Supplementary Material
Transient LASSO: Transient Large-Scale Scene Reconstruction

DOMINIK SCHEUBLE*, Mercedes-Benz AG, Germany and Technical University of Darmstadt, Germany
ANDREA RAMAZZINA*, Mercedes-Benz AG, Germany and Technical University of Munich, Germany
HANNO HOLZHÜTER, Microvision Inc, USA and Leibniz University Hannover, Germany
STEFANO GASPERINI, Technical University of Munich, Germany and Munich Center for Machine Learning, Germany
STEVEN PETERS, Technical University of Darmstadt, Germany
FEDERICO TOMBARI, Technical University of Munich, Germany and Google, Switzerland
MARIO BIJELIC, Princeton University, USA and Torc Robotics, USA
FELIX HEIDE, Princeton University, USA and Torc Robotics, USA

CCS Concepts: • **Computing methodologies** → **Reconstruction**.

ACM Reference Format:
Dominik Scheuble*, Andrea Ramazzina*, Hanno Holzhüter, Stefano Gasperini, Steven Peters, Federico Tombari, Mario Bijelic, and Felix Heide. 2025. Supplementary Material Transient LASSO: Transient Large-Scale Scene Reconstruction. In *SIGGRAPH Asia 2025 Conference Papers (SA Conference Papers '25), December 15–18, 2025, Hong Kong, Hong Kong*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3757377.3763911

This supplementary document provides further description to support the findings from the main manuscript. Specifically, we give detailed descriptions of the model architecture and all method hyperparameters. We provide additional evaluation details, and we offer a detailed description of the sensor employed and its imaging model. This document is organized as follows:

* These authors contributed equally to this work.

Authors' addresses: Dominik Scheuble*, Mercedes-Benz AG, Boeblingen, Germany and Technical University of Darmstadt, Darmstadt, Germany; Andrea Ramazzina*, Mercedes-Benz AG, Boeblingen, Germany and Technical University of Munich, Munich, Germany; Hanno Holzhüter, Microvision Inc, Seattle, USA and Leibniz University Hannover, Hannover, Germany; Stefano Gasperini, Technical University of Munich, Munich, Germany and Munich Center for Machine Learning, Munich, Germany; Steven Peters, Technical University of Darmstadt, Darmstadt, Germany; Federico Tombari, Technical University of Munich, Munich, Germany and Google, Zurich, Switzerland; Mario Bijelic, Princeton University, Princeton, NJ, USA and Torc Robotics, Ann Arbor, USA; Felix Heide, Princeton University, Princeton, NJ, USA and Torc Robotics, Ann Arbor, USA.

SA Conference Papers '25, December 15–18, 2025, Hong Kong, Hong Kong

ACM ISBN 979-8-4007-2137-3/2025/12
https://doi.org/10.1145/3757377.3763911

Contents

1 ADDITIONAL DETAILS ON METHOD AND TRAINING

In this section, we present more details about our algorithm, including model architecture and training.

1.1 Training Details

In this section, we further describe the training process. We train the proposal network $\mathbf{f}_{\mathrm{p}}$, the geometry and the appearance networks $\mathbf{f}_{\mathrm{g}}$, $\mathbf{f}_{\mathrm{a}}$, together with the spatial embedding f_e for 200,000 steps, jointly optimizing the the hardware-specific parameters. These parameters include the per-pixel offset ϑ_{p}, the pulse shape $\hat{g}$, as well as the high-flux response $\hat{h}$.

The total training loss is defined as:

$$\mathcal{L} = v_1\mathcal{L}_b + v_2\mathcal{L}_g + v_3\mathcal{L}_n + v_4\mathcal{L}_a + v_5\mathcal{L}_{\langle}, \tag{1}$$

where $v_1 = 1.0$, $v_2 = 0.5$, $v_3 = 0.1$, $v_4 = 0.1$, $v_5 = 0.1$.

The first 1,000 steps are used as a warm-up phase, during which only the density field $\mathbf{f}_{\mathrm{g}_{\mathrm{o}}}$ is optimized using the loss $\mathcal{L}_g$. This bootstrapping strategy helps prevent early entanglement between geometry and appearance, thereby avoiding poor local minima.

For the reflectance regularization loss $\mathcal{L}_a$, we sample $\boldsymbol{\epsilon}_x$ uniformly from the interval $(-10^{-3}, 10^{-3})^3$ and gradually reducing it to zero over 10,000 iterations.

The high-flux mask s is obtained by evaluating each pixel for saturation, as waveforms from retroreflectors exhibit a characteristic initial peak that saturates the LiDAR's dynamic range, as described by Fig 3 and Fig. 5.

