Lidar Waveforms are Worth 40x128x33 Words (Supplementary Material)

Dominik Scheuble[1,4] Hanno Holzhüter[2] Steven Peters[4] Mario Bijelic[3,5] Felix Heide[3,5]
[1]Mercedes-Benz AG [2]MicroVision [3]Torc Robotics [4]TU Darmstadt [5]Princeton University

In this supplemental document, we present additional details on the sensor, the testing and training dataset, the full waveform lidar (FWL) forward model, the neural DSP, and the training process. We also provide additional quantitative and qualitative results to support the findings from the main manuscript.

Contents

1. Sensor and Dataset

We add details of the used sensor in Sec. 1.1, a thorough explanation of ground truth generation in Sec. 1.2, and an overview of the splits used for fine-tuning and testing in Sec. 1.3.

1.1. Sensor

We provide details of the used sensor in Sec. 1.1.1 and extend the discussion of sensor effects in Sec. 1.1.2.

1.1.1. Sensor Configuration

Fig. 1 shows the MOVIA™ L lidar [17] sensor we use for our experiments. It is an automotive production-grade, solid-state FWL sensor that processes the full waveform data digitally during acquisition. In off-the-shelf operation, the sensor discards waveform data once the transfer of point cloud, as well as additional data to downstream algorithms, is complete. In this work, we configure the MOVIA™ L to stream out full waveform data. The limited data rate, however, imposes a reduction in the number of waveforms per frame from 80×128 down to 40×128. Waveform data is streamed through a debug interface, which reduces the off-the-shelf frame rate of 15 Hz (as mentioned in the main paper) to approximately 0.5 Hz due to limitations on the data transfer rate. This experimental capturing setup prevents waveform acquisition in highly dynamic driving scenarios and with moving objects, as the waveforms would suffer from severe motion blur. We refer to Sec. 5 for a discussion on how we envision a potential real-time application of streaming waveform data in a production setting. All hardware of the sensor remains untouched.

Figure 1. **Experimental FWL Sensor.** We adapt a MicroVision MOVIA™ L Sensor [17] to output full waveform data.

1.1.2. Sensor Measurement Principle

Fig. 2 depicts the measurement principle of our FWL under low-flux conditions. For every pixel (m, n), the sensor emits N consecutive laser pulses g into the scene during cycles $z \in \{1, 2, 3, ..., N\}$. The emitted photons are reflected from an object in the scene and return together with ambient light photons, specified by the ambient flux $a(t)$, to the sensor. As the measurement time T is much smaller than the timescale T_a at which the ambient light changes ($T << T_a$), we assume the ambient light to be constant over the N laser emission cycles, such that $a(t) \approx a$. In every laser emission cycle z, a time-to-digital-converter (TDC) in the sensor generates timestamps $\tau_e^{(z)}$ of a trigger event e generated by a returning photon. Sorting timestamps of trigger events into a temporally binned histogram and counting the number of trigger events per bin k yields the waveform κ, as denoted in the main paper. Despite SPADs being able to detect single photons, not every incident photon generates a trigger event; instead, photons cause trigger events $\tau_e^{(z)}$ with a specific detection probability μ. After a trigger event $\tau_e^{(z)}$, the SPAD is quenched [2, 3] and cannot detect further photons until a dead time T^{dead} has passed. As the SPADs in our FWL operate in free-running asynchronous mode [21], further photons within the same laser emission cycle can be detected. By aggregating all trigger events $\tau_e^{(z)}$ per temporal time bin k over the cycles z, the full waveform κ is outputted by the sensor as follows

$$\kappa(k) = \sum_{z=1}^{N} \kappa_z(k). \tag{1}$$

$\kappa_z(k)$ is the aggregated measurement from the z cycle given by

$$\kappa_z(k) = \sum_{i} \mathbf{1}\{\tau_i^{(z)} \in k\}, \tag{2}$$

where the identity $\mathbf{1}$ is one for all trigger events $\tau_i^{(z)}$ falling into the bin k and zero otherwise. From Fig. 2, the benefit of operating in free-running SPADs becomes visible. The object peak is clearly visible in the waveform, as photons from the object can be detected even after initial trigger events from ambient light have occurred. This is different in first-photon sensors [9], where only the initial trigger event is recorded and the SPAD is deactivated for the remainder of the cycle z. In first-photon sensors, photons from strong ambient light cause pile-up distortions that heavily skew the waveform to earlier time bins. In contrast, in our FWL, ambient light causes only a constant rise of the noise floor, and the object peak remains visible. This is confirmed by Fig. 3, which shows three waveforms from pixels pointing towards the sky. An increasing amount of sunlight only causes the described rise in noise floor.

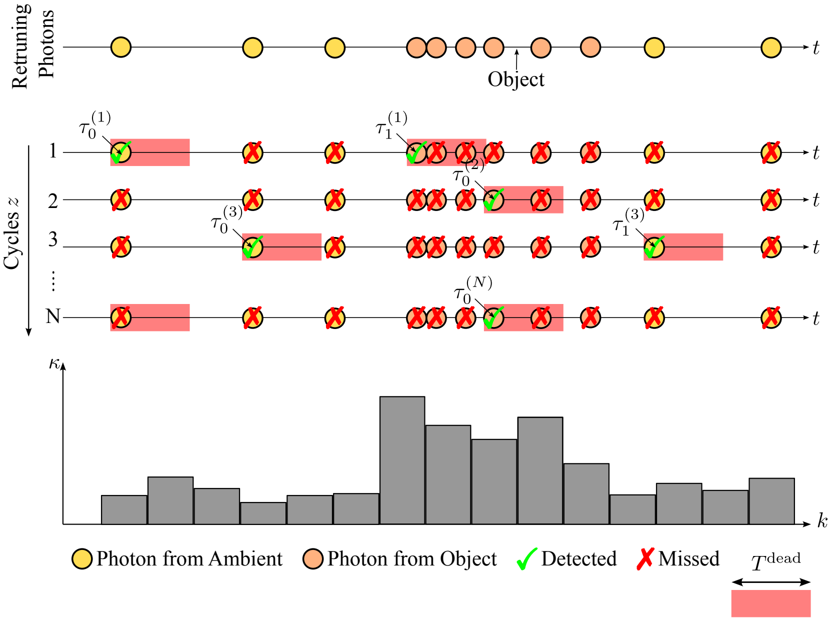


Figure 2. **Sensor Measurement Principle.** We illustrate the sensing of our FWL in low-flux conditions. For each pixel, our FWL emits N consecutive pulses. The emitted photons are reflected and are registered together with photons from ambient light. As the SPADS in our FWL are free-running, photons from the object can be registered even after initial trigger events from ambient light. When counting all trigger events per bin k over the N pulse emission cycles, a waveform κ emerges, where the object becomes visible.

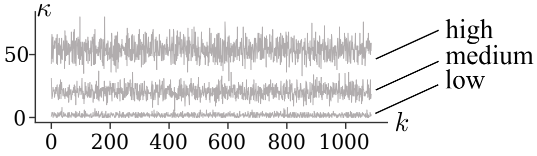


Figure 3. **Waveforms in Ambient Light.** An increase in ambient light causes a constant rise of the noise floor and no skew of the waveform shape.

1.2. Ground Truth Generation

To supervise and evaluate our neural DSP, we rely on a multi-echo point cloud as ground truth. For the outdoor dataset, we accumulate lidar maps from a Velodyne-VLS 128 from different positions using [23] for ground truth. In the weather chamber, we use scans from a Leica ScanStation P30 laser scanner (1.2mm + 10 parts per million (ppm) distance accuracy and approximately 157M points per scan) as ground truth. Independent of the source of the ground truth scans, we project the dense ground truth scans into the FoV of the sensor. To this end, we follow the pipeline as shown in Fig. 4.

Alignment As a first step, we localize our sensor in the dense ground truth scan. To this end, we align the sensor-derived point cloud from the FWL to the ground truth scan using a standard Iterative-Closest-Point (ICP) algorithm from [25], yielding the transformation $\mathbf{T}_{\text{ICP}} \in \mathbb{R}^{4\times4}$. To ensure the best possible alignment, we only use high-SNR points from the sensor-derived point cloud and an initial guess $\mathbf{T}_{\text{initial}}$. For the outdoor dataset, we always capture from a static starting point and thus use the static transformation between Velodyne and FWL as an initial guess. In the weather chamber, the known recording position of the sensor vehicle is used as an initial guess.

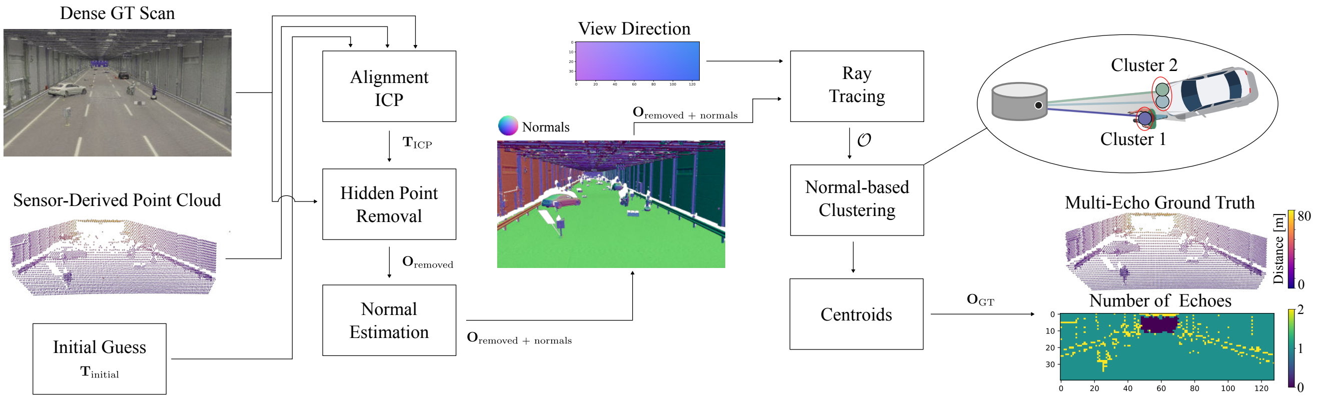


Figure 4. **Multi-Echo Ground Truth (GT) Generation.** We first align the sensor in the dense GT scan using an ICP algorithm. After hidden-point removal, we perform ray tracing and find all points of the GT scan that fall within the diverging laser beam. We then cluster these points to extract the multi-echo ground truth point cloud.

Hidden Point Removal As the ground truth scan is obtained from scans in multiple positions to avoid occlusions, it includes points that are invisible from the current FWL position. As the DSPs are unable to reconstruct these points, they must be removed from the ground truth. To this end, we first project the full ground truth scan into the sensor FoV through $\mathbf{T}_{\text{ICP}}$, then we apply the hidden point removal algorithm presented by Katz *et al.* [13]. The result is a dense ground truth scan $\mathbf{O}_{\text{removed}}$ that features shadow areas behind opaque objects invisible to the FWL, see Fig. 4.

