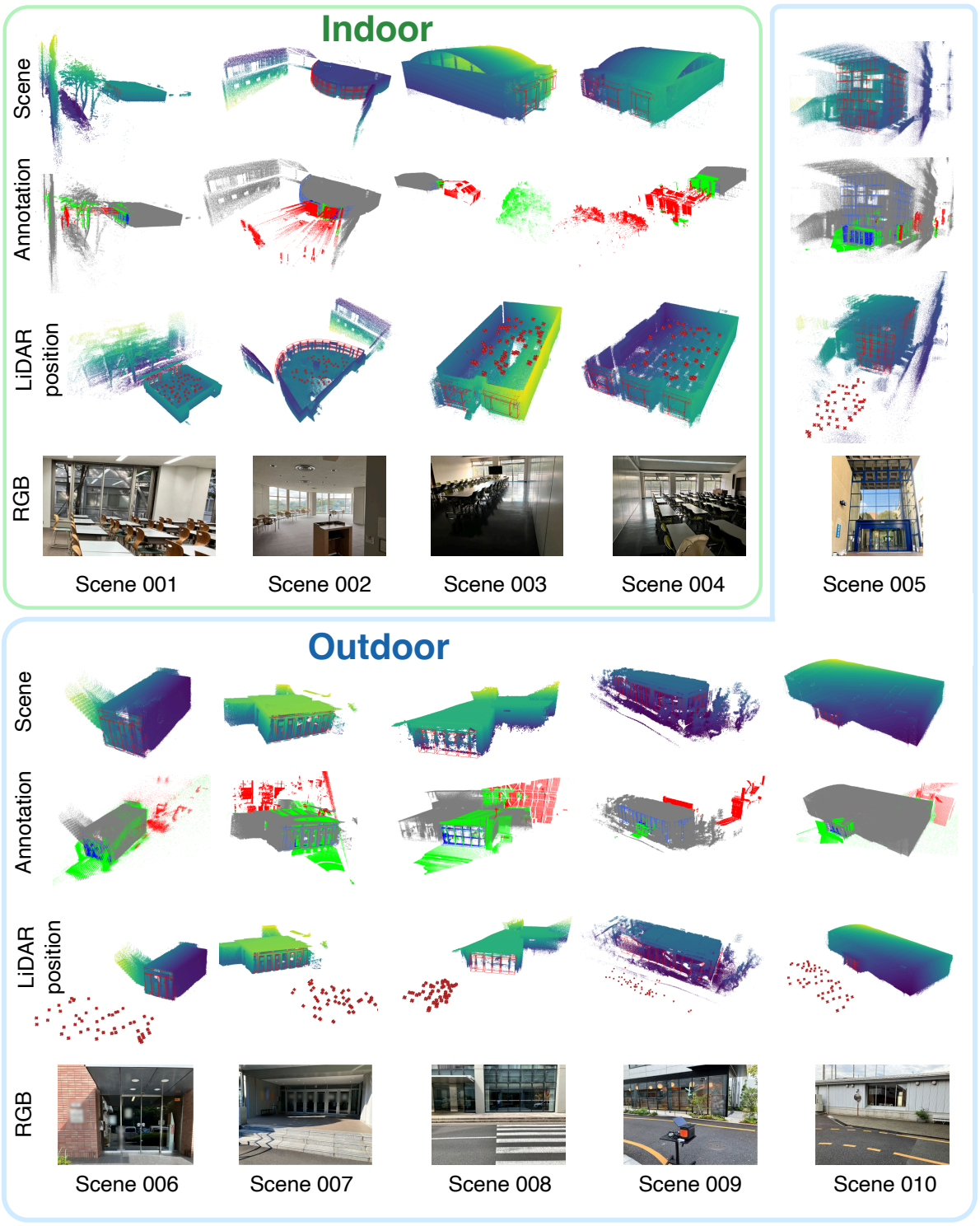

Figure 1. **All scenes in Ghost-FWL.** Our dataset includes both indoor and outdoor scenes. Based on the dense 3D maps as shown in Scene, we annotated FWL data with semantic labels: *Ghost* (red), *Object* (green), *Glass* (blue), *Noise*. Gray regions are excluded from annotation. Red crosses indicate the LiDAR positions during data acquisition. The RGB images show the scenery of the capture locations.

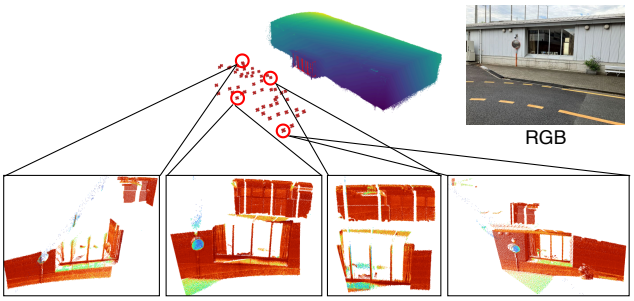

Figure 2. Example of LiDAR position and perspective.

path reflections through glass or other reflective structures including colored glass (Scene 008 glass in Fig. 1) and film-covered glass (Scene 002 glass in Fig. 1) surfaces. These temporal characteristics, suppressed or entirely lost in conventional point-cloud outputs, are crucial for analyzing and identifying ghost reflections, making FW LiDAR fundamentally more informative for ghost detection and removal.