1.2 Network Details

Next, we outline the network architecture underlying our model. Following the approach of [Müller et al. 2022], we map each 3D spatial location $\mathbf{x}$ to a hash-encoded representation, which serves as input to the subsequent multi-headed field networks f_g and f_a, modeling respectively the geometry and the appearance of the scene. Each network is implemented as a shallow, fused MLP, whose detailed architecture is shown in Tables 1-3. The hash encoding comprises 16 levels, each with 2 features, and supports a maximum resolution of 2048.

Field f_e				
Layer #	**Layer Type**	**Activation**	**Input Shape**	**Output Shape**
0	Linear	ReLU	$hash_{dim}$	128
1	Linear	ReLU	128	128
2	Linear	ReLU	128	128
3	Linear	ReLU	128	χ_{dim}

Table 1. Architecture of the MLP f_e.

Here, $\chi_{dim} = 20$ denotes the dimensionality of the spatial embedding $\boldsymbol{\chi}$. The volumetric density and normal are then inferred using the spatial embedding $\boldsymbol{\chi}$ and the encoded spatial position $\gamma(\mathbf{x})$ with dimension $\gamma_{dim} = 10$ The architecture of f_g is detailed in Table 2. We employ a truncated exponential as activation for f_{g_o} for the density. We add a trainable bias to the output of f_{g_n}, which initializes all normals as upright, which helps with stability during training. We employ no activation for f_{g_n}, but afterwards normalize them to unit vectors.

Field f_g				
Layer #	**Layer Type**	**Activation**	**Input Shape**	**Output Shape**
0	Linear	ReLU	χ_{dim}	128
1	Linear	ReLU	128	128
2a	Linear	ReLU	128	1
2b	Linear	None	128+γ_{dim}	3

Table 2. Architecture of the MLP f_g. For f_{g_n} the output dimension is 3, for f_{g_o} the output dimension is 1

Similarly, the field f_a receives as input the spatial embedding χ and the encoded spatial position of dimension γ_{dim}. The architecture is detailed in Table 3.

Field f_a				
Layer #	**Layer Type**	**Activation**	**Input Shape**	**Output Shape**
0	Linear	ReLU	$\chi_{dim} + \gamma_{dim}$	128
1	Linear	ReLU	128	128
2	Linear	ReLU	128	128
3a	Linear	Sigmoid	128	1
3b	Linear	Sigmoid	128	1
3c	Linear	Sigmoid	128	1

Table 3. Architecture of the multiheaded MLP f_a.

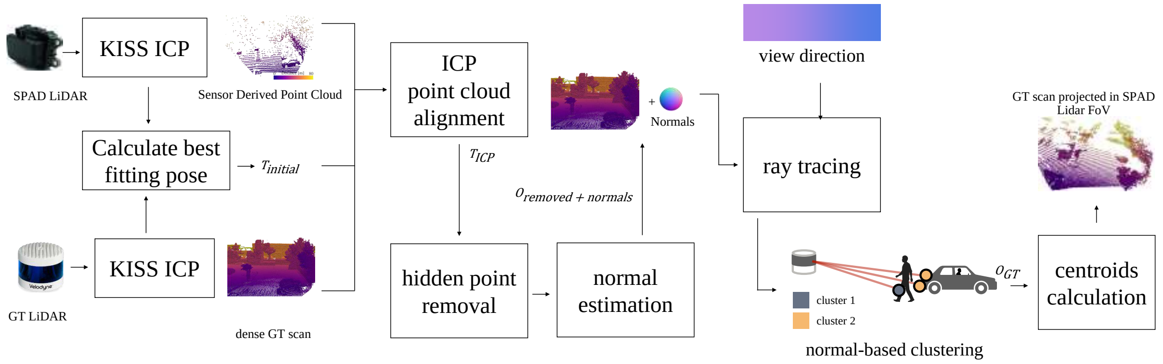


Fig. 1. **Ground Truth Point Cloud Generation.** We create the ground truth point cloud by first aligning the FWL sensor point cloud within the dense ground truth (GT) scan using an Iterative Closest Point (ICP) algorithm. After removing hidden points due to scene occlusions, we do ray tracing to identify all points of the GT scan that lie within the diverging laser beam. These points are then clustered to obtain the final ground truth point cloud.

2 EVALUATION DETAILS

2.1 Ground-truth Point Cloud Generation

To evaluate the scene reconstruction, we generate a dense ground-truth point cloud from the reference Velodyne-VLS 128 LiDAR. More in detail, we accumulate LiDAR scans captured by a Velodyne-VLS 128 across different timestamps into a single point cloud that we align using Kiss-ICP [Vizzo et al. 2023]. The resulting dense point cloud is our ground truth, which we project into the field of view of the SPAD LiDAR sensor. An overview of the ground truth generation is shown in Figure 1. In the following, we describe the steps we followed. The ground-truth pipeline closely follows the one introduced by [Scheuble et al. 2025].