Normal Estimation For a subsequent processing step, surface normals are required. To this end, we apply the normal estimation technique as implemented by [25] to yield the point cloud with per-point normal information $\mathbf{O}_{\text{removed + normals}}$.

Ray Tracing Next, we perform ray tracing with the view direction against the dense ground truth point cloud $\mathbf{O}_{\text{removed + normals}}$ by considering the diverging laser beam. To this end, we check for every viewing direction $\mathbf{v}_{m,n} \in \mathbb{R}^3$ if a point $\mathbf{o} \in \mathbf{O}_{\text{removed + normals}}$ falls into a cone representing the diverging laser beam. Specifically, we add points $\mathbf{o}$ to the set $\mathcal{O}_{m,n}$ such that

$$\mathcal{O}_{m,n} = \left\{ \mathbf{o} \text{ if } \mathbf{v}_{m,n}^T \frac{\mathbf{o}}{\|\mathbf{o}\|} > \cos \gamma^{\text{div}} \right\}, \tag{3}$$

where γ^{div} denotes the beam divergence of the emitted laser beam. Computing $\mathcal{O}_{m,n}$ for all pixels (m, n) yields the set of points $\mathcal{O}$ that are visible for every pixel.

Normal-Based Clustering As a result of the beam divergence, $\mathcal{O}$ likely contains clusters of points at distinct distances. This is indicated in Fig. 4, where the diverging beam hits both a pedestrian and the white car, forming two distinct clusters consisting of a single point and two points, respectively. As the number of clusters is typically unknown beforehand, we apply a DBSCAN [5] clustering algorithm, which does not require a predefined number of clusters. However, DBSCAN requires a maximum distance threshold, d^{DBSCAN}, that allows points to be classified as belonging to the same cluster. We find that using a fixed maximum distance d^{DBSCAN} is unsuitable as points $\mathbf{o}$ on upright objects tend to be closer together than points on the road. This is since the effect of the beam divergence intensifies as viewing direction $\mathbf{v}_{m,n}$ and surface normal $\mathbf{n}_{m,n}$ become increasingly orthogonal. Thus, we opt to choose a dynamic threshold depending on the surface normal $\mathbf{n}_{m,n} \in \mathbb{R}^3$

$$d_{m,n}^{\text{DBSCAN}} = \frac{2 \tan \gamma^{\text{div}} \max \left\{ \|\mathbf{o}\| \text{ if } \mathbf{o} \in \mathcal{O}_{m,n} \right\}}{\mathbf{n}_{m,n}^T \mathbf{v}_{m,n}}. \tag{4}$$

Note that Eq. (4) can be seen as an approximation of the pulse width of the returning laser pulse. Upright objects cause smaller pulse widths, while flat objects elongate the returning pulse.

Centroids Finally, after clustering all points $\mathbf{o}$, we compute multiple ground truth distances per pixel by taking the median distance from every cluster. We then map the distances to 3D, yielding the multi-echo point cloud $\mathbf{O}_{\text{GT}}$. On the right of Fig. 4, we show a multi-echo ground truth point cloud and the number of echoes/clusters per pixel. We find that especially object discontinuities are the source of multiple echoes.

1.3. Dataset Statistics and Splits

As described in the main paper, we capture both an outdoor dataset in different lighting conditions and a controlled environment dataset in a weather chamber. In the weather chamber, we capture both clear-reference recordings and recordings in foggy conditions. We use 198 frames of the outdoor dataset for fine-tuning. We use the controlled environment dataset exclusively for testing. For the test set in foggy conditions, we use 13 frames in different positions with meteorological visibilities of 50 and 80m. For the test set in clear conditions, we combine 20 frames from the outdoor dataset with 18 frames from the controlled environment clear-reference dataset.

For the synthetic dataset, we extract 6480 frames for training and 200 frames for validation from three different CARLA maps. We use 838 frames of another map exclusively for testing. We add retroreflectors and fog simulation to the synthetic data according to Tab. 1.

2. Sensing Forward Model

In the following, we provide additional details on the full waveform lidar imaging model. Specifically, details on low-flux transients are described in Sec. 2.1, high-flux transients in Sec. 2.2, and fog transients in Sec. 2.3. We provide additional validations of the forward model in Sec. 2.4. Finally, we detail in Sec. 2.5 the integration of the forward model into CARLA.

2.1. Low-Flux Transients

In low-flux conditions, we follow Goudreault *et al.* [8] and model the incident photon flux $\psi_{i,j}$ for a pixel (i, j) as

$$\psi_{i,j}(t) = \frac{\rho_{i,j}}{4d_{i,j}^2} g\left(t - 2\frac{d_{i,j}}{\mathrm{c}}\right) + a_{i,j}(t). \tag{5}$$

We refer to the main document for a detailed explanation of the notations. The reflectivity $\rho_{i,j}$ is modeled with the Cook-Torrence model [1], considering both specular and diffuse reflections, such as

$$\rho_{i,j} = \underbrace{\frac{\alpha_{i,j}^4 s_{i,j} \cos\theta_{i,j}}{4[\cos^2\theta_{i,j}(\alpha_{i,j}^4 - 1) + 1]^2[\cos\theta_{i,j}(1 - k_{i,j}) + k_{i,j}]^2}}_{\text{Specular}} + \underbrace{\Omega_{i,j}\cos\theta_{i,j}}_{\text{Diffusive}}, \tag{6}$$

where $s \in [0, 1]$ describes the specular and $\Omega \in [0, 1]$ the diffusive part of the reflectivity. The specular part is defined by material roughness $\alpha \in [0, 1]$ and a parameter $k = (\alpha + 1)^2/8$. Both specular and diffusive reflectivity strongly depend on the incident angle $\theta_{i,j}$ defined by

$$\cos\theta_{i,j} = \mathbf{n}_{i,j} \cdot \mathbf{v}_{i,j}, \tag{7}$$

where $\mathbf{n}_{i,j} \in \mathbb{R}^3$ and $\mathbf{v}_{i,j} \in \mathbb{R}^3$ are the pixel-individual surface normals and viewing direction, respectively. As shown in Fig. 2 (main document), the material parameters α, s and Ω are extracted from CARLA using material cameras [8]. For extraction of the normals $\mathbf{n}_{i,j}$, we rely on the adapted ray-tracer for a FWL model as presented in [19].

To simulate multi-echo waveforms, we linearly combine waveforms from neighboring pixels into a single waveform for every pixel (m, n). This is described with Eq. (6) in the main paper as

$$\psi_{m,n}(t) = \sum_{i,j \in \mathcal{N}(m,n)} K_{i,j}\psi_{i,j}(t). \tag{8}$$

Following [8], we model the spatial profile of the pulse K with a Gaussian such that

$$K_{i,j} = A \times 2^{-\mathcal{R}(i)^2 - \mathcal{R}(j)^2}, \tag{9}$$

where $\mathcal{R}(z) \equiv z \bmod a - 2$ and a is the uneven integer upsampling factor as defined in the main paper. A is a normalization factor such that the sum of $K_{i,j}$ over all indices is 1.

2.2. High-Flux Transients

As discussed in the main paper, objects consisting of highly retroreflective materials cause additional peaks and multipath effects for pixels directly illuminating retroreflective objects and blooming effects on neighboring pixels not illuminating these objects directly. We provide a more detailed analysis on primary, secondary and multipath peaks in Sec. 2.2.1 and for blooming effects in Sec. 2.2.2.

2.2.1. Primary, Secondary and Multipath Peak

High-flux conditions, e.g., returns from retroreflector, affect the captured waveform. For sensors, where the deadtime T^{dead} is smaller than the pulse width T^g, this causes the appearance of multiple distinct peaks in the waveform from the same object. In our particular case, those are two peaks, with the deadtime separating the first and second peaks. As introduced in the main paper, we model this behavior as a combination of a primary peak f^{prim} and a secondary peak f^{sec} such that

$$\kappa'_{m,n}[k] = f^{\text{prim}}(k, d_{m,n}, \theta^{\text{prim}}) + f^{\text{sec}}(k, d_{m,n}, \theta^{\text{sec}}). \tag{10}$$

The vast number of returning photons even at earlier segments of the returning pulse at time t_0, cause initial trigger events τ_0 with expected detection probability of $\mathbb{E}\left(\mathcal{P}\left(\tau_0 = t_0 \mid ...\right)\right) \approx 1$ across multiple cycles z. This causes a large count of events in the histogram κ in the bin containing t_0, resulting in a steeply rising peak, as shown in Fig. 5. As described in the main paper, we denote this peak as f^{prim}. As shown in Fig. 6, we describe the primary peak as a narrow Gaussian distribution given by

$$f^{\text{prim}}(k, d_{m,n}, \theta^{\text{prim}}) = \min\left(a^{\text{prim}} \exp\left(-\frac{k - 2(d_{m,n})/c + \mu^{\text{prim}}}{2(\sigma^{\text{prim}})^2}\right), a_{\text{max}}\right), \tag{11}$$

where $\theta^{\text{prim}} = \left\{a^{\text{prim}}, \sigma^{\text{prim}}, \mu^{\text{prim}}\right\}$ parameterize height, width and offset from the nominal distance $d_{m,n}$. We set $a^{\text{prim}} = 270$ and model saturation by setting $a_{\text{max}} = 255$. As shown in Fig. 7, the primary peak is constantly shifted by half a pulse length to the front of the theoretically returning pulse g. As the primary peak exhibits samples on both its leading and falling edge, we use this as justification to model f^{prim} as a Gaussian distribution, instead of a single saturated bin at the beginning of the returning pulse. We find the offset from the pulse length T^g as $\mu^{\text{prim}} = T^g/2\Delta \approx 20$ and empirically choose $\sigma^{\text{prim}} = 0.8$ to describe the primary peak for our sensor.

