Robust 3D Object Detection using Probabilistic Point Clouds from Single-Photon LiDARs

Supplementary Material

We provide additional details and results that complement the main paper in this report. We include more results with real captures (section 1), ablation studies (section 2), implementation details and additional results from our 3D detection benchmark (section 3), results with more 3D detectors (section 4), histogram denoising and compression methods (section 5) and raw histogram examples under low SBR (section 6).

1. Recognition with Real Captures

In this section, we provide further details about our LiDAR setups and more results on real PPC captures.

1.1. Camera Setups

We use a LiDAR sensor with an external laser for our indoor captures. This allows us to control various camera and scene parameters (*e.g.*, exposure time, laser power, and ambient illumination) over a wide range. We use a commercial LiDAR sensor for our outdoor captures. This allows us to have a portable low-power LiDAR setup for outdoor environments. Here are the details of both setups:

Indoor Camera Setup: Our indoor setup uses a SPAD-LiDAR sensor with an external class 4 laser. Fig. 1 shows the front view of our setup with the HORIBA FLIMera [5] camera. The temporal resolution of the camera is about 380ps, which is in line with the full-width at half-maximum (FWHM) of the instrument response function (IRF) of the device. We set up our camera system with the Katana laser [3], which is a high-powered pulsed picosecond laser system by OneFive. The laser has a wavelength of 532nm (green). We operate the laser under low power settings (ranging from 50-100mW). We use a flash illumination setup with a diffuser to illuminate the field of view of the sensor. We use a 3.8mm focal length lens for a wider field of view of the scene. The laser system, as well as the FLIMera sensor, is connected to an external computer to receive the synchronization signal and trigger for the capture.

Outdoor Camera Setup: Our outdoor camera setup uses a commercial LiDAR sensor, which is a more portable camera and uses low input power. Fig. 2 shows the front view of the setup with Adaps [9] camera. The camera is rated for an accuracy of more than 5cm up to a range of 30m. The setup has a wide FOV (120°horizontal and 90°vertical). We also vary the exposure time from 0.1s to 1s to simulate various signal levels. The camera can operate at a very low power input (<10W) and is connected to a small portable AC power source. The camera also saves low-resolution (256x192) grayscale images, which are used for visualizations only. We also mount a smartphone camera in our setup to simultaneously capture high-res RGB images, used for visualizations only for some static scenes shown in the main paper and supplementary report.

Figure 1. **Indoor Camera Setup:** The Figure shows the front view of our FLIMera camera setup (left) and the sensor (right).

Figure 2. **Outdoor Camera Setup:** The Figure shows the front view of our Adaps camera setup (left) and the sensor (right).

1.2. 3D Object Detection Results

Fig. 3 and 4 show a comparison of our approach with the baselines using real indoor and outdoor captures. Matched Filtering baseline suffers from noise and often detects false positives. Thresholding frequently misses small or farther objects in the scene. Baselines struggle with farther chairs in indoor captures and farther cars and pedestrians in outdoor captures. Our approach detects most objects with tight bounding boxes.