Calculating the Pose. We employ KISS-ICP also on the sensor-derived point clouds from the SPAD lidar. Due to the noisier point clouds of the SPAD LiDAR, we refine the poses extracted this way. Using a distance criterion, we find the closest match (the best fitting pose) between the SPAD LiDAR and GT LiDAR poses. We repeat this process for every pose of the SPAD LiDAR and extract a refined pose $T_{\text{initial}} \in \mathbb{R}^{4\times4}$, which is subsequently used as a starting point to project the dense ground truth in the SPAD LiDAR FoV.

Aligning the Point Clouds We further refine T_{initial} by aligning the SPAD LiDAR point cloud using the standard Iterative Closest Point (ICP) algorithm from Open3D [Zhou et al. 2018] with the dense GT, obtaining the transformation $\mathbf{T}_{\text{ICP}} \in \mathbb{R}^{4\times4}$.

Removing Hidden Points Each ground truth point cloud includes points that are not visible from the current pose of the SPAD sensor at a given position due to occlusion by the scene elements. We removed these ”hidden points” from the ground truth. We achieve this by projecting the full aggregated and aligned ground truth point cloud into the sensor's field of view using the $\mathbf{T}_{\text{ICP}}$ transform, then we leverage the algorithm of Katz *et al.* [Katz et al. 2007]. This gives us a dense ground truth scan $\mathbf{O}_{\text{removed}}$ with shadow areas behind objects that are non-transparent, as shown in Figure 1.

Estimating Normals We use surface normals for later processing. We obtain such normals using the technique from Open3D [Zhou et al. 2018]. This enriches each point with normal information, leading to $\mathbf{O}_{\text{removed+normals}}$.

Ray Tracing We do ray tracing by taking into account the diverging LiDAR beam and considering the view direction to the ground truth point cloud $\mathbf{O}_{\text{removed+normals}}$. So, for each viewing angle $\mathbf{v}_{m,n} \in \mathbb{R}^3$ for all points $\mathbf{o} \in \mathbf{O}_{\text{removed+normals}}$ within the cone of the diverging LiDAR beam. In particular, we add points $\mathbf{o}$ to the set $O_{m,n}$

Fig. 2. **Full waveform LiDAR.** Display of the LiDAR sensor used for our full waveform data recording.

so that $O_{m,n} = \left\{ \mathbf{o} \text{ if } \mathbf{v}_{m,n}^T \frac{\mathbf{o}}{||\mathbf{o}||} > \cos \gamma^{\text{div}} \right\}$ with γ^{div} being the emitted LiDAR beam's divergence. We get a set of points O visible from all pixels by computing $O_{m,n}$ for all pixels (m, n).

Normal-based Clustering Due to the beam divergence, the point cloud generated by O likely often contains clusters of points at distinct distances. This phenomenon is illustrated in Figure 1, where the diverging beam simultaneously strikes both a pedestrian and a car, resulting in two separate clusters—one composed of a single point and the other of two points. Since the number of clusters is generally unknown in advance, we employ the widely-known DBSCAN [Ester et al. 1996] clustering algorithm, which does not require specifying the number of clusters beforehand. However, DBSCAN relies on a few key parameters that need to be set: the maximum distance threshold d^{DBSCAN} to determine whether points belong to the same cluster and the minimum number of points to form a new cluster. We observe that a fixed threshold is inadequate for d^{DBSCAN}, as points on upright objects (e.g., tree trunks) are typically closer together than those on flat surfaces such as the ground. This variation arises because the effect of beam divergence increases as the angle between the viewing direction $\mathbf{v}_{m,n}$ and the surface normal $\mathbf{n}_{m,n}$ becomes more orthogonal. To address this, we adopt a dynamic threshold that adapts based on the surface normal $\mathbf{n}_{m,n} \in \mathbb{R}^3$ as:

$$d_{m,n}^{\text{DBSCAN}} = \frac{2 \tan \gamma^{\text{div}} \max \left\{ ||\mathbf{o}|| \text{ if } \mathbf{o} \in O_{m,n} \right\}}{\mathbf{n}_{m,n}^T \mathbf{v}_{m,n}}. \tag{2}$$