The secondary peak is due to the fact that SPADs are operated in asynchronous free-running mode with dead times T^{dead} shorter than the pulse length T^g. For a FWL with longer dead times, modeling this peak can be omitted. As indicated by Fig. 2 (main paper) and Fig. 5, this causes additional trigger events τ_i for $i > 0$ (after the events from the primary peak) within the same returning pulse. However, non-idealities, such as slow voltage ramp-up, prevent full SPAD recovery in every measurement cycle (see e.g. fourth cycle in Fig. 5). This limits the rise of the secondary peak to lower amplitudes. Additionally, the fall-off in intensity of the emitted pulse causes fewer returning photons in later segments of g, as indicated by Fig. 5. This results in a reduced number for photons arriving during $t \in [t_0 + T^{\text{dead}}, t_0 + T^g]$. Additionally, varying dead times (see e.g. second cycle) due to SPAD imperfections, prevent another steep rise at time $t_0 + T^{\text{dead}}$. Similar to the primary peak, the rise of the secondary peak experiences intermediate samples on the rise to its maximum, see Fig. 7. We find that both the rise and the fall-off are well captured by an exponentially modified Gaussian distribution given by

$$f^{\text{sec}}(k, d_{m,n}, \theta^{\text{sec}}) = \frac{h^{\text{sec}}\sigma^{\text{sec}}}{\tau^{\text{sec}}}\sqrt{\frac{\pi}{2}} \exp\left(\frac{1}{2}\left(\frac{\sigma^{\text{sec}}}{\tau^{\text{sec}}}\right)^2 - \frac{k - \mu^{\text{sec}} - Td_{m,n}}{\tau^{\text{sec}}}\right) \operatorname{erfc}\left(\frac{1}{\sqrt{2}}\left(\frac{\sigma^{\text{sec}}}{\tau^{\text{sec}}} - \frac{k - \mu^{\text{sec}} - Td_{m,n}}{\sigma^{\text{sec}}}\right)\right), \tag{12}$$

where $\theta^{\text{sec}} = \{h^{\text{sec}}, \mu^{\text{sec}}, \sigma^{\text{sec}}, \tau^{\text{sec}}\}$ parameterize peak height, offset from the nominal distance $d_{m,n}$, peak width and steepness of the falling peak, respectively. The parameter $T = 2c/\Delta$ defines the mapping from distance in meters to temporal bin units. Although, we find that the majority of secondary peaks are well captured by the shape of the exponentially modified Gaussian, see Fig. 6, no clear distribution emerges for θ^{sec}. We thus vary the parameters randomly during simulation as described in Sec. 2.5 to allow the neural DSP to generalize to a wide range for secondary peak shapes.

Finally, according to the main paper, multipath effects emerging from multiple reflections between sensor and retroreflective objects are included by extending Eq. (10) such that

$$\kappa''_{m,n}[k] = \kappa'_{m,n}[k] + \frac{r}{d^2_{m,n}} g'(k - 4\Delta\frac{d_{m,n}}{\mathrm{c}}), \tag{13}$$

where $r = 3.9872$ denotes a constant reflectivity factor. Technically, more than one multipath peak can occur. Practically, the attenuation - inversely proportional to the squared distance $d_{m,n}$ - prevents more than one multi-path peak in all captured scenarios. We simulate quantum shot noise in the Poisson process of the recorded trigger events [11] to model realistic waveform sensing.

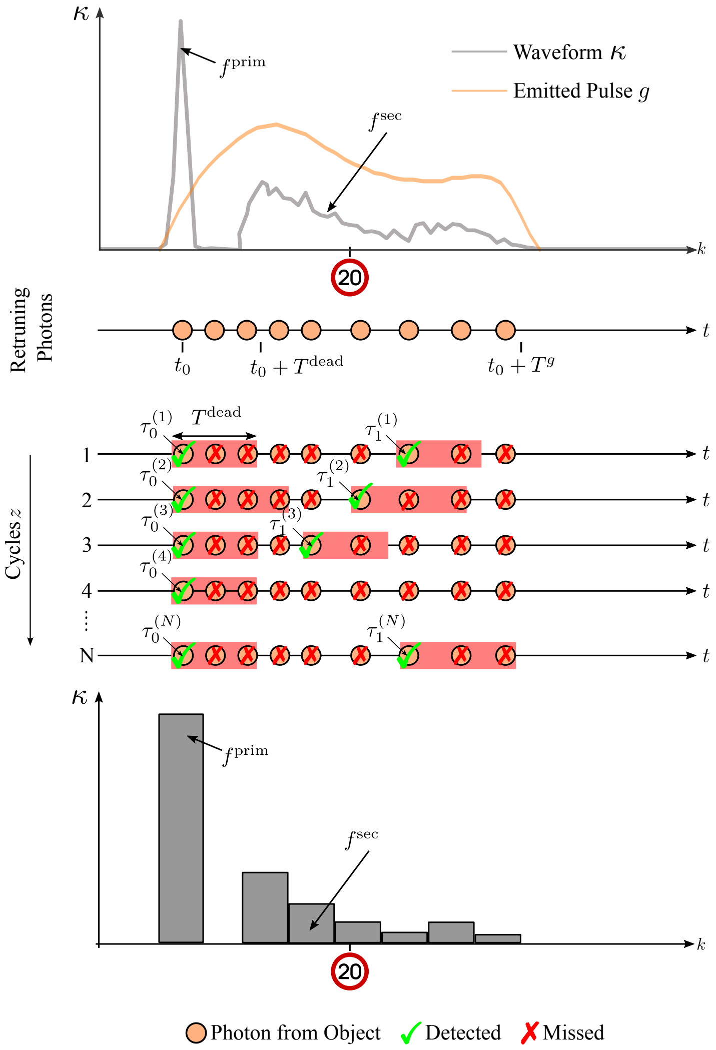


Figure 5. **Measurement in High-Flux Conditions.** Retroreflectors cause a primary and secondary peak. We show the waveform as measured by the sensor overlaid with the emitted pulse g. In the center, the detected and missed photons across various laser emission cycles are illustrated. Note how the first photon is always detected resulting in the primary peak. After the dead time T^{dead}, photons are detected with reduced probability, leading to the secondary peak.

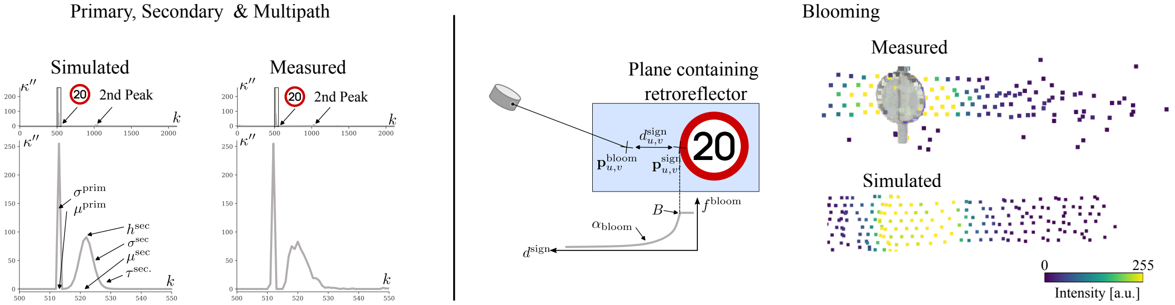


Figure 6. **High-Flux Transients.** We compare simulated primary and secondary peak to the measured ones for a pixel directly illuminating a retroreflector (left). Pixels that do not illuminate retroreflective objects directly are affected by blooming. We compare measured and simulated blooming effects for a round traffic sign (right) and find good correspondence for both cases.

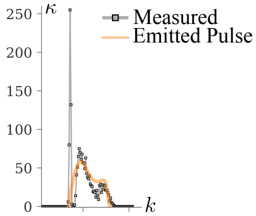


Figure 7. **Measured Primary and Secondary Peak.** The measured primary peak exhibits samples on its leading and falling edge. We use this as justification to model the primary peak as a Gaussian distribution. The steep rise and fall-off of the secondary peak is well captured by the exponentially modified Gaussian. The secondary peak spans until the end of the emitted pulse g as depicted in the overlay.

2.2.2. Blooming

As described in the main paper, blooming causes incident photon flux on neighboring pixels that do not illuminate retroreflective objects directly. The incident photon flux on a neighboring pixel (u, v) is defined as

$$\psi_{u,v}^{\text{bloom}}(t) = f^{\text{bloom}}(d_{u,v}^{\text{sign}})g(t - 2\frac{\|\mathbf{p}_{u,v}^{\text{bloom}}\|}{\text{c}}). \tag{14}$$

To this end, we probe where the ray - defined by the viewing direction - intersects with the plane containing the retroreflector at $\mathbf{p}_{u,v}^{\text{bloom}}$ as shown in Fig. 6. Then we find the closest point on the retroreflector $\mathbf{p}_{u,v'}^{\text{sign}}$ along the same line u due to the sensor's line-wise readout. Next, the Euclidean distance $d_{u,v}^{\text{sign}}$ given as

$$d_{u,v}^{\text{sign}} = \|\mathbf{p}_{u,v}^{\text{bloom}} - \mathbf{p}_{u,v'}^{\text{sign}}\|, \tag{15}$$

is used to quantify the magnitude of the blooming effect. Analogous to [6], we observe an exponential decrease of the blooming effects with increasing distance from the retroreflective object, which we model as

$$f^{\text{bloom}}(d_{u,v}^{\text{sign}}) = B\exp(-\alpha^{\text{bloom}}d_{u,v}^{\text{sign}}), \tag{16}$$

where B and α^{bloom} define magnitude and drop-off rate. As a result, blooming affects an increasing amount of neighboring pixels when the retroreflector is in close proximity to the sensor, but affects fewer pixels for far-away retroreflective objects due to greater attenuation caused by increasingly larger distances $d_{u,v}^{\text{sign}}$. As shown in Fig. 6 qualitatively, this model aligns well with the measured effect. Similarly to the secondary peak, we observe that no clear distribution for B and α^{bloom} emerges, and thus vary the parameters during training. We employ Poisson sampling to noise the contribution of the blooming effect.

2.3. Scattering Transients

In foggy conditions, we follow [12] and model the initial scattering peak with an exponentially modified Gaussian distribution f^{fog} defined as

$$f^{\text{fog}}(t, \theta^{\text{fog}}) = \frac{h^{\text{fog}}\sigma^{\text{fog}}}{\tau^{\text{fog}}}\sqrt{\frac{\pi}{2}}\exp\left(\frac{1}{2}\left(\frac{\sigma^{\text{fog}}}{\tau^{\text{fog}}}\right)^2 - \frac{t-\mu^{\text{fog}}}{\tau^{\text{fog}}}\right)\operatorname{erfc}\left(\frac{1}{\sqrt{2}}\left(\frac{\sigma^{\text{fog}}}{\tau^{\text{fog}}} - \frac{t-\mu^{\text{fog}}}{\sigma^{\text{fog}}}\right)\right), \tag{17}$$

where $\theta^{\text{fog}} = \left\{h^{\text{fog}}, \mu^{\text{fog}}, \sigma^{\text{fog}}, \tau^{\text{fog}}\right\}$ parameterize the initial scattering peak's amplitude, offset, width and steepness of the falling edge. We fit Eq. (17) to a holdout calibration set of foggy waveforms collected in the fog chamber and find the distributions as shown in Fig. 8, which we sample from during training of the neural DSP. However, the effect of fog extend beyond the initial scattering peak. As described by [10], fog causes an exponential attenuation of the light returning from actual objects in the scene such that

$$\psi_{m,n}^{\text{fog}}(t) = \exp(-\alpha^{\text{fog}}t)\psi_{m,n}(t) + f^{\text{fog}}(t, \theta^{\text{fog}}) + \beta^{\text{fog}}, \tag{18}$$

where α^{fog} quantifies the fog thickness. Although the fog thickness is controlled in the fog chamber, we vary α^{fog} during training to allow the neural DSP to generalize to a wide variety of different fog thicknesses. The effect of the constant term β^{fog} is illustrated in Fig. 8. The real foggy waveform exhibits a great number of low-intensity peaks throughout the measurement that are not present in the corresponding waveform in clear conditions. This is likely related to multi-scattering in foggy conditions. By adding β^{fog} and subsequent noising with Poisson distributed noise, this multi-scattering effect can also be observed in the simulated foggy waveform as shown in Fig. 8.