C.1. Annotation Strategy

Annotating FWL data that contain ghost reflections are fundamentally challenging. Ghost returns are virtual reflections that may not correspond to any physical surface in the scene; therefore, their spatial location, intensity, and temporal patterns vary depending on the geometry and reflectance of glass or other reflective materials. As a result, no direct ground-truth reference exists for identifying where ghost peaks should appear. In addition, raw FWL data include a large amount of noise originating from natural illumination such as sunlight, as well as weak background reflections. These noise peaks occur randomly along the temporal axis and cannot be separated from ghost peaks by simple filtering or thresholding, making conventional annotation approaches unreliable.

To address these difficulties, we construct an annotation framework that jointly leverages a high-precision 3D map (GT) and accumulated FWL data. For each scene, we prepare (i) a high-accuracy 3D map, and (ii) multiple FWL frames obtained from the same environment. The FWL frames are first accumulated to obtain a high-SNR waveform, which suppresses stochastic noise while enhancing stable reflection components, including both object returns and consistent multi-path signals associated with ghost reflections. The accumulated waveform is then converted into a point cloud and compared with the GT map to determine whether each FWL peak corresponds to a real object surface, a glass region, a ghost reflection, or noise. Because ghost reflections typically appear at locations that deviate from the real-world geometry, the spatial discrepancy between the accumulated FWL-derived point cloud and the GT map provides a reliable and discriminative cue for identifying ghost peaks.

Labels are first assigned in the point-cloud domain and then transferred back to the corresponding peaks in the accumulated FWL data, yielding peak-level annotations suitable for supervised learning.

C.2. Annotation Pipeline

1. **GT Map Construction:** A high-precision 3D map is constructed using a commercial LiDAR sensor (Livox Mid-360 [9]) together with the SLAM algorithm fastlio2 [17]. After generating the map, ghost points and noise points are removed, and two spatial regions are manually defined using labelCloud [13]: the *glass region* $\mathcal{G}$, which indicates where glass surfaces are likely to exist, and the *reflection region* $\mathcal{R}$, which denotes a larger zone extending radially behind the reflective surfaces where multi-path reflections may occur. Because the alignment between the FWL-derived points and the GT map is performed manually, small alignment errors are unavoidable. To prevent these errors from negatively affecting the labeling process, $\mathcal{R}$ is intentionally defined to be larger than $\mathcal{G}$.
2. **Alignment with Accumulated FWL Data:** For each scene, 37–55 viewpoints are selected, and approximately 50 FWL frames are captured per viewpoint. Fig. 2 shows example of LiDAR positions and viewpoints. These frames are accumulated to obtain a high-SNR FWL signal. Peak detection is then applied to the accumulated waveform, and the detected peaks are converted into a point cloud. The accumulated FWL point cloud is manually aligned with the GT map to correct for discrepancies between their coordinate systems.
3. **Label Assignment:** Let $\mathcal{M}$ denote the set of GT map points. For each accumulated FWL point $\mathbf{x}$, we compute its nearest-neighbor distance to the GT map as

$$d(\mathbf{x}) = \min_{\mathbf{y} \in \mathcal{M}} \|\mathbf{x} - \mathbf{y}\| \tag{1}$$

Using this distance and the region definitions, each point is classified as follows:

- **Glass**:
$$\mathbf{x} \in \mathcal{G} \tag{2}$$
- **Object**:
$$d(\mathbf{x}) < \tau \quad \text{and} \quad \mathbf{x} \notin \mathcal{G} \tag{3}$$
- **Ghost**:
$$d(\mathbf{x}) > \tau \quad \text{and} \quad \mathbf{x} \in \mathcal{R} \tag{4}$$
- **Noise:** All remaining points that do not satisfy any of the above conditions.

Here, τ denotes the Euclidean distance threshold. Once these labels are assigned in the point-cloud domain, they are transferred back to the corresponding peaks in the

Table 2. Annotation parameters for each scene in the Ghost-FWL dataset. The threshold τ is the nearest-neighbor distance used to distinguish Object and Ghost points, and "# Glass Areas" denotes the number of manually annotated glass regions $\mathcal{G}$.

Scene	Threshold τ	# Glass Areas
001	0.5	8
002	0.5	48
003	0.5	10
004	0.5	9
005	0.5	15
006	0.5	5
007	0.5	17
008	0.5	12
009	0.5	15
010	0.5	4

accumulated FWL data, completing the peak-level annotation procedure. The parameters used to construct the Ghost-FWL dataset are summarized in Tab. 2.

C.3. Improving Annotation Quality via Waveform-Level Accumulation

The raw full-waveform signal recorded by a LiDAR sensor inevitably contains a considerable amount of noise. Such noise appears as randomly distributed peaks along the temporal axis and is primarily caused by external illumination, including sunlight, as well as weak background reflections from the ground and surrounding surfaces. To suppress these stochastic components, modern LiDAR systems employ an accumulation mechanism in which multiple laser pulses are emitted in the same direction and their corresponding waveforms are aggregated. Since true reflections from physical objects consistently appear at the same temporal position across pulses, while noise peaks vary randomly, accumulation enhances stable reflection components and averages out random noise.

We leverage the same accumulation strategy to improve the reliability of our annotation pipeline. As shown in Fig. 3, for each viewpoint, 50 FWL frames captured from the same location are aggregated to generate a high-SNR waveform. This process amplifies not only direct reflections from real objects but also consistent multi-path components responsible for ghost reflections, while effectively suppressing sporadic noise induced by sunlight or ground scattering. The accumulated waveform is then converted into a point cloud, ensuring that only physically meaningful reflection peaks remain. This high-quality representation provides a robust basis for distinguishing true objects from ghost reflections, enabling more reliable detection and highly accurate peak-level annotation of ghost returns.