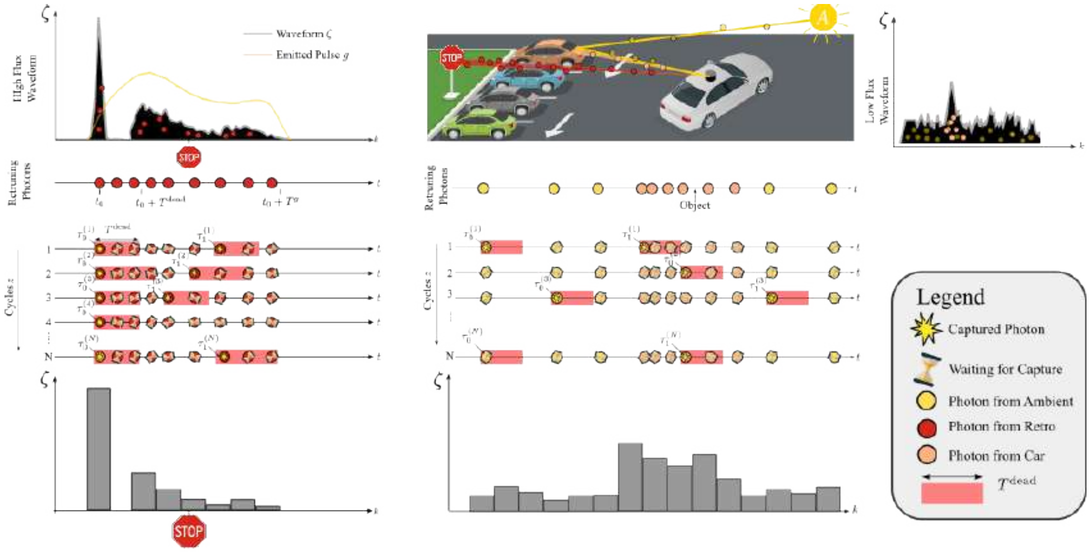
Considering our LiDAR domain, this equation can be seen as the approximation of the pulse width of the LiDAR pulse returning to the sensor. Following this analogy, upright objects lead to shorter pulse widths, and areas nearly parallel to the sensor rays (e.g., ground) lead to longer pulse widths.

Centroid Distances Computation After following the procedure, we obtain clusters for all points $\mathbf{o}$. Then, for every pixel, we calculate the median distance to every cluster, thereby computing ground truth distances. We map these distances to 3D, where we obtain the final multi-echo point cloud $\mathbf{O}_{\text{GT}}$. Figure 1 shows on the right such a multi-echo ground truth point cloud.

3 PROTOTYPE LIDAR SENSOR

Figure 2 shows the MOVIA™ L LiDAR sensor used in our experiments. It is an automotive-grade, solid-state full-waveform LiDAR (FWL) sensor that digitizes the complete optical return waveform during acquisition. The sensor integrates two focal plane arrays: a VCSEL array for emission and a SPAD array for detection. Both arrays have matching dimensions, enabling the use of a shared optics tube in a bi-axial configuration. Emitter–receiver pixel pairs are sequentially activated, enabling electronic scanning of the field of view without any moving parts.

In its default configuration, the sensor discards waveform data after transferring the point cloud and additional metadata to downstream processing. For our experiments, we reconfigure the sensor to stream full waveform data via a debug interface.



(a) High Flux Behaviour

(b) Nominal Low Flux Behaviour

Fig. 3. (a) Measurement in high-flux conditions. Our sensor can capture multiple returning photons within a single cycle. After detecting the first photon, the sensor enters a dead time T_{dead}, during which subsequent photons are registered with reduced probability. In scenes with retroreflective materials, this behavior results in a pronounced primary peak, followed by a secondary peak due to delayed detections after T_{dead}. We overlay the measured waveform with the emitted pulse g and illustrate photon detections across multiple laser cycles, highlighting this temporal structure. (b) Measurement in nominal low-flux conditions. In this regime, the probability of multiple returns within a cycle is negligible. Each SPAD in our free-running FWL system integrates returning photons across N consecutive pulses, registering both signal and ambient light photons. By aggregating all trigger events per bin k over the N cycles, a stable waveform ζ emerges in which the object becomes visible without saturation effects. Unlike in high-flux scenarios, the dead time T_{dead} does not distort the waveform.

3.1 Measurement Principle in Low-Flux Conditions

Figure 3(b) illustrates the sensing principle under nominal low-flux conditions. For each pixel (m, n), the sensor emits N consecutive short laser pulses g over cycles $z \in 1, \ldots, N$. Photons reflected from the scene, along with ambient light photons, arrive at the detector and may trigger the SPADs. Since the total acquisition time T is much shorter than the ambient variation timescale T_a, we assume constant ambient flux $a(t) \approx a$ over the N cycles.

The SPADs operate in free-running mode [Tontini 2024], meaning they can detect multiple photons per cycle. Each detection generates a timestamp $\tau_e^{(z)}$ via a time-to-digital converter (TDC), and after each trigger event, the SPAD enters a dead time T_{dead} during which it is inactive. Because of the asynchronous design, detections can resume within the same cycle after T_{dead}.

By aggregating all timestamps into temporal histogram bins k across the N cycles, we obtain the waveform ζ.

This accumulation forms a stable waveform in which the object peak becomes visible, even in the presence of ambient noise. Unlike first-photon LiDAR systems [Gupta et al. 2019], which record only the earliest trigger and suffer from pile-up effects, our FWL captures a more faithful response. As shown in Figure 4, increasing sunlight only raises the noise floor without obscuring the object signal.