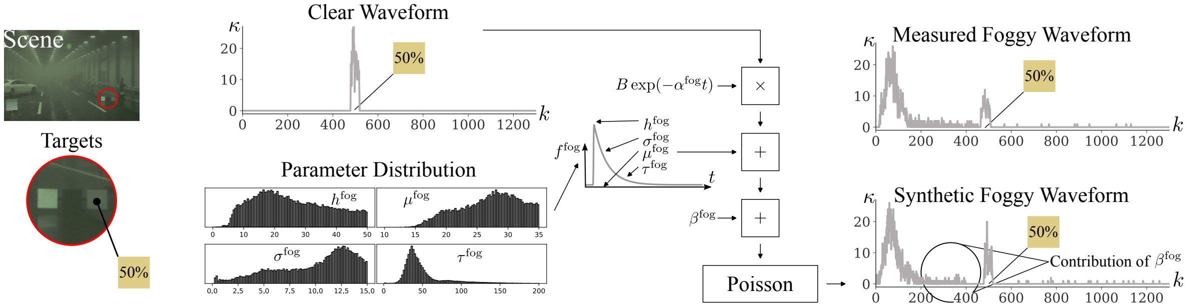


Figure 8. **Scattering Transients**. We compare a waveform for a pixel illuminating a diffuse target with 50% reflectivity in clear conditions with the real and simulated foggy conditions waveforms. Qualitatively, we observe a good correspondence between real and simulated fog by including the initial scattering peak, exponential attenuation and multiple scattering in our model given by Eq. (18). We fit the initial scattering peak to a holdout calibration set of waveforms and extract distributions for the parameters θ^{fog}.

2.4. Validation of Forward Model

The ablation studies in Tab. 3 (main document) validate realistic waveform simulation, as the simulated data directly aids in improving reconstruction on a large number of real-world waveforms captured in various conditions. It is challenging to compare the accuracy of simulated waveforms to real-world measured waveforms directly as a large number of scene parameters and the reflectivity required for the forward model are unknown in outdoor driving scenarios. As a result, parameters are sampled during training - as described in Sec 2.5 - which has proven to be effective as the neural DSP is able to generalize robustly to different outdoor scenarios.

Nevertheless, we report qualitative and quantitative comparisons on selected waveforms captured from diffuse reflection targets with calibrated reflectivity (90%, 5%, 50%) in the controlled weather chamber environment as shown in Fig. 9. To this end, we consider the low-flux model and extract normals and distance from the dense ground truth scan. As the calibrated targets are perfectly diffuse, we consider only the diffuse part of the reflectance ρ in Eq. 6. The qualitative findings from Fig. 9 indicate a good agreement between the simulation and the measured waveform. For quantitative evaluation, we use root-mean-square error (RMSE) as an evaluation metric between the measured $\kappa^{\text{meas.}}_{m,n}$ and the simulated waveform $\kappa^{\text{sim.}}_{m,n}$ as

$$\text{RMSE}_{m,n} = \sqrt{\frac{\sum_{k=0}^{T-1}\left(\kappa^{\text{sim.}}_{m,n}[k] - \kappa^{\text{meas.}}_{m,n}[k]\right)}{T}}, \tag{19}$$

for a pixel (m, n).

We average over all available waveforms on the different diffuse targets and ablate different parts of the forward model to validate its effectiveness. In clear conditions, using the full low-flux transient model with the measured pulse g yields an RMSE of 0.768, while the generic pulse of [8] degrades accuracy to 0.987 RMSE (+28.5%). To evaluate the scattering model, we capture the calibrated reflection targets in fog with 80 m visibility. We fit Eq. 17 and a fog thickness $\alpha^{\text{fog.}}$ to the selected waveforms. The full scattering model achieves an RMSE of 1.843. Omitting the fog simulation altogether reduces simulation accuracy to 3.370 RMSE (+82.8%), while omitting only the scattering peak f^{fog} yields 3.084 RMSE (+67.3%). The qualitative findings for the 50% reflection target in Fig. 8 confirm the accuracy of the scattering model. Lacking calibrated retroreflective targets, we validate the high-flux model on selected waveforms from traffic signs instead. Quantitatively, omitting the secondary peak f^{sec} reduces accuracy to 4.025 RMSE (+245.5%), while no multipath peak yields an RMSE of 1.230 (+5.6%) versus our full high-flux model (1.165 RMSE).

2.5. CARLA Simulation

A sufficient amount of training data is required to use the FWL in a learning-based framework. We thus integrate our forward model into CARLA [4] by extending the full waveform lidar model as presented in [8, 19]. As shown in Fig. 2 (main paper), we extract scene geometry $d_{i,j}$, reflectivity $\rho_{i,j}$ and ambient light $a_{i,j}$ in a supersampled fashion with $a = 3$ and downsample to model multipath transients. To allow for a realistic simulation, we use the measured pulse of the real sensor for simulation and adjust the spatial and temporal resolution to match the real one.

To allow our neural DSP to generalize to a wide range of effects, we vary the previously described parameters for high-flux and foggy transients during simulation. We list the distribution used for simulating high-flux and foggy transients in Tab. 1.

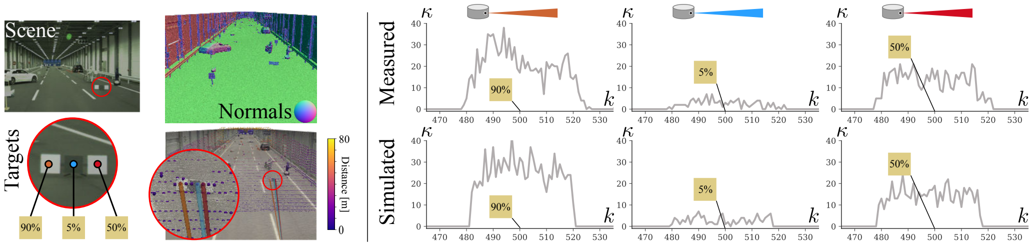


Figure 9. **Validation of Forward Model with Diffusive Targets**. We compare simulated and measured waveforms from three different diffuse reflection targets by inserting reflectivity, normals and distance into the forward model. Qualitatively, we observe a good correspondence between measurements and simulation.

	Name	Symbol	Distribution
Primary	Amplitude	α^{prim}	$\mathcal{N}(270, 0)$
	Offset	μ^{prim}	$\mathcal{N}(20, 0.1)$
	Width	σ^{prim}	$\mathcal{N}(0.8, 0.1)$
Secondary	Height	h^{sec}	$\mathcal{N}(100, 40)$
	Offset	$\mu^{\text{sec.}}$	$\mathcal{N}(20, 0.5)$
	Width	σ^{sec}	$\mathcal{N}(3, 1)$
	Steepness	τ^{sec}	$\mathcal{N}(10, 2)$
Bloom	Blooming magnitude	B	$\mathcal{N}(100, 10)$
	Drop-off rate	α^{bloom}	$\mathcal{N}(3, 1)$
Fog	Exponential decay	α^{fog}	$\mathcal{N}(0.05, 0.01)$
	Bias	β^{fog}	$\mathcal{N}(0.03, 0.01)$
	Initial scattering peak	θ^{fog}	see Fig. 8

Table 1. **Simulation Parameters.** To allow our neural DSP to generalize to a wide range of effects, we vary the aforementioned parameters during simulation and augmentation. When no clear distribution emerges, we sample random parameters from a normal distribution $\mathcal{N}$ with a certain mean and standard deviation.

3. Neural DSP

We provide architectural details of the proposed neural DSP in Sec. 3.1, discuss the extension for super-resolution in Sec. 3.2, provide training details in Sec. 3.3, and discuss architectural hyperparameters in Sec. 3.4.

3.1. Architectural Details

In a nutshell, our neural DSP divides waveforms into patches and then enables cross-communication between patches to facilitate learning. A detailed overview of the neural DSP's architecture is provided in Tab. 2. We utilize spatio-temporal transformers to capture temporal and local spatial context, and leverage a U-Net structure with down- and upsampling layers to enable global context.

Spatio-Temporal Transformers We use spatio-temporal transformers as an elementary building block for our neural DSP. As described in the main paper, the spatio-temporal transformer decomposes attention into spatial and temporal attention. To this end, we split features over n_{heads} to compute the attention over. The parameter n_{depth} specifies the number of times the transformer block in Fig. 3 of the main paper is repeated. In consecutive repetitions, the attention windows for computing spatial attention are shifted to allow cross-window communication following the default SWIN approach [16]. The MLP features a single hidden layer that extends the feature dimension by a factor n_{MLP}. We use a GELU activation function as non-linearity in the MLP.

Downsampling Downsampling is performed using patch merging layers. They first concatenate features from a $H \times W \times \widetilde{T} \times D$ feature map from 2x2 neighborhoods to a $H/2 \times W/2 \times \widetilde{T} \times 4D$ feature map. Then, a linear projection layer is applied to yield feature maps of size $H/2 \times W/2 \times \widetilde{T} \times 2D$ followed by a final layer normalization.

Upsampling Patch expanding is used for upsampling. To this end, spatially downsampled features are first projected to 2 times the feature resolution and then rearranged to undo the downsampling in the spatial dimension. As shown for the decoder blocks in Tab. 2, features from residual connections are then concatenated and subsequently processed with a linear projection to restore the original feature map resolution.