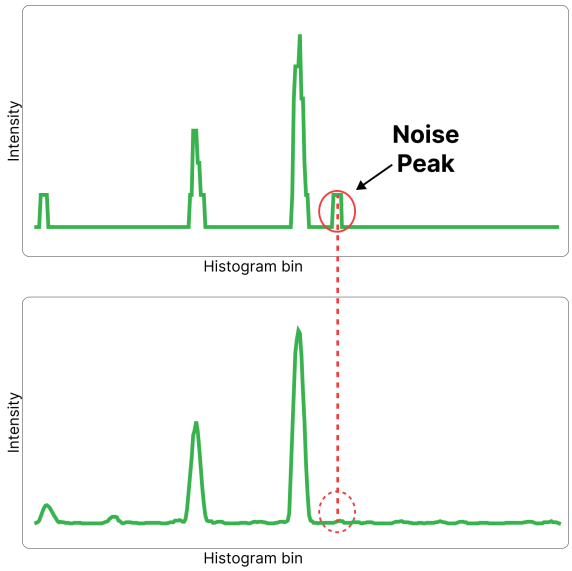

Figure 3. Example of 1 frame FWL data(top) and 50× accumulation FWL data (bottom).

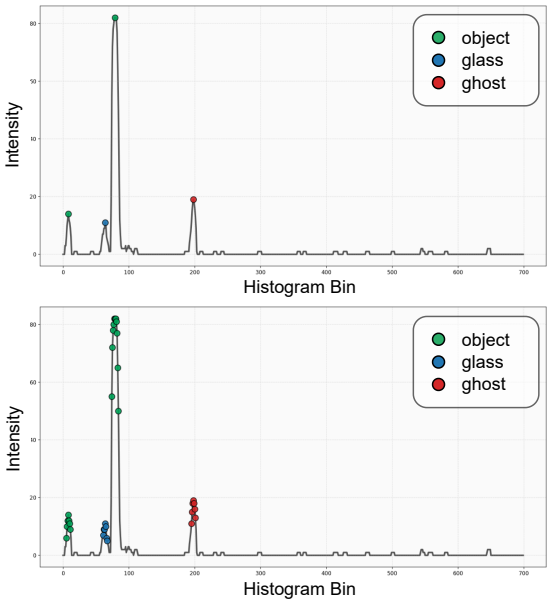

Figure 4. Example of raw annotation (top) and processed annotation for training (bottom). The circles indicate the annotation points for each class: *Object* (green), *Glass* (blue) and *Ghost* (red).

C.4. Annotation Processing for Training

During training, both the input FWL data and the annotations must be down-sampled due to GPU memory limitations. However, peak annotations exist only at the single point at the exact peak of the histogram, so they can be lost during down-sampling along the T-axis when the sampling interval is large. Therefore, during training, we ex-

Table 3. FWL-MAE Architecture.

Name	Layer setting	Output dimension
Patch Embedding	3D Conv Flatten + Transpose	$B \times N_{\text{patch}} \times D_{\text{encoder}}$
Positional Encoding	Sinusoidal Position Encoding (PE)	$B \times N_{\text{patch}} \times D_{\text{encoder}}$
Masking	Select Unmasked Tokens + PE	$B \times N_{\text{unmasked}} \times D_{\text{encoder}}$
Encoder	Transformer Blocks $\times$ 6 LayerNorm	(pretraining) $B \times N_{\text{unmasked}} \times D_{\text{encoder}}$ (fine-tuning) $B \times N_{\text{patch}} \times D_{\text{encoder}}$
Concatenation Tokens	Linear Projection [$D_{\text{encoder}} \rightarrow D_{\text{decoder}}$] Unmasked Features + PE Masked Tokens + PE Concat (Unmasked N_{unmasked}, Masked N_{masked})	$B \times N_{\text{patch}} \times D_{\text{decoder}}$
Peak Position Head	Linear [$D_{\text{decoder}} \rightarrow K$] Sigmoid $\times$ (T - 1)	$B \times N_{\text{patch}} \times K$
Peak Width Head	Linear [$D_{\text{decoder}} \rightarrow K$] Softplus	$B \times N_{\text{patch}} \times K$
Peak Height Head	Linear [$D_{\text{decoder}} \rightarrow K$] Softplus	$B \times N_{\text{patch}} \times K$
Decoder	Transformer Blocks $\times$ 6 Linear [$D_{\text{decoder}} \rightarrow (H_{\text{patch}} \times W_{\text{patch}} \times T_{\text{patch}})$]	$B \times N_{\text{mask}} \times (H_{\text{patch}} \times W_{\text{patch}} \times T_{\text{patch}})$
Classification Head	Linear [$D_{\text{encoder}} \rightarrow \frac{D_{\text{encoder}}}{2}$] + ReLU + Dropout Linear [$\frac{D_{\text{encoder}}}{2} \rightarrow (H_{\text{patch}} \times W_{\text{patch}} \times T_{\text{patch}} \times C)$] Reshape Patches $\rightarrow$ FWL data	$B \times C \times T \times H \times W$

pand each annotation around the original peak position by its full width at half maximum (FWHM). This expansion ensures that, the ground-truth peak annotation still covers the peak of the same waveform even after down-sampling. The comparison between raw and processed annotations for training is shown in Fig. 4.