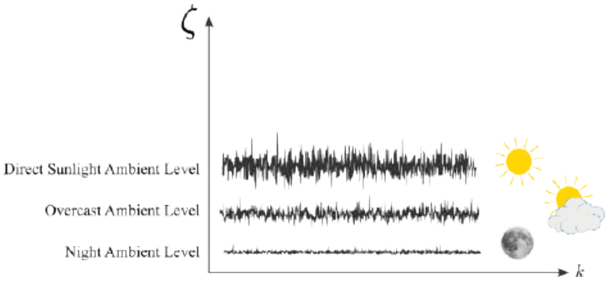


Fig. 4. Illustration of the ambient light contribution for three scenarios: moon-lit night, overcast day, and direct sunlight.

3.2 Measurement Behavior in High-Flux Conditions

In real-world outdoor environments, the low-flux assumption often breaks down. Strong reflectors, such as traffic signs or retroreflective tape, return a burst of photons within a very narrow time window, potentially saturating the SPADs. Figure 3(a) depicts the sensor's behavior under high-flux conditions, such as with retroreflective targets. In these high-flux conditions, the sensor's free-running SPADs enter a dead time after the first detection and fail to register subsequent photons immediately.

The first detected photon consistently triggers a response, creating a sharp, narrow peak. Subsequent photons arriving within the dead time T_{dead} are ignored, and those arriving afterward are detected with reduced probability, forming a secondary peak. This non-linear response—specific to free-running SPADs under saturation—is clearly visible in the waveform, and we model it by learning a custom scene response $\hat{h}$ described in Eq. 20 of the main document.

Our modeling of the estimated total photon flux in Eq. 20 assumes that low- and high-flux returns are distinguishable by a threshold B_T. This assumption is validated with Fig. 5, where the distribution of the intensities in a captured scene is shown over the measurable intensity range (0–1). High-flux returns saturate the detector, contributing only to the rightmost side of the histogram in Fig. 5. However, the majority of peaks are from low-flux returns that contribute to the left side of the histogram. In contrast, almost no peaks are in the center of the LiDAR's intensity range. This clear separation enables the usage of threshold B_T in the modeling, as the measurements do not approach this B_T but stay either well below or well above it.

3.3 Noise Modeling

In the low-flux regime, the measured waveform $\zeta_{\mathbf{p}}$ can be described as a random variable with respect to $\lambda_{\mathbf{p}}$. As described in the main document in low-flux conditions, the response $\zeta_{\mathbf{p}}^{L}$ is modeled as a per-bin-independent Poisson distribution[Pediredla et al. 2018; Rapp and Goyal 2017] of $\lambda_{\mathbf{p}}$, that is

$$\zeta_{\mathbf{p}}^{L}[k] = \text{Poisson}(\lambda_{\mathbf{p}}[k]), \tag{3}$$

allowing us to use the observed sensor signal as unbiased supervision for our model.

In 3.2 we introduced the learned scene response $\hat{h}$, which allows to approximate $\zeta_{\mathbf{p}}^{H}$ from the main document which was defined as,

$$\zeta_{\mathbf{p}}^{H}[k] = \mathcal{P}\left(\int_{k\Delta}^{(k+1)\Delta} h(t;\Theta)\delta\left(t - 2\frac{d}{c}\right) + a_{\mathbf{p}}\, dt\right). \tag{4}$$

Our approximation therefore, becomes,

$$\zeta_{\mathbf{p}}^{H}[k] = (\hat{h} * w) + \mathcal{A}_{\mathbf{p}} + \gamma, \tag{5}$$