Name	Layer setting	Output dimension
Learned Matched Filter	Temporal Conv (Padded) $[1 \times 1 \times 39, 1]$	$H \times W \times T \times 1$
Tokenization	Rearrange $H \times W \times T \times 1 \longrightarrow H \times W \times \widetilde{T} \times T/\widetilde{T}$ Layer Norm Linear Projection $[T/\widetilde{T}, D]$ Layer Norm	$H \times W \times \widetilde{T} \times D$
Encoder 1	Spatio-Temporal-Transformer $[2, 2, 2]$ Patch Merging	$\frac{1}{2}H \times \frac{1}{2}W \times \widetilde{T} \times 2D$
Encoder 2	Spatio-Temporal-Transformer $[2, 2, 2]$ Patch Merging	$\frac{1}{4}H \times \frac{1}{4}W \times \widetilde{T} \times 4D$
Bottleneck	Spatio-Temporal-Transformer $[2, 2, 2]$	$\frac{1}{4}H \times \frac{1}{4}W \times \widetilde{T} \times 4D$
Decoder 2	Patch Expanding Concat [Encoder 2 outputs, Patch Expanding outputs] Linear Projection $[4D, 2D]$ Spatio-Temporal-Transformer $[2, 2, 2]$	$\frac{1}{2}H \times \frac{1}{2}W \times \widetilde{T} \times 2D$
Decoder 1	Patch Expanding Concat [Encoder 1 outputs, Patch Expanding outputs] Linear Projection $[2D, D]$ Spatio-Temporal-Transformer $[2, 2, 2]$	$H \times W \times \widetilde{T} \times D$
Offset Head	Linear Projection $[D, 1]$ Simoid	$H \times W \times \widetilde{T} \times 1$
Classification Head	Linear Projection $[D, 2]$ Softmax	$H \times W \times \widetilde{T} \times 2$

Table 2. **Neural DSP Architecture.** The "Concat" operation indicates that we concatenate elements along the feature dimension. Spatio-Temporal Transformers are parameterized by $[n_{\text{depth}}, n_{\text{heads}}, n_{\text{MLP}}]$.

3.2. Super-Resolution Neural DSP

To use our neural DSP to render super-resolution point clouds, we use another spatio-temporal transformer block that can be added to the output of the Decoder 2 defined in Tab. 2. As shown in Tab. 3, we first expand features by projecting with a factor a^2, then rearrange the features to a super-resolution feature map of size $aH \times aW \times \widetilde{T} \times D$ and then process the features in another spatio-temporal transformer block. The offset and classification heads as defined in Tab. 2 can then be used to render the super-resolution point clouds.

Name	Layer setting	Output dimension
Expansion	Linear Projection$[D, a^2 D]$	$H \times W \times \widetilde{T} \times a^2 D$
Rearrangement	Rearrange $H \times W \times \widetilde{T} \times a^2 D \longrightarrow aH \times aW \times \widetilde{T} \times D$	$aH \times aW \times \widetilde{T} \times D$
Norm	Layer Norm	$aH \times aW \times \widetilde{T} \times D$
Feature Processing	Spatio-Temporal Transformer [2, 2, 1]	$aH \times aW \times \widetilde{T} \times D$

Table 3. **Neural DSP Extension for Super-Resolution.** This super-resolution block can be added to the neural DSP described in tab. 2 to render super-resolution feature maps with an upscaling factor a. Spatio-Temporal Transformers are parameterized by $[n_{\text{depth}}, n_{\text{heads}}, n_{\text{MLP}}]$.

3.3. Training and Implementation Details

We implement the neural DSP in PyTorch and train with the Adam optimizer with a constant learning rate of 3×10^{-4}. The model is trained for 80 epochs on 4 Nvidia V100 GPUs with a batch size of 2 on synthetic CARLA data. We then finetune for another 80 epochs on the captured real-world dataset. We use Eq. (10) and Eq. (14) to augment additional traffic signs, as well as Eq. (18) for fog into the real data. We emphasize that the neural DSP has never seen any real fog data during training. As described in Tab. 1, parameters for retroreflector and fog augmentation are varied randomly.

As discussed in the main paper, we supervise both the output of the offset and the classification head. The overall loss $\mathcal{L}$ is given by

$$\mathcal{L} = \lambda_{\text{offset}}\mathcal{L}_{\text{offset}} + \lambda_{\text{class}}\mathcal{L}_{\text{class}}, \tag{20}$$

where $\lambda_{\text{offset}} = 0.1$ and $\lambda_{\text{class}} = 1.0$ are weighting factors. For the extension to render super-resolution point clouds, we apply offset and classification heads to both the super-resolution and low-resolution feature maps and supervise both with the loss defined in Eq. (20), yielding the overall loss $\mathcal{L}'$

$$\mathcal{L}' = \lambda_{\text{SR}}\mathcal{L}_{\text{SR}} + \lambda_{\text{LR}}\mathcal{L}_{\text{LR}}, \tag{21}$$

where $\lambda_{\text{SR}} = 1.0$ and $\lambda_{\text{LR}} = 0.1$ are weighting factors.

3.4. Hyperparameters

Following standard practice, we tune the hyperparameters of the neural DSP on the synthetic validation set. Using Recall as metric, the selected hyperparameters ($\widetilde{T} = 33, D = 32,$ window 2x4) achieve 82.04%. Increasing the number of patches to $\widetilde{T} = 66$ reduces Recall to 80.63% (-1.41%), whereas $\widetilde{T} = 16$ yields 81.05% (-0.99%) Recall. Altering the feature dimension to $D{=}16$ yields 79.54% (-2.50%) and $D = 64$ returns 77.89% (-4.15%). Varying the SWIN attention window size to 1x4 yields 79.83% (-2.21%) Recall, whereas 2x8 windows return 79.57% (-2.47%).

4. Additional Results

In the following, we provide a detailed analysis in support of the results presented in the main paper. We provide additional details about the baseline DSPs in Sec. 4.1. We present details about SNR calculation and the used metrics in general in Sec. 4.2. Furthermore, a maximum range metric is defined in Sec. 4.3. Additional qualitative results can be found in Sec. 4.4. Ablations of different layers and inputs of our neural DSP are introduced in Sec. 4.5. Finally, we discuss qualitative super-resolution results in Sec. 4.6 and details about the comparisons with commercial lidar sensors in Sec. 4.7.

4.1. Baseline DSPs

Conventional (Conv.) Peakfinding We compare to a conventional on-device peak-finding baseline as shown in Fig. 10. To this end, we process waveforms κ for each ray individually. First, we employ matched filtering by convolving the waveform κ for an individual pixel with the measured emitted laser pulse g, yielding the filtered waveform κ_{filt}. Following [8], we then perform ambient light subtraction by subtracting the noise floor approximated with the median, such that

$$\kappa'_{\text{filt}} = \kappa_{\text{filt}} - \text{median}(\kappa_{\text{filt}}). \tag{22}$$

We then identify local maxima in κ'_{filt} by simple comparison of neighboring values and enforcing a distance threshold d_{thresh} that defines a minimum required distance between peaks. This ensures that only a single distance is estimated per peak in the case of noisy peaks. The time bin of each local maxima can then be converted to a distance, and we keep up to 4 peaks per waveform in the distance vector $\mathbf{d}$. The height of the peak - the intensity - is denoted with $\mathbf{i}$. We then threshold the intensity with the threshold i_{thresh} and only keep distances $\mathbf{d}$ that have an intensity value above this threshold. As a result, the conventional DSP can suppress points in sky regions. Lastly, we threshold the distances with a minimum distance d_{min} to yield the final estimated distances $\mathbf{d}''$. This is required since the waveform κ includes a peak caused by backscatter from the front cover before the emitted pulse can leave the sensor.

In foggy conditions, we choose a higher minimum distance d_{min} to suppress the initial scattering peak better, choose a lower intensity threshold i_{thresh}, and only consider the last peak of $\mathbf{d}$ – with respect to the distance – that is found.

Baseline Transient Imaging Methods As described in the main paper manuscript, the transient imaging methods by Lindell *et al.* [15] and Peng *et al.* [18] struggle in sky regions without points, as in order to suppress the sky points, they would need to predict a waveform with a single peak at zero distance. Furthermore, the KL divergence loss employed by these methods provides no supervision in these sky regions. We thus extend these methods to predict a sky mask and supervise the denoising process only on occupied pixels. To predict the sky mask, we employ a lightweight convolutional network that operates along the temporal dimension of the last feature map. As illustrated by Tab. 4, we perform three downsampling steps along the temporal axis and then predict a classification score of whether a pixel belongs to the sky or an object. We supervise this classification score with a binary cross-entropy loss.

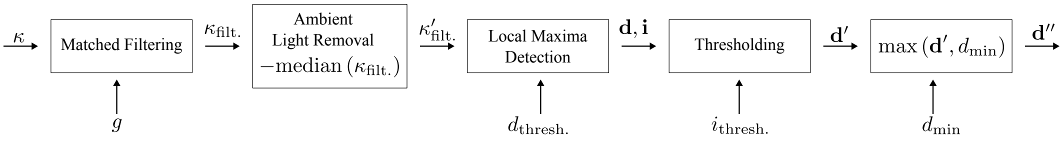


Figure 10. **Conventional (Conv.) Peak Finding.** Typical on-device DSPs process waveforms κ individually. As these DSPs are proprietary, we mimic conventional peak-finding by performing matched filtering, ambient noise removal, local maxima identification, and intensity and distance-based thresholding.

Name	Layer setting	Output dimension
Temporal Downsampling 1	Conv (padding) $[(D, 32), 9, 4]$ ReLU Max Pooling$[8, 8]$	$H \times W \times 264 \times 32$
Temporal Downsampling 2	Conv (padding) $[(32, 64), 9, 4]$ ReLU Max Pooling$[8, 8]$	$H \times W \times 33 \times 64$
Temporal Downsampling 3	Conv (padding) $[(64, 128), 9, 4)]$ ReLU Max Pooling$[7, 7]$	$H \times W \times 5 \times 128$
Classification Head	Flatten Linear Layer$[128 \times 5, 2]$ Softmax	$H \times W \times 2$

Table 4. **Sky-mask Extension.** "Conv" denotes a one-dimensional convolution along the temporal axis defined by [(number of input channels, number of output channels), kernel size, padding], and "Max Pooling" denotes max pooling along the temporal axis defined by [kernel size, stride].

4.2. Evaluation Metrics

As the neural DSP can predict multiple distances per pixel and our ground truth is also multi-echoed, we evaluate our method on the point cloud level. To this end, we compare the predicted point clouds $\mathbf{O}_{\text{pred}}$ to the projected ground truth point cloud $\mathbf{O}_{\text{GT}}$ as discussed in Sec. 1.2.