D. Implementation Details of the FWL-based Ghost Removal Framework

This section presents the implementation details of the FWL-based Ghost Removal Framework. Table 3 shows a detailed architecture of FWL-MAE. All training and processing were performed on a computer equipped with an Intel Xeon w5-3535X CPU and a single RTX 6000 Ada Generation GPU, running the Ubuntu 22.04 operating system.

D.1. Full Waveform LiDAR Masked Autoencoder

This subsection describes the implementation details of the self-supervised pretraining method, the Full-Waveform LiDAR Masked Autoencoder (FWL-MAE), which is designed to obtain latent representations of histograms from FWL data.

Model. The implementation of the Transformer-based baseline model is inspired by VideoMAE [16]. We first apply 3D convolutions to the FWL data to generate patch embeddings $(B, N_{\text{patch}}, D_{\text{encoder}})$, where B is the batch size, N_{patch} is the number of patches, and D_{encoder} is the encoder embedding dimension. For masking, we randomly sample spatial patches in the (x, y) region and mask all temporal bins along the T axis within each selected patch.

Only the unmasked patches are fed into a Transformer encoder composed of six Transformer blocks with six attention heads in each block. The Transformer Encoder outputs feature vectors for the unmasked regions with shape $(B, N_{\text{unmasked}}, D_{\text{encoder}})$, where B is the batch size, N_{unmasked} is the number of unmasked patches, and D_{encoder} is the encoder embedding dimension.

The Transformer decoder consists of six Transformer blocks with six attention heads in each block. It takes as input the feature vectors of the unmasked regions and the mask tokens $(B, N_{\text{patch}}, D_{\text{decoder}})$. The decoder then reconstructs the FWL data corresponding to the masked patches.

In the Peak Head, the peak *position* (p), *amplitude* (a), and *width* (w) of the FWL data are estimated simultaneously. The ground-truth peak position, amplitude, and full width at half maximum (FWHM) are extracted directly from the input FWL data. Under the assumption that peaks farther from the LiDAR sensor are less reliable and less informative, the Peak Head estimates only the K peaks, where K is set to 4 in this paper. If fewer than K peaks exist in input FWL data, the missing ground-truth peak position, amplitude, and width are padded with zeros. Since peak prediction is performed at the patch level, the ground-truth peak parameters for each patch are obtained by averaging the peak values within the corresponding spatial patch region.

In our setting, the masking ratio of FWL data was set to 70%. The input FWL data size is $(H, W, T) = (128, 128, 256)$ and the patch size is $(H_{\text{patch}}, W_{\text{patch}}, T_{\text{patch}}) = (16, 16, 256)$. The embedding dimension is $D_{\text{encoder}} = 768, D_{\text{decoder}} = 384$.

Preprocessing. The raw FWL data were reshaped to $(H, W, T) = (128, 128, 256)$ before being fed into the model through the following preprocessing. First, the top and bottom 90 bins corresponding to reflections from the ceiling and floor and the front 25 bins containing noise from internal sensor reflections were removed. The remaining data were then uniformly down-sampled along the T axis and randomly cropped into $H \times W$ size. The preprocessed FWL data is then fed into the model.

Training. We performed pretraining with FWL-MAE using 8,933 unlabeled frames captured in a mobile environment. We used AdamW [10] as optimizer. The AdamW hyperparameters were set to $\beta_1 = 0.9, \beta_2 = 0.999$, $\epsilon = 1 \times 10^{-8}$, weight decay of $\lambda = 1 \times 10^{-2}$, and a learning rate of $\alpha = 1 \times 10^{-3}$. The batch size was set to 32, and training was performed for 100 epochs. The weighting coefficients for the loss function $\mathcal{L}_{\text{FWL-MAE}}$ are $\lambda_p = 1.0$, $\lambda_a = 1.0$, and $\lambda_w = 0.5$.

D.2. Ghost Detection and Removal

This subsection describes the method for peak-level classification for ghost detection and removal. To detect and remove ghosts, our method takes the FWL data $\boldsymbol{V} \in \mathbb{R}^{H \times W \times T}$ as input and estimates the class probabilities $\boldsymbol{P} \in \mathbb{R}^{H \times W \times T \times C}$ for the categories *Glass*, *Ghost*, *Object*, and *Noise*.

Model. To extract informative features from the FWL data, we use the encoder pretrained with FWL-MAE and keep its weights frozen to obtain latent representations $f_\theta(\boldsymbol{V})$. A lightweight classification head composed of two linear layers is then applied to predict the class probabilities for all FWL data coordinates.

Preprocessing. The input size to the model and the preprocessing procedure are the same as those used during the pretraining with FWL-MAE.

Training. We used a data split of 13,853 for training, 2,994 for validation, and 1,427 for testing. The training and validation sets contain data captured in Scene 001, 003, 004, 005, 006, 008 and 010. The testing set contains data captured in Scene 002, 007 and 009. Although the training and validation sets were captured in the same scenes, no overlapping frames were used. The test set consists solely of unseen scenes that are not included in either the training or validation data. In this study, we used the above split, but it can be modified as needed.

We used AdamW [10] as optimizer. The AdamW hyperparameters were set to $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1 \times 10^{-8}$, weight decay of $\lambda = 1 \times 10^{-2}$, and a learning rate of $\alpha = 1 \times 10^{-3}$. The batch size was set to 32, and training was performed for 100 epochs.