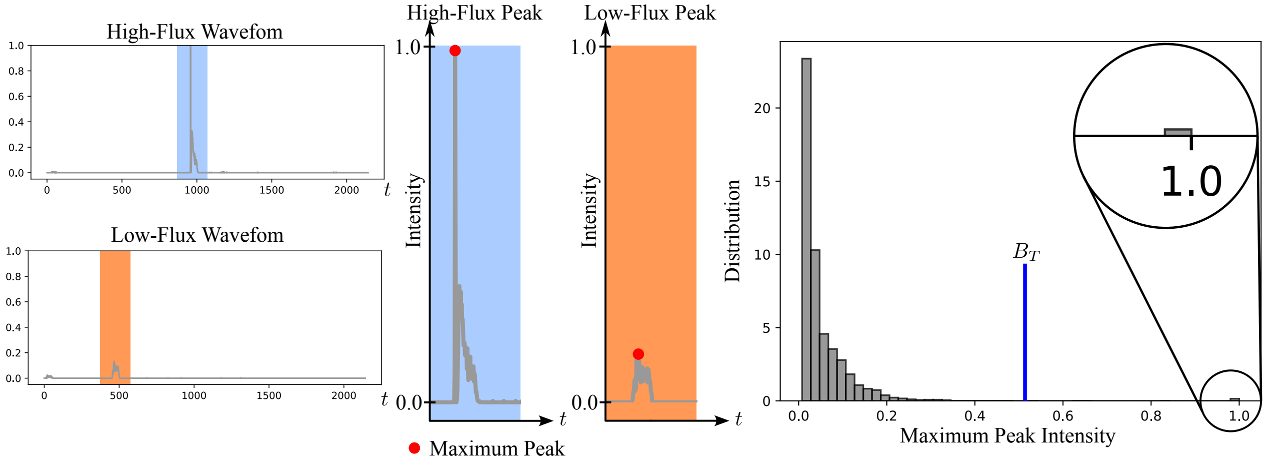


Fig. 5. Intensity distribution in high- and low-flux conditions. Peaks of high-and low-flux conditions are clearly separable. Most peaks belong to the low-flux regime, as shown on the left side of the histogram. High-flux peaks saturate the intensity range of the sensor (1.0) and contribute to the rightmost part of the histogram (zoom-in). As low- and high-flux regions are clearly separable, this motivates introducing the threshold B_T.

with w from Eq. 19 in the main document and γ being a normally distributed random variable modeling the standard deviation in high flux.

4 PEAK-FINDING

We employ conventional argmax DSP peak-finding to emulate on-device waveform processing as shown in Fig. 6. Inspired by [Goudreault et al. 2023], the peakfinding ingests waveforms ζ for each ray individually. First, matched filtering by convolving the waveform ζ with the measured emitted laser pulse g is used to increase signal-to-noise ratio, yielding the filtered waveform ζfilt. Then, the contribution of ambient light is removed by subtracting the noise floor, approximated with the median. Next, a distance d' is identified by employing an argmax operation on ζ'_{filt} to identify the highest peak. The height of the peak - the intensity - is denoted with I'. We then threshold with the intensity threshold V_{min} and only keep distances d with an intensity value above this threshold. As a result, the conventional DSP can suppress points in sky regions.

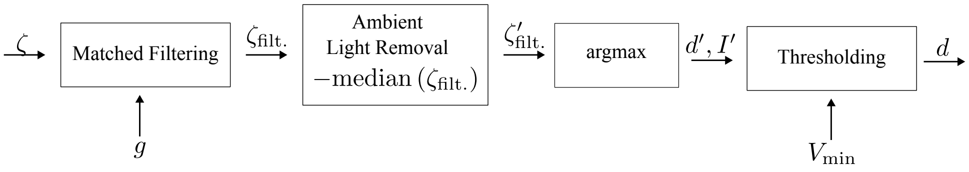


Fig. 6. Peak-Finding applied to the waveform captured with our full waveform SPAD lidar to derive sensor-derived point cloud. The peakfinding algorithm ingests the waveform ζ and searches for the highest peak to estimate the distance d to a target.

5 METRICS

We perform evaluations of the reconstruction on a pointcloud level. To this end, we compare the predicted pointclouds $\mathbf{O}_{\text{pred}}$ with the projected GT pointclouds $\mathbf{O}_{\text{GT}}$. We detail the Chamfer Distance, Distance Accuracy and Recall metrics used in the main paper in the following.

Chamfer Distance (CD) is given by

$$\text{CD}(\mathbf{O}_{\text{pred}}, \mathbf{O}_{\text{GT}}) = \frac{1}{n_{\text{pred}}} \sum_{i}^{n_{\text{pred}}} \|\mathbf{o}_i - \text{NN}(\mathbf{o}_i, \mathbf{O}_{\text{GT}})\| + \frac{1}{n_{\text{GT}}} \sum_{j}^{n_{\text{GT}}} \|\mathbf{o}_j - \text{NN}(\mathbf{o}_j, \mathbf{O}_{\text{pred}})\|, \tag{6}$$

where $\mathbf{o}_i \in \mathbb{R}^3$ is a point in the predicted pointcloud $\mathbf{O}_{\text{pred}}$ and $\mathbf{o}_j \in \mathbb{R}^3$ is a point in the ground truth pointcloud $\mathbf{O}_{\text{GT}}$ n_{GT} and n_{pred} denote the number of points in the predicted and ground truth point clouds. The nearest neighbor function NN is defined as

$$\text{NN}(\mathbf{o}, \mathbf{O}') = \text{argmin}_{\mathbf{o}' \in \mathbf{O}'} \|\mathbf{o} - \mathbf{o}'\|, \tag{7}$$

for an arbitrary point $\mathbf{o}$ and a point cloud $\mathbf{O}'$.