Chamfer Distance We evaluate using the Chamfer Distance (CD) given by

$$\text{CD}(\mathbf{O}_{\text{pred}}, \mathbf{O}_{\text{GT}}) = \frac{1}{n_{\text{pred}}} \sum_{i}^{n_{\text{pred}}} \|\mathbf{o}_i - \text{NN}(\mathbf{o}_i, \mathbf{O}_{\text{GT}})\| + \frac{1}{n_{\text{GT}}} \sum_{j}^{n_{\text{GT}}} \|\mathbf{o}_j - \text{NN}(\mathbf{o}_j, \mathbf{O}_{\text{pred}})\|, \tag{23}$$

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

$$\text{NN}(\mathbf{o}, \mathbf{O}') = \text{argmin}_{\mathbf{o}' \in \mathbf{O}'} \|\mathbf{o} - \mathbf{o}'\|, \tag{24}$$

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

Recall We further compare the recall of the predicted point cloud, denoting the ratio of the scene that is correctly reconstructed. To this end, points in the predicted point cloud are considered true positive (TP) if

$$\text{TP} := \text{NN}(\mathbf{o}_{\text{pred}}, \mathbf{O}_{\text{GT}}) < d_{\text{true}}, \tag{25}$$

where $d_{\text{true}} = 39.87$cm corresponding to 10 temporal bins. We identify false negative (FN) points by evaluating whether a ground truth point has a predicted point in its neighborhood, such that

$$\text{FN} := \text{NN}(\mathbf{o}_{\text{GT}}, \mathbf{O}_{\text{pred}}) >= d_{\text{true}}. \tag{26}$$

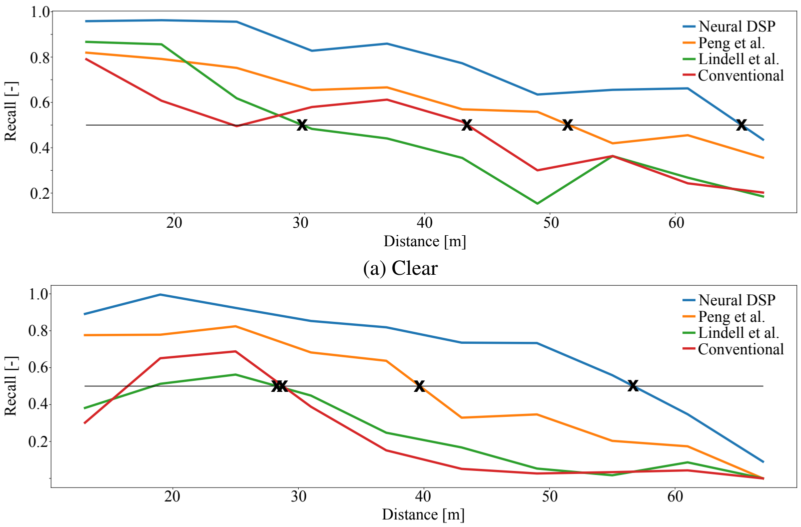
SNR Calculation We evaluate CD and Recall on different signal-to-noise ratio (SNR) bins, showcasing the benefit of our neural DSP in low-SNR conditions. We calculate the signal by querying the waveform κ at the time bin that corresponds to the distance of a ground-truth point. We approximate the noise by taking the median along the waveform κ, allowing us to formulate SNR as

$$\text{SNR}(\mathbf{o}_{\text{GT}}) = \frac{\kappa\left[\|\mathbf{o}_{\text{GT}}\|\right]}{\text{median}(\kappa)}, \tag{27}$$

where [...] denotes the indexing operation along the waveform κ.

4.3. Maximum Range Metric

Inspired by commercial sensors, we introduce a more intuitive maximum range metric on low-reflective targets. For example, Velodyne specifies its maximum range on target with 10% reflectivity.[22] We use points of the low-SNR bin (0–2) as low-reflective targets and sort predicted points into distance bins. We calculate the distance-binned Recall and define maximum range as the farthest distance at which 50% of Recall is still achieved.



(a) Clear

(b) Fog

Figure 11. **Maximum Range Metric.** Maximum range is calculated as the farthest distance where 50% Recall for low intensity points is achieved. Our proposed neural DSP compares favorably to all baseline methods in both a) clear and b) foggy conditions.

More formally, we define the low-reflectivity point clouds $\mathbf{O}_{\text{GT}}^{\text{low}}$ and $\mathbf{O}_{\text{pred}}^{\text{low}}$ by filtering with the SNR score from Eq. (27). Then, we extract a subset $\mathcal{B}_b$ of points that fall into a distance bin b, such that

$$\mathcal{B}_i = \left\{\mathbf{o} \in \mathbf{O}_{\text{pred}}^{\text{low}} \mid \|\mathbf{o}\|_2 \in [d_b, d_{b+1})\right\}, \quad d_b = 7(i-1), \quad b = 1, \ldots, 10. \tag{28}$$

We define 10 distance bins covering the measurement range of the FWL and calculate the distance-binned Recall as

$$\text{Recall}_b = \frac{|\text{TP} \cap \mathcal{B}_b|}{|\text{TP} \cap \mathcal{B}_b| + |\text{FN} \cap \mathcal{B}_b|}, \quad b = 1, \ldots, 10 \tag{29}$$

Then, we search for the rightmost intersection of the distance-binned Recall to extract the maximum range, as indicated by Fig. 11.

As indicated by Fig. 11, our proposed neural DSP performs favorably in both clear and foggy conditions, extending the maximum range by 15m over the best-performing baseline. This emphasized again the benefit of our method in low-SNR conditions.

4.4. Additional Qualitative Results

We provide additional qualitative results in clear conditions in Fig. 12. As shown in the first two rows, our neural DSP is capable of reconstructing both high-frequency details and achieving an overall high-quality reconstruction performance, despite the higher ambient light conditions in outdoor scenarios. The related transient imaging methods by Lindell *et al.* and Peng *et al.* reconstruct the low-frequency details of the scene but struggle with finer details. The benefit of our neural DSP is

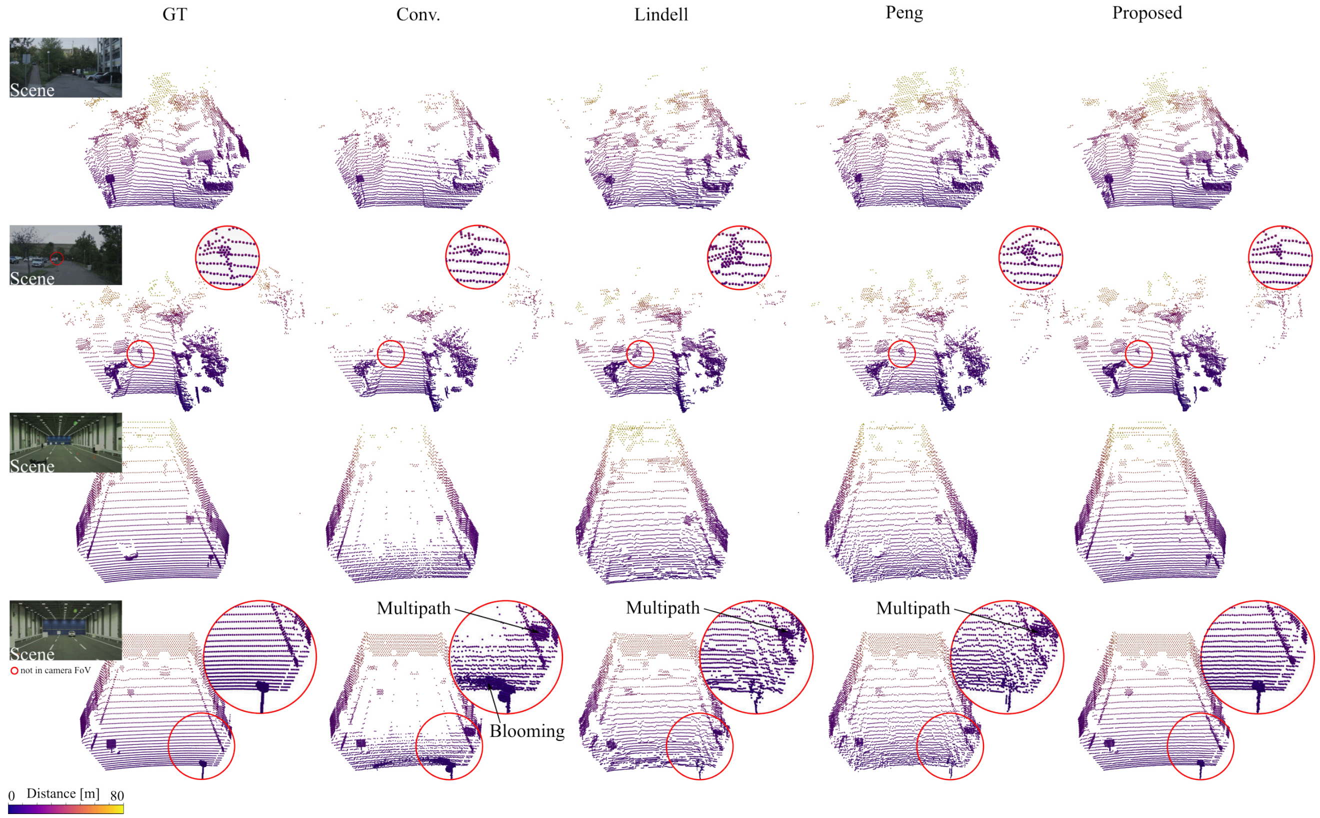


Figure 12. **Additional Results on the Clear Test Set.** Our neural DSP enables high-quality reconstruction results on both outdoor data (first two rows) and controlled environment data (bottom two rows). The conventional baseline misses low-SNR objects, whereas the transient imaging methods are unable to restore high-frequency details. The benefit of our Neural DSP in reducing blooming is especially visible in the bottom row for the traffic sign on the right in close proximity to the sensor.

especially pronounced when encountering retro-reflective materials, as seen in the last row for the traffic sign on the right, which is positioned close to the sensor. All baseline methods suffer from severe blooming and multipath effects, whereas our neural DSP can suppress these distortions.

Furthermore, we provide additional qualitative results in foggy conditions in Fig. 13. We find that our neural DSP consistently suppresses the false initial scattering peaks and recovers large parts of the scene, as seen in the second row, where our neural DSP is the only method to fully reconstruct the back wall of the weather chamber. In comparison to the clear condition in e.g. Fig. 12, it is noticeable that fewer points on e.g. the ground at further distances are not reconstructed. When inspecting these areas in the waveform, it can be seen that due to the exponential decay in fog, no signal remains to allow for any reconstruction. Hence, this is not related to the reconstruction method itself, but rather depends on the laser power emitted by the sensor. Increasing the laser power would also enhance reconstruction in these areas.

4.5. Neural DSP Ablation Experiments

To assess the effectiveness of our neural DSP, we ablate different modules of the neural DSP in Tab. 5. The neural DSP, utilizing all modules, achieves the best overall reconstruction results in terms of CD and Recall. When ablating the learned matched filter, the reconstruction performance of the neural DSP deteriorates, resulting in an increase of over 18 cm CD while achieving similar Recall. This can be attributed to the fact that the matched filter increases SNR, which subsequently allows for the reconstruction of attenuated peaks, especially at far distances. As a result, no nearest neighbor near a far-away ground truth point can be found, causing a significant increase in Chamfer distance, see second term of Eq. (23). However, as the point density of lidar sensors decreases quadratically with distance, Recall is affected to a greater extent by points in closer proximity to the sensor, thus suffering less from missed points at farther distances.