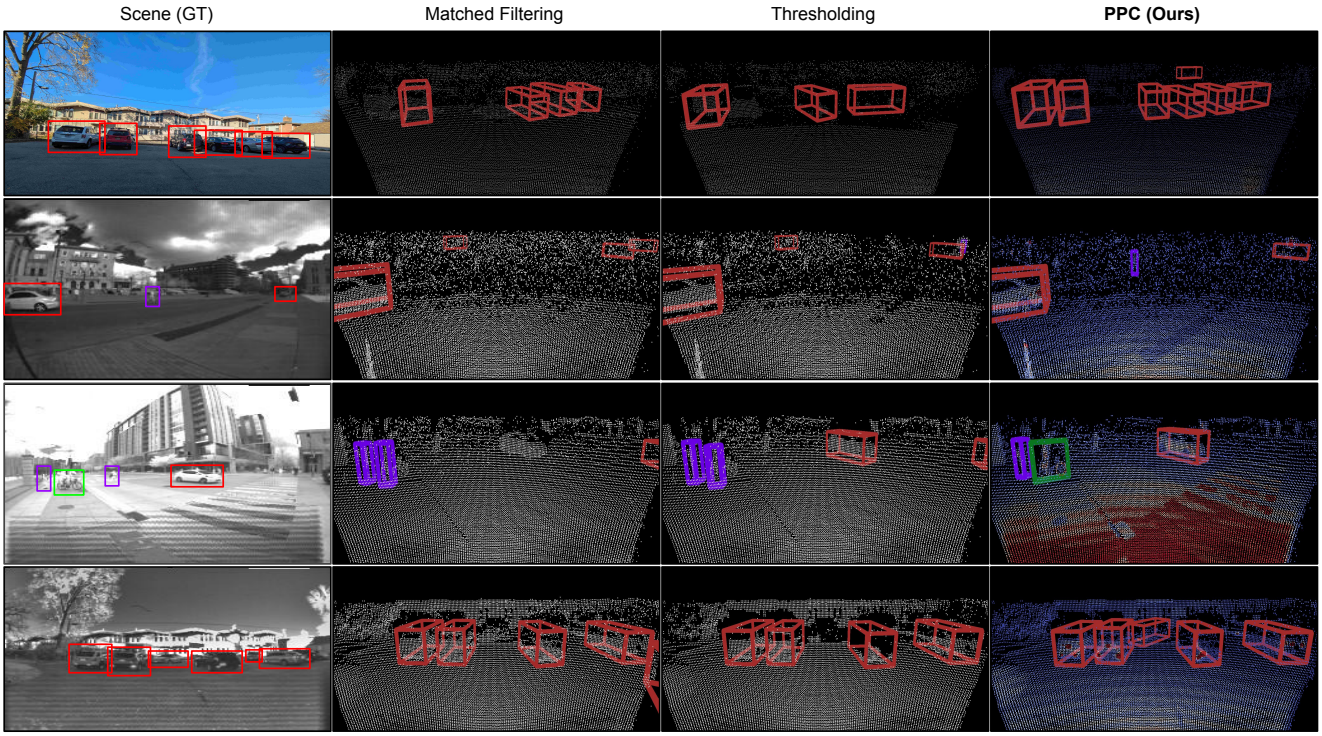


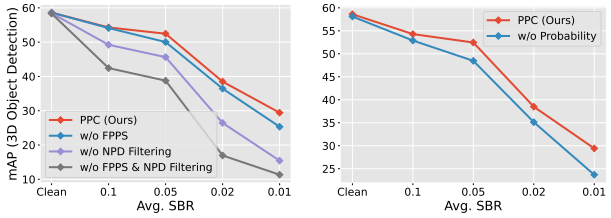
Figure 3. **3D Detection Results on Real Outdoor Captures:** Figure compares our method with the baselines under very challenging low SBR conditions. The first scene includes 6 cars, and baselines fail to detect farther cars. The second scene shows 2 cars and a pedestrian, and the baselines struggle to detect the distant pedestrian. The third scene includes a car, two pedestrians, and a cyclist. The ground truth objects are marked in the camera scene images for easier visualization. PPC detects most objects (*e.g.* cars, pedestrians, and cyclists) in all scenes with accurate bounding boxes.

Figure 4. **3D Detection Results on Real Indoor Captures:** Figure compares our method with the baselines under challenging low SBR conditions. Baselines fail to detect many small and farther objects, *e.g.* chairs in the back, in many scenes. PPC detects most objects (*e.g.* chair, table, and couch) with tight bounding boxes.

2. Ablation Studies

In this section, we include the ablation studies of our proposed method. We evaluate various design choices on the SUN RGB-D benchmark using the VoteNet architecture.

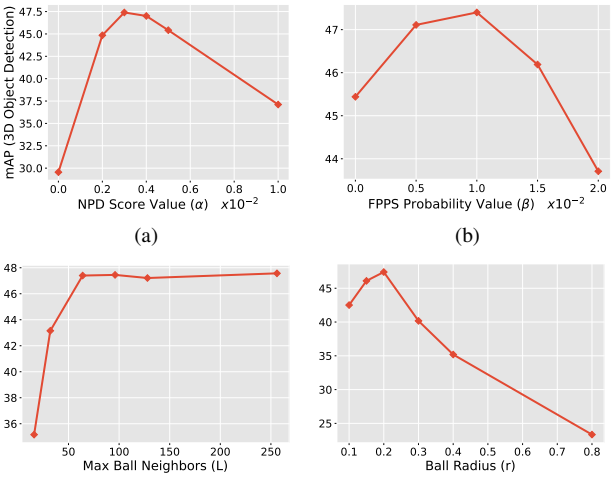
First, we analyze the effectiveness of both NPD Filtering and FPPS individually. Fig. 5a shows the results of our approach without NPD and FPPS. NPD filtering shows a significant gain in performance, especially under very low SBR scenarios. FPPS shows an additional 2-4% improvement in mAP under very low SBR conditions.