The Distance Accuracy is defined as a one-sided CD, given as

$$\text{Distance Accuracy}(\mathbf{O}_{\text{pred}}, \mathbf{O}_{\text{GT}}) = \frac{1}{n_{\text{pred}}} \sum_{i}^{n_{\text{pred}}} \|\mathbf{o}_i - \text{NN}(\mathbf{o}_i, \mathbf{O}_{\text{GT}}), \tag{8}$$

for a point $\mathbf{o}_i$ in the predicted pointcloud.

Recall of the predicted point cloud is calculated to estimate the ratio of the scene that is correctly reconstructed. To this end, we must classify whether a predicted point is a true positive. We do this by searching for a nearest neighbor in the GT pointcloud, defined by

$$\text{NN}(\mathbf{o}_{\text{pred}}, \mathbf{O}_{\text{GT}}) < d_{\text{true}}, \tag{9}$$

where $d_{\text{true}} = 40$cm. False negative points are classified by evaluating whether a ground truth point has a predicted point in its neighborhood, such that

$$\text{NN}(\mathbf{o}_{\text{GT}}, \mathbf{O}_{\text{pred}}) >= d_{\text{true}}. \tag{10}$$

By counting the number of true positive and false negative points, Recall is calculated using the standard formula.

6 BASELINES

We train all baselines with their default training settings. For TransientNerF [Malik et al. 2023], we replace the required convolution kernel in their image formation model with the true measured pulse $\hat{g}$ to ensure fair evaluation. Pointcloud-based reconstruction methods ingest the sensor-derived point clouds after peakfinding (see above) as detailed in Fig. 8 (Main). We train [Huang et al. 2023] and [Tao et al. 2024] using the pointcloud extracted from the transient measurements through [Goudreault et al. 2023]. The RGB based methods have instead been trained with captures acquired using the reference RGB camera presented in Sec. 5 (Main) and visualized in Fig.3 (Main). We run COLMAP to extract camera poses that we use during the optimization.

7 ADDITIONAL RESULTS

We show in Fig. 7 additional comparison results of our method against the employed baselines for scene reconstruction tasks. Specifically, we show distance maps for the different scenes presented in Fig. 8 (Main) for improved comparisons. As can be seen in the figure, our method achieves more accurate depth reconstruction compared to the baselines. The distance map visualization reveals that our method enables superior reconstruction results. It produces dense distance maps, while peak finding and NFL suffer from large regions, especially on the ground, that remain insufficiently reconstructed.

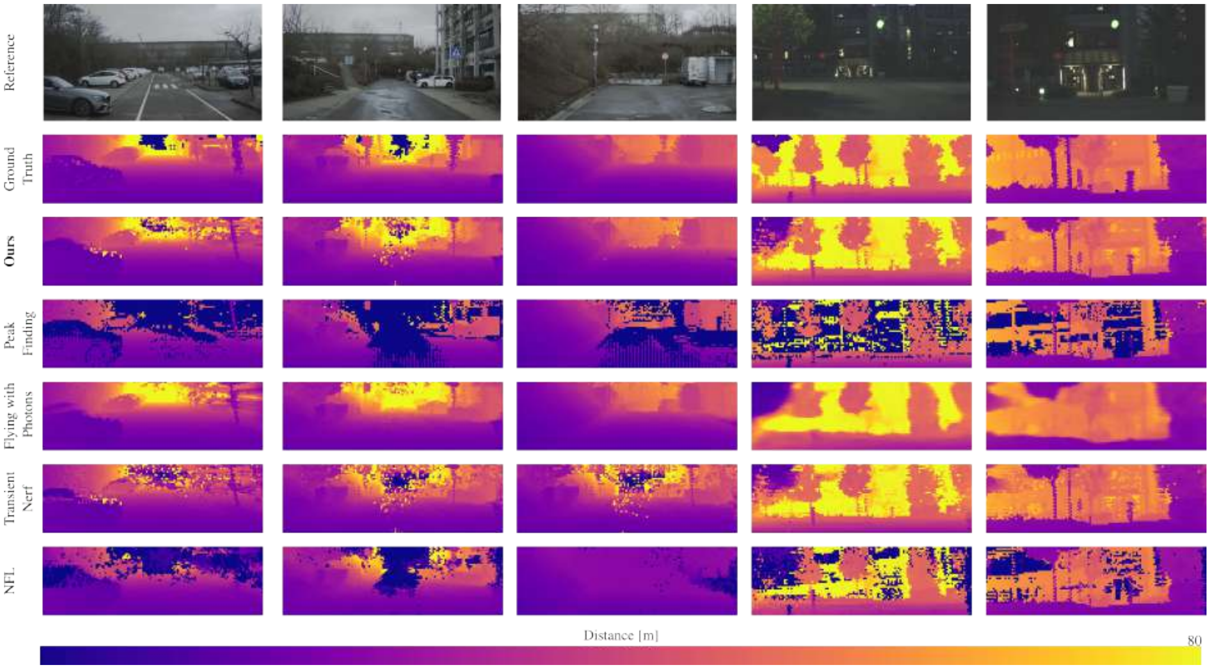


Fig. 7. Qualitative comparison of predicted depth maps, row-wise, for our proposed method (Transient LASSO) and recent baseline methods. For the corresponding point cloud visualization, we refer to Fig. 8 of the main manuscript.