Next, we ablate both temporal and spatial attention in the spatio-temporal transformers. This prevents the neural DSP from capturing local and spatial context. As shown in Table 5, this results in an additional 13cm increase in CD. The decrease in

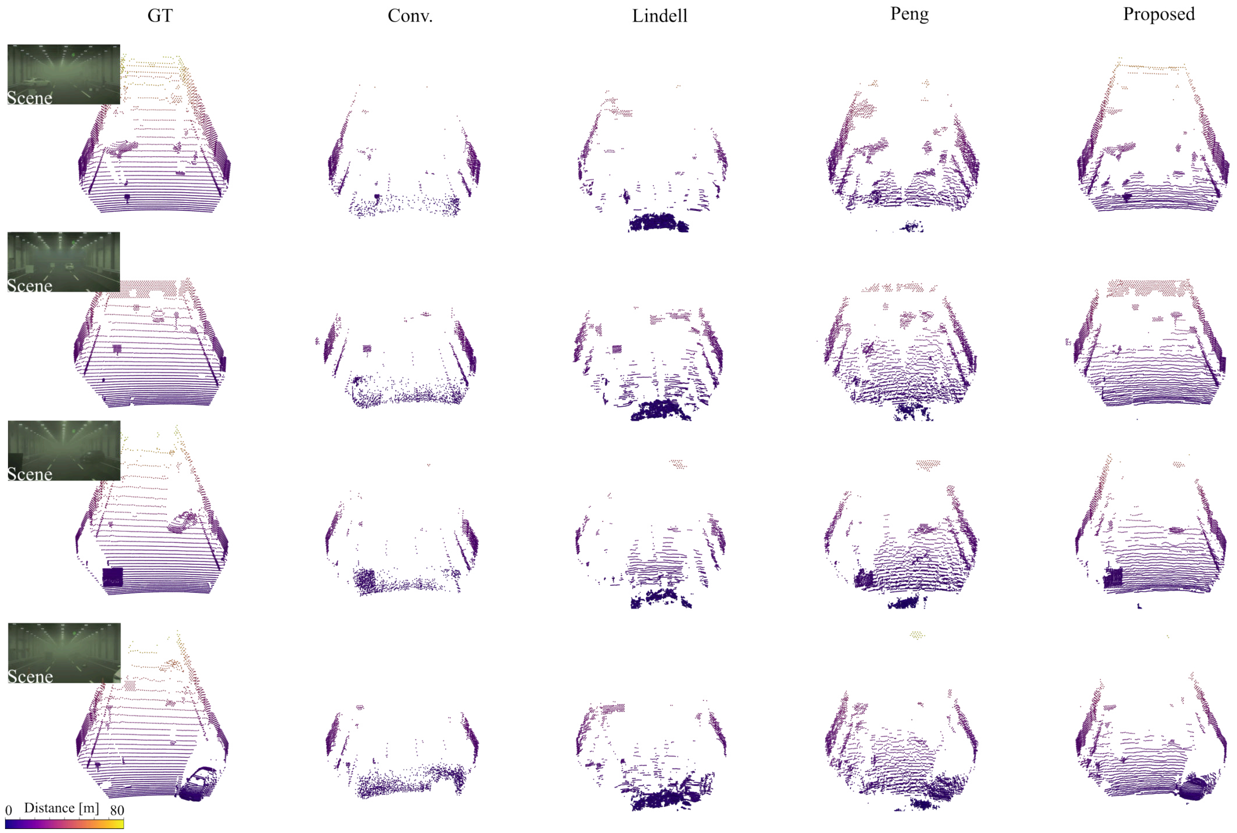


Figure 13. **Additional Results on the Foggy Test Set.** Our neural DSP is the only method to provide satisfying reconstruction results in foggy conditions. The baseline methods are either unable to reconstruct the attenuated object peaks altogether or struggle with suppressing false positive peaks from backscattered photons.

Recall by over 3 percentage points underlines the benefit of the temporal attention computation since, without patches, they cannot communicate along the temporal attention. Conversely, patches can still communicate along the spatial dimension without the spatial attention due to the U-Net structure.

Finally, we ablate the U-Net Structure and only process waveforms with a single spatio-temporal transformer. This prevents the neural DSP from capturing global context, causing the largest decrease in performance in both CD and Recall, as shown in Table 5.

Matched Filter	Temporal Att.	Spatial Att.	U-Net	CD [m] ↓	Recall [%] ↑
✗	✓	✓	✗	0.627	82.51
✗	✗	✓	✓	0.614	83.87
✗	✓	✗	✓	0.617	84.69
✗	✓	✓	✓	0.480	86.79
✓	✓	✓	✓	**0.397**	**87.79**

Table 5. **Ablation Studies for Different Modules on Synthetic Data.** The reconstruction performance deteriorates on both metrics when individual modules are ablated.

4.6. Additional Super-Resolution Results

We provide additional qualitative super-resolution results in Fig. 14 with an upsampling factor $a = 3$. Both baseline methods, Tuilp [24] and ILN [14], take the conventionally processed point cloud projected to a range image as input. The conventionally processed point cloud tends to be noisy, especially in outdoor conditions (see the first two rows). Both super-resolution

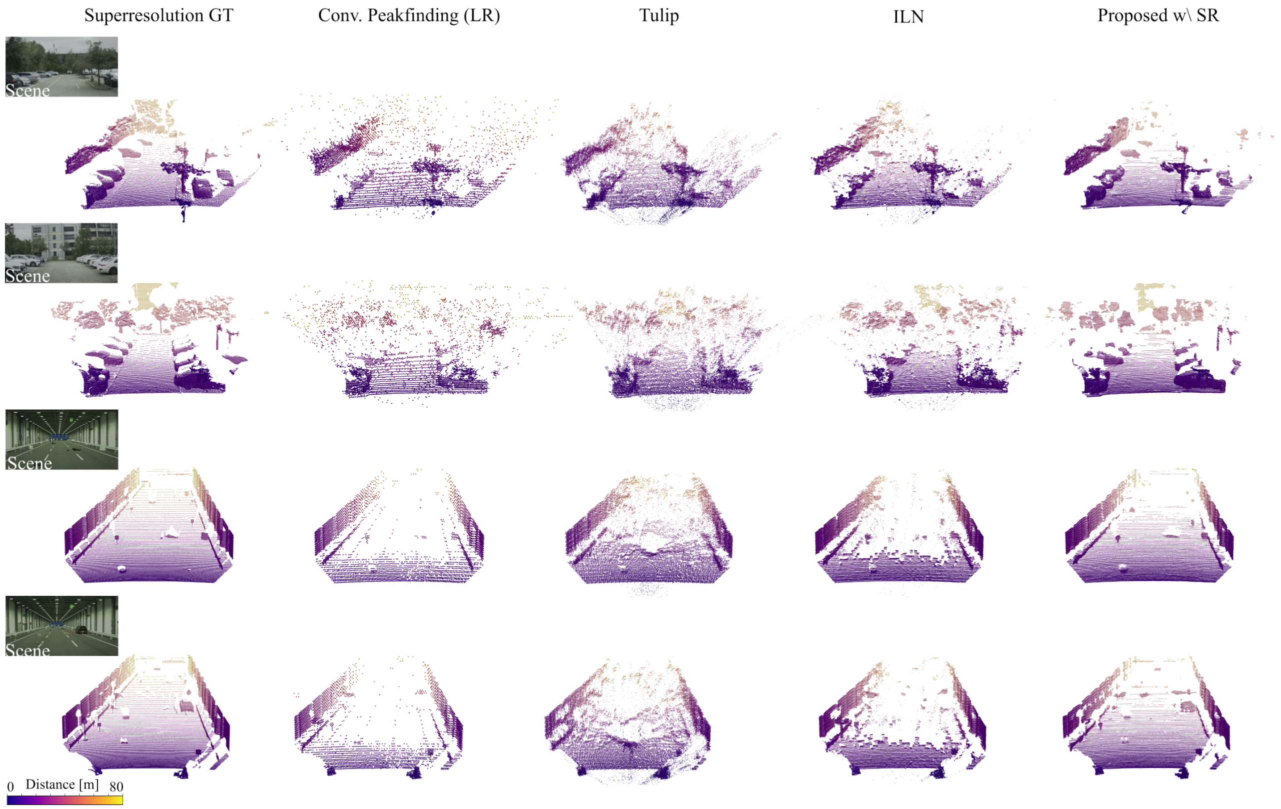


Figure 14. **Additional Super-Resolution Results.** Both baseline methods Tulip [24] and ILN [14] ingest the conventional low-resolution (LR) point cloud. Since this point cloud tends to be noisy in outdoor conditions (first two rows), the baseline super-resolution methods propagate this noise. Our neural DSP, however, is able to render super-resolution point clouds even in challenging outdoor conditions and proves to be effective for detecting small obstacles on the ground (bottom two rows).

baseline methods propagate this noise, and the super-resolution point cloud suffers from flying pixels. This can be attributed to the fact that existing lidar super-resolution methods are designed for simulated point clouds or sensors used in ground truthing applications, which output essentially noiseless point clouds. However, automotive lidar sensors for real-world applications must be cost-effective and typically exhibit more noise points than ground-truthing sensors. In contrast to the baselines, our proposed neural super-resolution DSP enables the rendering of super-resolution points without flying pixels. As shown in the bottom two rows, super-resolution is especially helpful for detecting small obstacles on the ground.

In contrast to baseline methods, our neural DSP can leverage superresolution cues present in the waveform. These superresolution cues contain implicit information about the underlying scene geometry, which in turn allows our neural DSP to extract higher-resolution point clouds. One superresolution cue is the number of peaks or echoes present in the waveform. Multiple echoes from each ray indicate discontinuities, and a higher-resolution sensor would resolve them into distinct surfaces. In contrast, single echoes indicate a single surface. Furthermore, the width of the returning pulse reveals information about the surface normal. Widened pulses indicate upright normals, whereas surfaces with normals facing the sensor produce a narrower return. The neural DSP can leverage surface normal information for interpolating the effects of a finer scan pattern. The baseline methods do not have access to this information and can only extrapolate from their nearest neighbors, which limits their performance in long-range and low SNR conditions.

4.7. Commercial Lidar Comparison

We compare our neural DSP against the peak finding from an Aeva Aries II and a Luminar Iris. We also provide a comparison with the Velodyne VLS-128, which is used for ground truthing applications. In Fig. 15, we show qualitative comparisons of the point clouds output by the different sensors. As the FoV and resolution vary from sensor to sensor, we compare - as described in the main paper - point cloud distance accuracy to the dense ground truth scan $\mathbf{O}_{\mathrm{GT}}$. We define distance accuracy

(DA) as one-sided Chamfer Distance given by

$$\mathrm{DA}(\mathbf{O}_{\text{sensor}}, \mathbf{O}_{\text{GT}}) = \frac{1}{n_{\text{sensor}}} \sum_{i}^{n_{\text{sensor}}} \|\mathbf{o}_i - \mathrm{NN}(\mathbf{o}_i, \mathbf{O}_{\text{GT}}), \tag{30}$$

where $\mathbf{O}_{\text{sensor}} = \left\{\mathbf{o}_i \in \mathbb{R}^3\right\}_i^{n_{\text{sensor}}}$ defines the lidar point cloud output by the respective sensor. Due to the different ranges of the sensors, we evaluate only up to 80m range.