Loss function. We adopt the focal loss [6], which mitigates the impact of class imbalance in multi-class classification. Our task poses a challenging classification problem involving highly imbalanced data, where minority classes such as *Ghost* coexist with the majority *Noise* class. The loss function is defined as follows:

$$\mathcal{L}_{\text{Focal}} = -\sum_{c=1}^{C} \alpha_c (1 - \boldsymbol{p}_c)^\gamma \log \boldsymbol{p}_c, \tag{5}$$

where C is the number of class, $\boldsymbol{p}_c$ denotes the predicted probability for the ground-truth class c. α_c is a weighting factor for each class (Glass, Ghost, Object, Noise), used to compensate for class imbalance, and γ is the focusing parameter that controls the trade-off between easy and hard samples. The number of classes are set as $C = 4$ (Glass, Ghost, Object, and Noise). The parameters were set to $\alpha_{\text{glass}} = 0.25$, $\alpha_{\text{ghost}} = 0.7$, $\alpha_{\text{object}} = 0.05$, $\alpha_{\text{noise}} = 0.0001$, and $\gamma = 2.0$.

Inference. We describe the inference procedures for downstream tasks such as SLAM and object detection, as well as for qualitative evaluation of classification.

First, as in the training phase, the top and bottom 90 bins corresponding to reflections from the ceiling and floor and the front 25 bins containing noise from internal sensor reflections were removed.

During inference, we use FWL data with the same size $(H, W, T) = (128, 128, 256)$ as those used during training. The raw data are down-sampled along the T-axis. Next, unlike in training where random cropping is applied, the FWL data are cropped sequentially starting from the patch coordinate $(x, y) = (0, 0)$ The FWL data are then processed sequentially using a sliding window applied to cropped regions that do not overlap. If the window extends beyond the valid range, the outside area is padded with zero. The inference results obtained sequentially through the sliding window process are then merged, and finally up-sampled to match the original input shape. During up-sampling, zeros

are padded between values to prevent the number of predicted classes from artificially increasing. During inference, the predicted class was determined as the one with the highest probability if it exceeded 0.5; otherwise, it was assigned to *Undefined*.

E. Experiments and Results

This section summarizes additional experiments that could not be included in the main paper.

E.1. Ghost Denoising Evaluation

Detail of Metrics. We evaluate ghost detection at two levels: peak-level and point-level. For peak-level evaluation, we follow Scheuble et al. [14] and report recall, measuring the proportion of correctly detected ghost peaks among all ground-truth ghost peaks. In Recall, only the peaks within the FWL data are detected, and evaluation is performed on their positions.

For point-level evaluation, we introduce the *Ghost Removal Rate*, which measures the proportion of ghost points successfully removed after converting predicted peaks to 3D point clouds. This metric is inspired by the snow removal rate in Charron et al. [1] and assesses practical denoising effectiveness in downstream tasks. For each GT point, if no point in the denoised points was found within a radius r, the GT point was counted as removed. The Removal Rate is then defined as the ratio of removed GT points to the total number of GT points:

$$\text{Ghost Removal Rate} = \frac{N_{\text{removed}}}{N_{\text{GT}}}, \tag{6}$$

where N_{removed} denotes the number of removed ghost points, and N_{GT} is the total number of GT points. The radius was set to $r = 0.001$ in meters.

E.1.1. Ghost Classification Evaluation

Metrics. We follow Scheuble et al. [14] and report recall, measuring the proportion of correctly detected ghost peaks among all ground-truth ghost peaks.

Comparative Methods. To investigate the effectiveness of the proposed FWL-MAE pretrained encoder, we compared five models: (1) the proposed model incorporating FWL-MAE (**Ours**), (2) the model without FWL-MAE (**Ours w/o FWL-MAE**), (3) the model pretrained using a general MAE designed for transient imaging (**MARMOT** [15]), (4) the model for transient imaging (**Lindell et al.** [7]) and (5) the most commonly used 3D convolution-based model (**3D U-Net [2]**.

For (1), the proposed method applied self-supervised pretraining using FWL-MAE and then finetuned the model on the Ghost-FWL dataset. For (2), the model was trained from scratch on the Ghost-FWL dataset without FWL-MAE, using random weight initialization. For (3), the same architecture was pretrained with MARMOT [15] , a general MAE design for transient imaging, and then finetuned on the Ghost-FWL dataset. For (4), Lindell et al. [7] is a 3D convolution-based depth estimation model takes transient imaging as input. Although it can also utilize intensity image, we use only the transient input in our experiments. For (5), 3D U-Net [2] is employed as a commonly used 3D convolution-based method capable of handling 3D inputs.

Table 4. Comparison of ghost removal performance with other methods.

Method	Recall (↑)
3D U-Net [2]	0.391
Lindell et al. [7]	0.641
MARMOT [15]	0.746
Ours w/o FWL-MAE	0.704
Ours	**0.751**

Results. As shown in Table 4, the proposed Transformer-based model with FWL-MAE achieves superior performance in ghost-detection recall compared with models using alternative pretraining strategies as well as existing 3D convolution-based approaches. These results demonstrate the effectiveness of incorporating FWL-MAE, which enables pretraining that more effectively captures the physical characteristics of FWL data.

Fig. 5 shows the classification results of peak in the FWL data. The prediction results of 3D convolution-based models, 3D U-Net [2] and Lindell et al. [7], contain many regions where the class labels are incorrectly estimated. In contrast, the proposed Transformer-based models, pretrained with MARMOT [15] or FWL-MAE, achieve more accurate classification. However, compared to the ground truth, they still produce a larger number of predictions labeled as ghost. This occurs because noise peak positions are sometimes classified as ghost. Although misclassifying noise as ghost has limited impact on downstream tasks, further improvements in classification accuracy remain desirable.