8 ADDITIONAL ABLATION EXPERIMENTS

We include additional ablation experiments and report them in Tab 4. Specifically, we train our proposed method without different losses, namely the ambient light consistency and the retroreflector losses, as well as varying the size of the embedding χ. Notably, our final configuration of losses and network hyperparameters achieves the best performance.

Method	**CD [m]** ↓	**Dist. Acc. [m]** ↓	**Recall [%]** ↑
w/o $\mathcal{L}_a$	0.831	0.512	56.77
w/o $\mathcal{L}_h$	0.657	0.410	58.72
$\chi_{dim} = 10$	0.663	0.386	59.12
Final	**0.642**	**0.359**	**60.12**

Table 4. Quantitative ablation results of our method for 3D scene reconstruction.

REFERENCES

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, Vol. 96. 226–231.

Felix Goudreault, Dominik Scheuble, Mario Bijelic, Nicolas Robidoux, and Felix Heide. 2023. LiDAR-in-the-Loop Hyperparameter Optimization . In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, Los Alamitos, CA, USA, 13404–13414. https://doi.org/10.1109/CVPR52729.2023.01288

Anant Gupta, Atul Ingle, Andreas Velten, and Mohit Gupta. 2019. Photon-Flooded Single-Photon 3D Cameras. In *Proc. CVPR*.

Shengyu Huang, Zan Gojcic, Zian Wang, Francis Williams, Yoni Kasten, Sanja Fidler, Konrad Schindler, and Or Litany. 2023. Neural LiDAR Fields for Novel View Synthesis . In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE Computer Society, Los Alamitos, CA, USA, 18190–18200. https://doi.org/10.1109/ICCV51070.2023.01672

Sagi Katz, Ayellet Tal, and Ronen Basri. 2007. Direct visibility of point sets. In *ACM SIGGRAPH 2007 papers*. 24–es.

Anagh Malik, Parsa Mirdehghan, Sotiris Nousias, Kiriakos N. Kutulakos, and David B. Lindell. 2023. Transient neural radiance fields for lidar view synthesis and 3D reconstruction. In *Proceedings of the 37th International Conference on Neural Information Processing Systems* (New Orleans, LA, USA) *(NIPS '23)*. Curran Associates Inc., Red Hook, NY, USA, Article 3133, 13 pages.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)* 41, 4 (2022), 1–15.

Adithya K. Pediredla, Aswin C. Sankaranarayanan, Mauro Buttafava, Alberto Tosi, and Ashok Veeraraghavan. 2018. Signal Processing Based Pile-up Compensation for Gated Single-Photon Avalanche Diodes. arXiv:1806.07437 [physics.ins-det] https://arxiv.org/abs/1806.07437

Joshua Rapp and Vivek K Goyal. 2017. A Few Photons Among Many: Unmixing Signal and Noise for Photon-Efficient Active Imaging. *IEEE Transactions on Computational Imaging* 3, 3 (2017), 445–459. https://doi.org/10.1109/TCI.2017.2706028

Dominik Scheuble, Hanno Holzhueter, Steven Peters, Mario Bijelic, and Felix Heide. 2025. Lidar Waveforms are Worth 40x128x33 Words. *2025 IEEE/CVF International Conference on Computer Vision (ICCV*.

Tang Tao, Longfei Gao, Guangrun Wang, Yixing Lao, Peng Chen, Hengshuang Zhao, Dayang Hao, Xiaodan Liang, Mathieu Salzmann, and Kaicheng Yu. 2024. LiDAR-NeRF: Novel LiDAR View Synthesis via Neural Radiance Fields. In *Proceedings of the 32nd ACM International Conference on Multimedia* (Melbourne VIC, Australia) *(MM '24)*. Association for Computing Machinery, New York, NY, USA, 390–398. https://doi.org/10.1145/3664647.3681482

Alessandro Tontini. 2024. *Advanced techniques for SPAD-based CMOS d-ToF systems*. Ph. D. Dissertation. Università degli studi di Trento.

Ignacio Vizzo, Tiziano Guadagnino, Benedikt Mersch, Louis Wiesmann, Jens Behley, and Cyrill Stachniss. 2023. KISS-ICP: In defense of point-to-point ICP–simple, accurate, and robust registration if done the right way. *IEEE Robotics and Automation Letters* 8, 2 (2023), 1029–1036.

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