(a) (b)

Figure 5. **Ablation Study of PPC Components:** Performance of our approach (a) without FPPS and NPD Filtering, and (b) without probability attribute.

Second, we show the performance of our method on point clouds without the probability attribute in Fig. 5b. This is equivalent to using our approach with a conventional point cloud (*i.e.*, all points with probability 1). The probability attribute accounts for about a 4-5% gain in mAP performance and is significant in very low SBR conditions.



(a) (b)

(c) (d)

Figure 6. **Ablation Study of PPC Hyperparameters:** Performance of our approach with varying (a) NPD Score Value, (b) FPPS Value, (c) Max Ball Neighbors, and (d) Ball Radius.

We also show the performance of our approach by varying the hyperparameters of NPD filtering (α) and FPPS (β). Fig. 6a and 6b show the mAP on the complete SUN RGB-D test set of all SBR levels. We chose the best performing value of $\alpha = 0.003$ and $\beta = 0.01$ for our models.

We also analyze our method by varying the hyperparameters of NPD score calculation, *i.e.*, max ball neighbors (L) and ball radius (r). Fig. 6c and 6d show the mAP on the complete SUN RGB-D test set of all SBR levels. We find an optimal NPD score value (α) for each experiment. Increasing the radius too much starts to hurt the performance, as noisy sparse points have more neighbors if the ball radius is bigger. Performance improves as the value of L increases, but saturates around 64. We chose the values of $r = 0.2$ and $L = 64$ for our models.

Table 1 analyzes the total per-scene runtime of our method on the SUN RGB-D dataset. We use a single RTX 2070 Super GPU machine for inference time calculation. Adding FPPS adds no computational overhead, whereas adding NPD filtering adds less than 8% of runtime with our implementation.

	Inference Time (ms)
PPC	95
PPC w/o FPPS	95
PPC w/o NPD	88
PPC w/o FPPS & NPD	88

Table 1. **Ablation Study of Runtime:** Our method adds no significant computational cost to runtime.

We analyze the performance of the Thresholding baseline by varying the threshold used for the model. Fig. 7 shows AP@25 results on the complete SUN RGB-D test set of all SBR levels. We select the best performing threshold (=1.1) for evaluating this baseline.

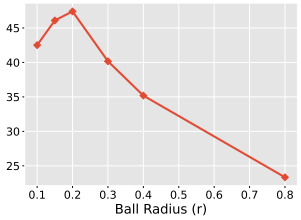


Figure 7. **Ablation Study of Thresholding baseline:** Performance of the Thresholding baseline with varying threshold used for the model.

Category	Matched Filtering	Thresholding	PointClean Net [12]	Score Denoise [7]	PathNet [16]	**PPC (Ours)**
Bed	54.37	67.33	53.59	68.57	67.53	**72.97**
Sofa	16.20	28.15	22.00	38.25	38.03	**45.19**
Table	29.61	37.99	30.20	33.93	32.17	**40.47**
Bathtub	6.04	25.14	2.71	14.04	13.46	**54.32**
Desk	7.97	14.16	7.78	8.33	9.18	**17.37**
Bookshelf	2.12	4.85	0.67	1.01	0.89	**9.80**
Chair	22.05	34.27	22.92	27.45	25.43	**47.47**
Night Stand	3.10	14.45	7.41	10.23	9.13	**30.49**
Dresser	2.58	4.64	2.80	2.44	2.09	**13.73**

Table 2. **Category-wise 3D Object Detection Results**: Table shows per category AP@25 results on the SUN RGB-D dataset under low SBR (0.02) conditions. Our approach outperforms all baselines and shows large gains for smaller object categories (below the line) like chairs and nightstands.

Avg. SBR	Car			Pedestrian			Cyclist		
	Easy	Mod	Hard	Easy	Mod	Hard	Easy	Mod	Hard
Matched Filtering	79.02	59.95	57.67	50.85	47.06	43.51	68.78	43.74	41.10
Thresholding	78.73	59.40	55.35	54.45	49.23	45.38	68.13	44.96	42.64
PPC (Ours)	**79.10**	**60.29**	**59.08**	**60.42**	**55.39**	**50.82**	**71.99**	**47.76**	**44.84**

Table 3. **KITTI 3D Detection Comparison**: Table shows mAP for easy, moderate, and hard difficulty levels on KITTI val split calculated with 11 recall positions for PV-RCNN architecture under low SBR (0.01) conditions.