E.1.2. Sensitivity analysis for classification threshold

We conducted a sensitivity analysis on the ghost detection threshold, resulting in recall values of 0.751, 0.702, and 0.622 at thresholds of 0.5, 0.6, and 0.7, respectively. The threshold of 0.5 was adopted for our main paper as it yielded the best performance.

E.1.3. Ablation Study of FWL-MAE

We conducted quantitative experiments on classification with varying amounts of training data to demonstrate the effectiveness of self-supervised pretraining with FWL-MAE.

Experimental Settings. When the amount of training data described in §D.2 is treated as 100%, we conducted addi-

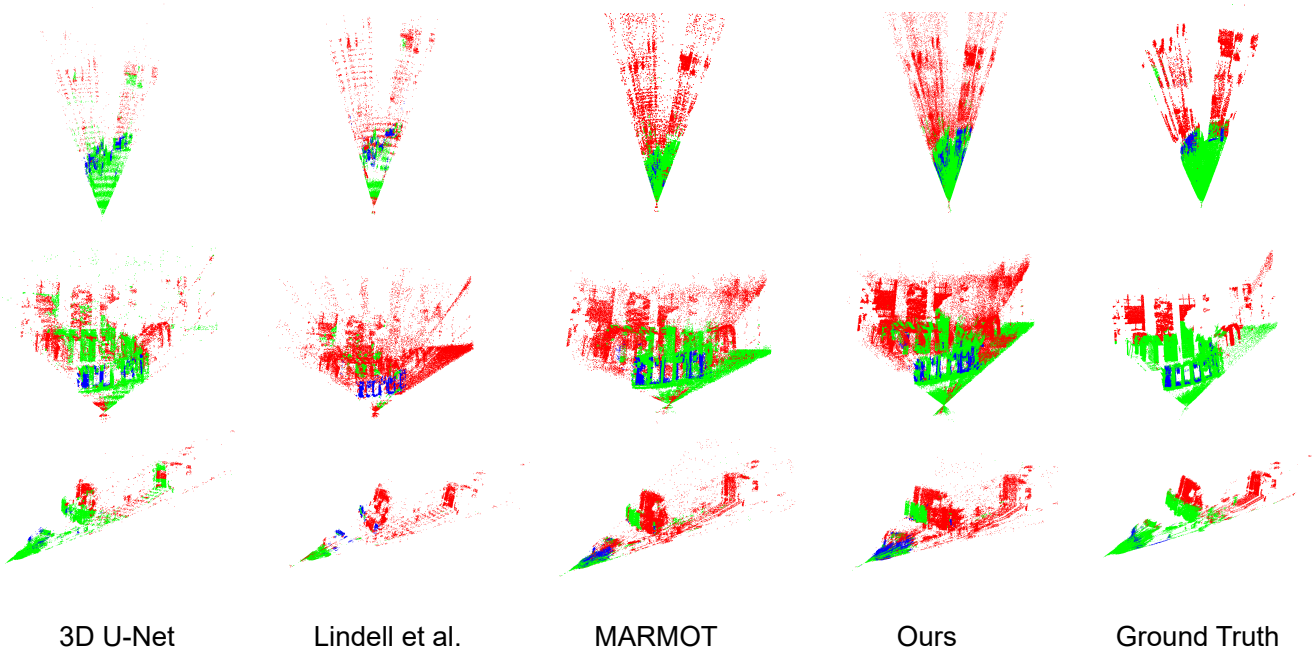

Figure 5. Peak classification results. Red, green and blue indicates *Ghost*, *Object* and *Glass*, respectively.

Table 5. Ablation study of FWL-MAE.

Train Data	Method	Recall (↑)
100%	Ours w/o FWL-MAE	0.704
	Ours	0.751 (+ 0.047)
70%	Ours w/o FWL-MAE	0.602
	Ours	0.692 (+ 0.090)
50%	Ours w/o FWL-MAE	0.403
	Ours	0.603 (+ 0.200)

tional experiments by reducing the training data to about 70% and 50%. For the 70% set contains data captured in Scene 001, 003, 004, 005 and 006, and for the 50% set contains data captured in Scene 001, 004, 005, and 006. The amount of test data was kept unchanged.

Results. Table 5 shows the recall scores of the proposed method with FWL-MAE pretraining compared to the method without FWL-MAE, evaluated under different amounts of training data. As the amount of training data decreases, the recall of the method without FWL-MAE drops substantially, while the proposed method maintains higher performance. Consequently, the performance gap between the two methods widens, highlighting the strong effectiveness of FWL-MAE pretraining, particularly in fewer data settings.

Table 6. Computational cost and inference time. FPS† denotes the inference speed for model input size $(H, W, T) = (128, 128, 256)$. FPS‡ denotes the inference for full-frame size $(H, W, T) = (332, 400, 256)$.