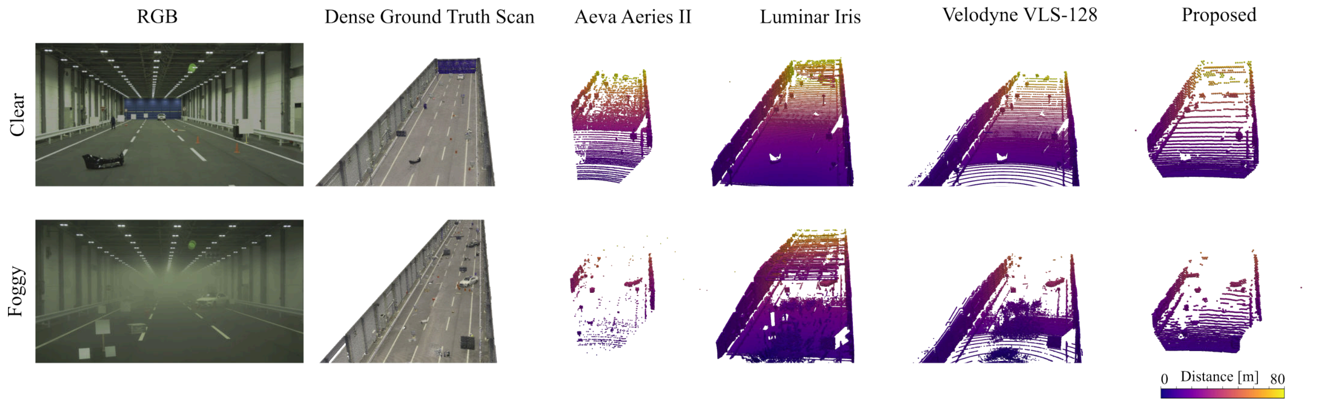


Figure 15. **Commercial Lidar Comparison.** We compare the peak finding of 3 commercially available lidar sensors to our proposed DSP. To this end, we compare against the dense ground truth scan in the weather chamber. We find that, especially in foggy conditions, our neural DSP improves scene reconstruction compared to all other sensors.

As discussed in the main paper, our neural DSP renders point clouds with higher distance accuracy compared to the other two commercially mass-producible lidars, surpassing only the Velodyne. This is supported by the qualitative findings in the first row of Fig. 15 and the error maps in Fig. 16. The Aeva exhibits more noise due to its frequency-modulated continuous wave scanning principle. Furthermore, objects placed in the scene are often detected with low distance accuracy as shown in Fig. 16. The Luminar produces a high-quality point cloud at first glance. However, it suffers from sensor-internal calibration offsets acting as a scaling factor to the point cloud, which decreases distance accuracy. This can be observed, e.g. on the side walls in the error maps in Fig. 16 that indicate larger distance error with increasing distance.

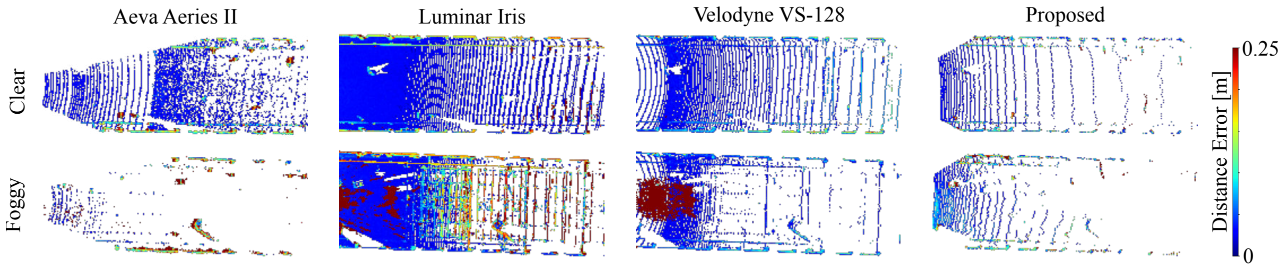


Figure 16. **Birds-Eye View Error Maps.** We compare point clouds for the scenes from Fig. 15 in a bird's-eye view where points are colored with the respective distance accuracy. The improved accuracy in fog is clearly visible. However, the benefit of our neural DSP is also visible in clear conditions; the proposed DSP mitigates large distance errors compared to Aeva and Luminar.

The benefit of our neural DSP becomes evident in foggy conditions, where the Luminar and Velodyne point clouds suffer from a severe amount of scatter points. Our neural DSP is able to suppress those, which leads to more accurate point clouds even when compared to the ground-truthing Velodyne sensor, see especially Fig 16 (second row). The Aeva's DSP can suppress backscatter, but only at the cost of providing an unsatisfactory scene reconstruction.

5. Multipath and Implementation Constraints

Our imaging model only simulates multipath effects as multiple reflections between the sensor and the target. However, another prominent multipath effect is the reflection from the ground to a target and back to the sensor. This effect is especially pronounced on wet roads, as the ground acts like a mirror-like surface. This is not modeled in the current iteration of the imaging model and, therefore, is likely not learned by our neural DSP. However, with accurate simulation, we estimate that our neural DSP's classification approach, along with its global context, would also allow the suppression of these false multipath peaks. As wet roads can occur up to 100 days per year [20], future work should simulate this effect, as it would further improve robustness in adverse weather conditions.

As described in Sec 1.1.1, the streaming of waveform data through the debug interface reduces the frame rate of the sensor to 0.5 Hz, causing severe motion blur for dynamic objects or in highly dynamic driving scenarios. To apply our neural DSP in an autonomous vehicle, it should either be run on-sensor, or a higher-bandwidth data transfer needs to be implemented. For on-device operation, we estimate that a Nvidia Jetson Nano DSP would allow real-time operation, as the current unoptimized version of the network already runs at 68 Hz with 400 MB of VRAM. However, data can also be streamed using existing standards, as the amount of waveform data is similar to that of a 10MP camera, which is already in use in current autonomous vehicles. The amount of data could be reduced by performing matched filtering and tokenization on-sensor and outsourcing the computationally heavier operations, e.g., the spatio-temporal transformer blocks, on a central compute unit. This would be advantageous, as the waveform features could be leveraged by, for example, a downstream object detector, allowing for end-to-end lidar perception. Future work could also rely on self-supervised pre-training, for instance, by reconstructing masked tokens from neighboring patches, as presented in e.g. [7] for timeseries data, to eliminate the need for ground-truth distance scans for supervision.

References

[1] Robert L. Cook and Kenneth E. Torrance. A reflectance model for computer graphics. *ACM SIGGRAPH Computer Graphics*, 15(3): 307–316, 1981. 5

[2] S. Cova, M. Ghioni, A. Lacaita, C. Samori, and F. Zappa. Avalanche photodiodes and quenching circuits for single-photon detection. *Appl. Opt.*, 35(12):1956–1976, 1996. 2

[3] Sergio Cova, Massimo Ghioni, Mark A. Itzler, Joshua C. Bienfang, and Alessandro Restelli. Chapter 4 - semiconductor-based detectors. In *Single-Photon Generation and Detection*, chapter Experimental Methods in the Physical Sciences, pages 83–146. Academic Press, 2013. 2

[4] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *1st Annual Conference on Robot Learning*, 2017. 9

[5] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, pages 226–231, 1996. 4

[6] Karl Christoph Goedel, Holger Maris Gilbergs, and Johannes Richter. Computer unit for a lidar device, and lidar device, 2023. US2023092128A1. 8

[7] Yuan Gong, Cheng-I Lai, Yu-An Chung, and James Glass. Ssast: Self-supervised audio spectrogram transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 10699–10709, 2022. 19

[8] Felix Goudreault, Dominik Scheuble, Mario Bijelic, Nicolas Robidoux, and Felix Heide. Lidar-in-the-loop hyperparameter optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13404–13414, 2023. 5, 9, 12

[9] Anant Gupta, Atul Ingle, Andreas Velten, and Mohit Gupta. Photon-flooded single-photon 3d cameras. In *Proc. CVPR*, 2019. 3

[10] Martin Hahner, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Fog simulation on real LiDAR point clouds for 3D object detection in adverse weather. In *IEEE International Conference on Computer Vision (ICCV)*, 2021. 8

[11] Felix Heide, Steven Diamond, David B Lindell, and Gordon Wetzstein. Sub-picosecond photon-efficient 3d imaging using single-photon sensors. *Scientific reports*, 8(1):1–8, 2018. 6

[12] Hanno Holzhüter. Method and device for optical distance measurement, 2024. US Patent 11,892,569. 8

[13] Sagi Katz, Ayellet Tal, and Ronen Basri. Direct visibility of point sets. In *ACM SIGGRAPH 2007 papers*, pages 24–es. 2007. 4

[14] Youngsun Kwon, Minhyuk Sung, and Sung-Eui Yoon. Implicit lidar network: Lidar super-resolution via interpolation weight prediction. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 8424–8430. IEEE, 2022. 16, 17

[15] David B. Lindell, Matthew O'Toole, and Gordon Wetzstein. Single-photon 3d imaging with deep sensor fusion. *ACM Trans. Graph.*, 37(4), 2018. 12

[16] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 10

[17] MicroVision. MOVIA™ L. `https://microvision.com/products/movia-l-automotive`, 2021. accessed: 2024-11-21. 2

[18] Jiayong Peng, Zhiwei Xiong, Hao Tan, Xin Huang, Zheng-Ping Li, and Feihu Xu. Boosting photon-efficient image reconstruction with a unified deep neural network. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):4180–4197, 2023. 12

[19] Dominik Scheuble, Chenyang Lei, Seung-Hwan Baek, Mario Bijelic, and Felix Heide. Polarization wavefront lidar: Learning large scene reconstruction from polarized wavefronts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21241–21250, 2024. 5, 9

[20] Dominik Scheuble, Clemens Linnhoff, Mario Bijelic, Lukas Elster, Philipp Rosenberger, Werner Ritter, and Hermann Winner. Simulating road spray effects in automotive lidar sensor models. In *2024 IEEE Intelligent Vehicles Symposium (IV)*, pages 659–666, 2024. 19

[21] Alessandro Tontini. *Advanced techniques for SPAD-based CMOS d-ToF systems*. PhD thesis, Università degli studi di Trento, 2024. 2

[22] Velodyne. Hdl-64e user's manual, 2007. 14

[23] Ignacio Vizzo, Tiziano Guadagnino, Benedikt Mersch, Louis Wiesmann, Jens Behley, and Cyrill Stachniss. 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): 1029–1036, 2023. 3

[24] Bin Yang, Patrick Pfreundschuh, Roland Siegwart, Marco Hutter, Peyman Moghadam, and Vaishakh Patil. Tulip: Transformer for upsampling of lidar point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15354–15364, 2024. 16, 17

[25] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018. 3, 4