3. Additional 3D Object Detection Results

In this section, we include implementation details, additional results, and analysis that supplement the experiments in the main paper.

3.1. Datasets

We evaluate our approach on 3D object detection benchmarks of SUN RGB-D [13] and KITTI [2]. SUN RGB-D consists of ∼10K RGB-D scans of *indoor* scenes annotated with 3D bounding boxes. The dataset also provides camera intrinsic and extrinsic parameters to convert depth scans to 3D point clouds. We use the standard evaluation protocol that considers 10 common object categories. KITTI dataset is a widely used *outdoor* autonomous driving dataset containing ∼7.4k annotated scenes with LiDAR point clouds. We follow the standard evaluation protocol using three categories: car, pedestrian, and cyclist.

3.2. Implementation Details

We implement our method using the MMDetection3D framework [1] provided by OpenMMLab and use the same evaluation procedure as the previous literature. For the SUN RGB-D dataset, the color information from the point clouds is not used for inference and is only used for visualization. For the KITTI dataset, the reflection intensity information from LiDAR point clouds is used as input for all methods.

3.3. Category-wise Performance

Table 2 shows per-category AP@25 results on the SUN RGB-D dataset for all methods under low SBR (0.02) conditions. Our approach shows significant gains for all categories, particularly larger gains for *small-sized* object categories (*e.g.* chair, nightstand, dresser) which typically suffer the most under low SBR conditions. Table 3 shows mAP on the KITTI val split for PV-RCNN architecture under low SBR (0.01) conditions, calculated with 11 recall positions in a standard format similar to previous works. PPC outperforms the baselines in all categories.

3.4. Visualizations and Observations

Fig. 8 to 11 show visualizations of 3D object detection on the SUN RGB-D dataset for all methods under different SBR conditions using the VoteNet architecture. Fig. 8 and 9 show complex scenes with a large number of small objects (*e.g.* chairs). Baselines fail to detect a lot of small and farther objects (last row of chairs), whereas PPC detects the most objects accurately. Fig. 10 and 11 show scenes with very low SBR conditions. Baselines fail to detect many objects, whereas our approach performs significantly better even in the presence of a large amount of noise. Fig. 12 shows a few failure cases for our method. PPC can sometimes detect multiple overlapping bounding boxes for the same object under noise. Couch or single-sitter couches are often detected as chairs by PPC or other baselines. Fig. 13 to 16 show scenes from the KITTI val dataset under varying SBR conditions using PV-RCNN architecture. Baselines fail to detect many objects, like farther cars and pedestrians, whereas our approach detects most objects.

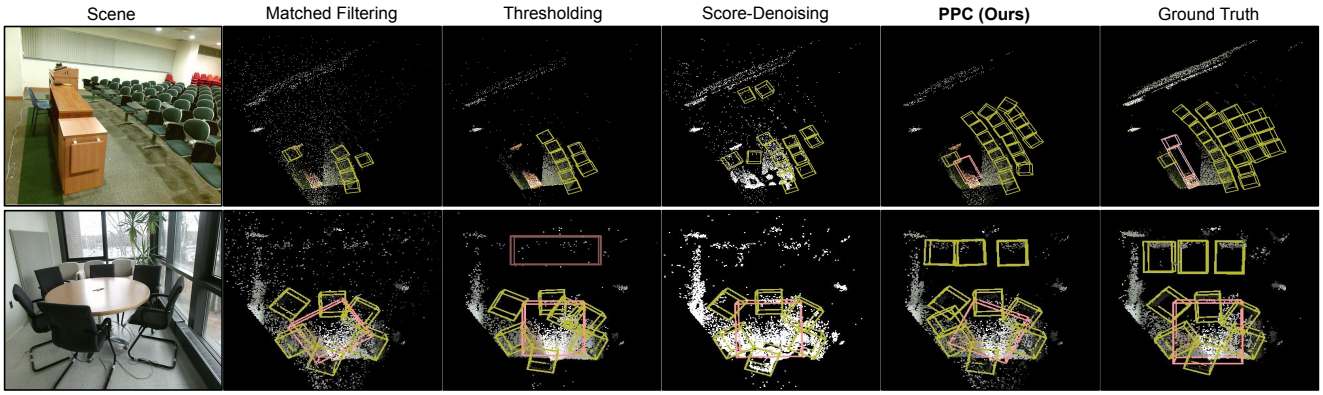


Figure 8. **3D Object Detection Results:** Figure shows scenes from the SUN RGB-D dataset under medium SBR (0.1) conditions. The first scene contains multiple rows of small objects (chair). Baselines fail to detect farther rows of chairs. Our approach detects most chairs and table.