Method	Params [M]	FLOPs [G]	FPS†	FPS‡
3D U-Net [2]	90.3	7610	6.6	0.45
Lindell et al. [7]	1.8	4090	7.0	0.48
Ours	194.1	23.5	**32.1**	**2.35**

E.1.4. Computational cost and inference speed

Table 6 summarizes the computational costs of our method compared to existing 3D CNN-based methods evaluated in Sec. E.1.1. Our Transformer-based approach achieves higher computational efficiency than 3D CNN-based methods. This efficiency stems from our strategy of dividing the input into spatial patches while treating the temporal dimension as a single tube, thereby reducing the total number of operations. In contrast, 3D CNNs involve computationally expensive convolutions across both spatial and temporal dimensions, resulting in more operations and slower inference. Although our method significantly improves throughput for the model input size (FPS†), the inference speed per frame (FPS‡) remains limited. This is primarily because our current framework requires iterative inference for each frame due to memory constraints and the high com-

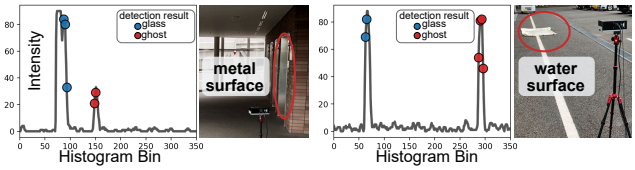

Figure 6. Our model's ghost detection on non-glass surfaces

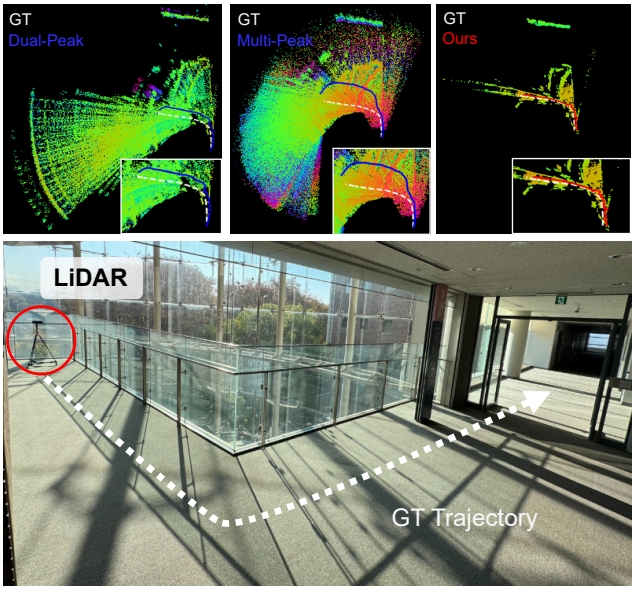

Figure 7. Trajectory and mapping results using Dual-Peak processing (left), Multi-Peak processing (center), and our ghost-removal method (right). The bottom image shows the scenery of the SLAM evaluation, recorded in a corridor enclosed by glass railings and doors. The scene is same to that used in the main paper Fig 5, with the RGB scene image and the visualization results for the Dual-Peak method additionally included.

putational overhead of processing an entire frame at once. Therefore, real-time performance remains challenging.

E.1.5. Efficacy for reflective materials

We conduct additional qualitative experiments on reflective materials other than glass, including water and metal surfaces, as shown in Fig. 6. Although our model is primarily trained on glass-induced ghosts, it also successfully detects ghosts arising from water and metal surfaces, suggesting that it captures material-invariant FWL characteristics of multi-path reflections. We further extend the Ghost-FWL dataset to include these additional surfaces.

E.1.6. Why Prior Ghost Removal Methods Fail on Mobile LiDARs

Conventional ghost removal methods [3, 18] rely heavily on geometric consistency, where ghost points are detected by comparing them with the corresponding real-object points. This assumption holds for stationary LiDAR systems with a full 360° field of view, because both the real object and its ghost reflection are simultaneously observed within the same scan. Consequently, prior approaches can exploit redundant geometric cues to detect and eliminate ghost structures. However, this assumption fundamentally breaks down in mobile LiDAR settings. Automotive LiDAR sensors typically provide a narrow field of view of only 80°–120° [4, 12], and the sensor continuously moves through the environment. As a result, the true object behind a glass surface often falls outside the field of view when the ghost is observed. This makes the simultaneous observation of real and ghost points highly unlikely, rendering geometry-based ghost detection intrinsically infeasible for mobile LiDAR. Moreover, the sparsity of mobile LiDAR point clouds and the dynamic nature of real-world scenes further weaken geometric cues, preventing the accumulation of consistent multi-view observations required by prior methods.

E.2. Evaluation on Downstream Applications

E.2.1. SLAM Experimental Scenes

Fig. 7 presents additional photograph of the SLAM environment together with the corresponding trajectory used in our experiment. The SLAM sequence was captured in an office corridor where strong glass reflections frequently produce ghost points, providing a challenging setting for evaluating the effectiveness of our ghost removal.

E.2.2. SLAM Ablation Studies

We compare against LiDAR signal processing strategies using commercial LiDAR's common factory default settings: Dual-Peak [4, 11] and Multi-Peak [8], which retain the two and three strongest intensity peaks, respectively. To further examine whether point cloud–based denoising can mitigate ghost artifacts, we conducted supplementary ablation experiments, separate from the main evaluations in this paper. In these experiments, each peak-selection strategy was combined with one of three preprocessing variants: no filtering (Raw), a Statistical Outlier Filter, or a Radius-based Outlier Filter. Both filtering modules are provided by Open3D [19]. For the Statistical Outlier Filter, we used a neighbor size of 20 and a standard deviation threshold of 2.0. For the Radius-based Outlier Filter, we set a minimum point count of 50 within a search radius of 0.5 m. We also applied Open3D's voxel downsampling with a voxel size of 1.0 for all methods. All methods employed the same SLAM backend, GLIM [5], ensuring a fair comparison.

We report the SLAM ablation results in Table 7. Although point-cloud–based denoising methods provide a modest improvement in SLAM accuracy, our waveform-based ghost removal delivers a larger and consistent accuracy gain. When compared to the baseline methods combined with the Statistical Outlier Filter, our ghost removal

Table 7. SLAM performance ablation with different point-cloud processing methods.

	Raw	Statistical Outlier Filter	Radius-based Outlier Filter
Dual-Peak ATE [m] (↓)	1.248±0.947	0.639±0.573	0.503±0.593
Multi-Peak ATE [m] (↓)	2.489±1.432	1.232±0.928	0.887±1.079
Ours ATE [m] (↓)	**0.294±0.244**	**0.248±0.171**	**0.328±0.303**
Dual-Peak RTE [m] (↓)	1.221±0.928	0.608±0.529	0.471±0.555
Multi-Peak RTE [m] (↓)	2.513±1.440	1.280±0.901	0.832±1.015
Ours RTE [m] (↓)	**0.288±0.237**	**0.243±0.157**	**0.319±0.291**

Table 8. Details of the object-detection test dataset (PedScene).

Scene	Indoor/ Outdoor	Pedestrians	Frames
PedScene 001	Indoor	2	14
PedScene 002	Outdoor	3	31
PedScene 003	Indoor	3	32
PedScene 004	Indoor	1	25
TOTAL	**Indoor 3 / Outdoor 1**	**1 / 2 / 3**	**102**

Figure 8. Example scenes from the PedScene dataset used for 3D object detection. The dataset includes both indoor and outdoor environments with glass surfaces, building entrances, and glass-walled sidewalks.

further reduces ATE by 54–76% and RTE by 53–78%. Similarly, relative to the Radius-based Outlier Filter, ATE decreases by 35–63% and RTE by 32–62%. These results demonstrate that conventional outlier filtering is insufficient for handling ghost artifacts, and that our waveform-driven ghost removal provides substantially more effective suppression, leading to the highest SLAM accuracy.

E.2.3. Object Detection Test Dataset

We summarize the details of the object-detection test dataset in Table 8. The dataset, referred to as *PedScene*, consists of four scenes recorded in indoor and outdoor environments containing glass surfaces, building entrances, and glass-walled sidewalks. Fig. 8 shows examples of these environments together with RGB images.

References

[1] Nicholas Charron, Stephen Phillips, and Steven L Waslander. De-noising of Lidar Point Clouds Corrupted by Snowfall. In *Conference on Computer and Robot Vision (CRV)*, pages 254–261, 2018. 7

[2] Özgün Çiçek, Ahmed Abdulkadir, Soeren S. Lienkamp, Thomas Brox, and Olaf Ronneberger. 3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 424–432, 2016. 7, 8

[3] Rui Gao, Mengyu Li, Seung-Jun Yang, and Kyungeun Cho. Reflective Noise Filtering of Large-Scale Point Cloud Using Transformer. *Remote Sensing*, 14(3):577, 2022. 9

[4] Hesai. AT128 Automotive-Grade 120° Long-Range Lidar - Hesai. https://www.hesaitech.com/product/at128. 9

[5] Kenji Koide, Masashi Yokozuka, Shuji Oishi, and Atsuhiko Banno. GLIM: 3D range-inertial localization and mapping with GPU-accelerated scan matching factors. *Robotics and Autonomous Systems*, 179:104750, 2024. 9

[6] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal Loss for Dense Object Detection. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007, 2017. 6

[7] David B. Lindell, Matthew O'Toole, and Gordon Wetzstein. Single-photon 3D imaging with deep sensor fusion. *ACM Trans. Graph.*, 37(4), 2018. 7, 8

[8] Livox. Avia. https://www.livoxtech.com/avia, . 9

[9] Livox. Mid-360. https://www.livoxtech.com/mid-360, . 3

[10] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations (ICLR)*, 2017. 6

[11] Ouster Inc. OS1 Mid-Range High-Resolution Imaging Lidar. https://data.ouster.io/downloads/datasheets/datasheet-rev7-v3p1-os1.pdf. 9

[12] RoboSense Technology Co., Ltd. Robosense M3. https://www.robosense.ai/en/rslidar/M3. 9

[13] Christoph Sager, Patrick Zschech, and Niklas Kuhl. labelCloud: A lightweight labeling tool for domain-agnostic 3d object detection in point clouds. *Computer-Aided Design and Applications*, 19(6):1191–1206, 2022. 3

[14] Dominik Scheuble, Hanno Holzhüter, Steven Peters, Mario Bijelic, and Felix Heide. Lidar Waveforms are Worth 40x128x33 Words. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 28913–28924, 2025. 7

[15] Siyuan Shen, Ziheng Wang, Xingyue Peng, Suan Xia, Ruiqian Li, Shiying Li, and Jingyi Yu. MARMOT: Masked Autoencoder for Modeling Transient Imaging. *arXiv preprint arXiv:2506.08470*, 2025. 7

[16] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. VideoMAE: masked autoencoders are data-efficient learners for self-supervised video pre-training. In *International Conference on Neural Information Processing Systems (NeurIPS)*, 2022. 5

[17] Wei Xu, Yixi Cai, Dongjiao He, Jiarong Lin, and Fu Zhang. Fast-lio2: Fast direct lidar-inertial odometry. *IEEE Transactions on Robotics*, 38(4):2053–2073, 2022. 3

[18] Jae-Seong Yun and Jae-Young Sim. Virtual Point Removal for Large-Scale 3D Point Clouds with Multiple Glass Planes . *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 43(2):729–744, 2019. 9

[19] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv preprint arXiv:1801.09847*, 2018. 9