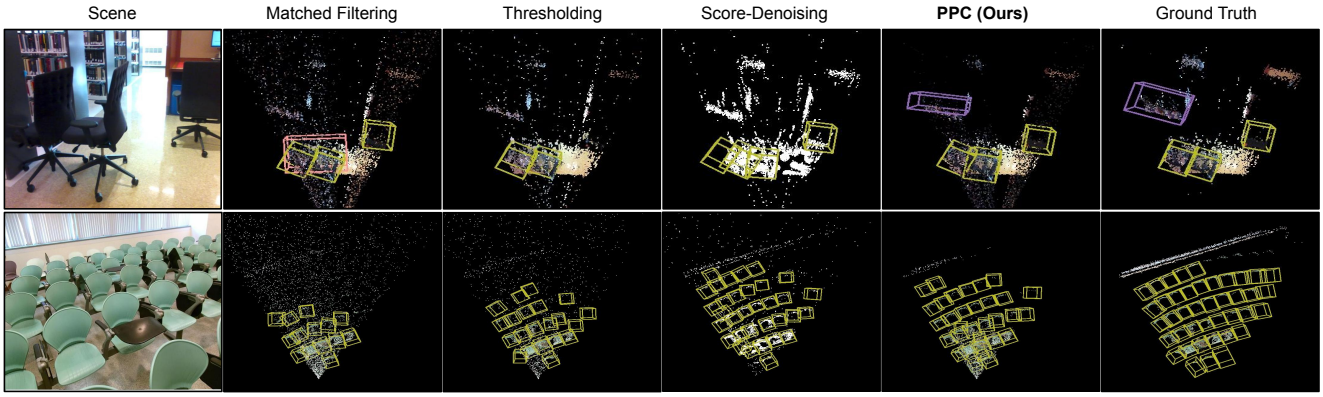


Figure 9. **3D Object Detection Results:** Figure shows scenes from the SUN RGB-D dataset under medium SBR (0.05) conditions. Scenes consist of small (chair) and farther objects (bookshelf). Our approach detects most objects with no false detections.

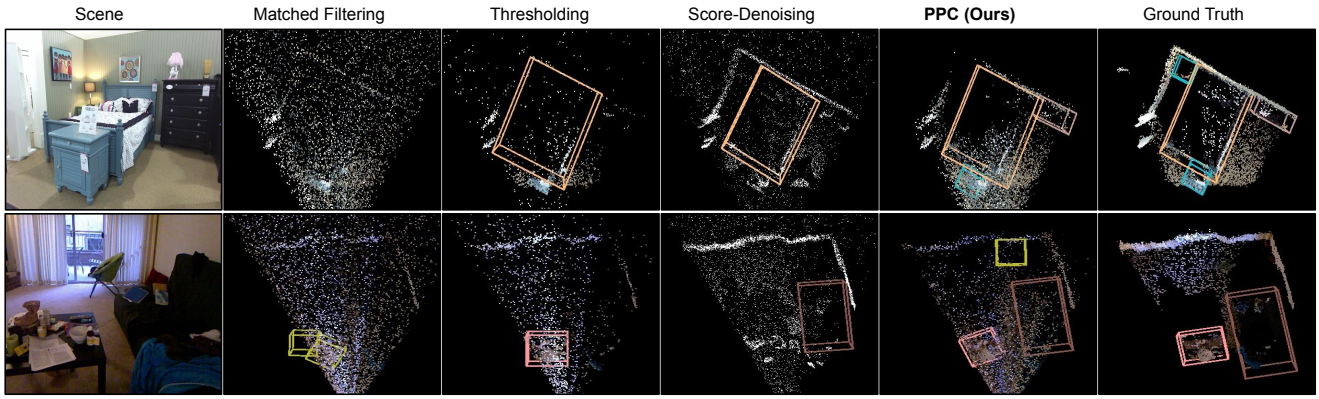


Figure 10. **3D Object Detection Results:** Figure shows scenes from the SUN RGB-D dataset under low SBR (0.02) conditions. Scenes consist of small (nightstand) and occluded objects (table). Our approach performs better than all baselines.

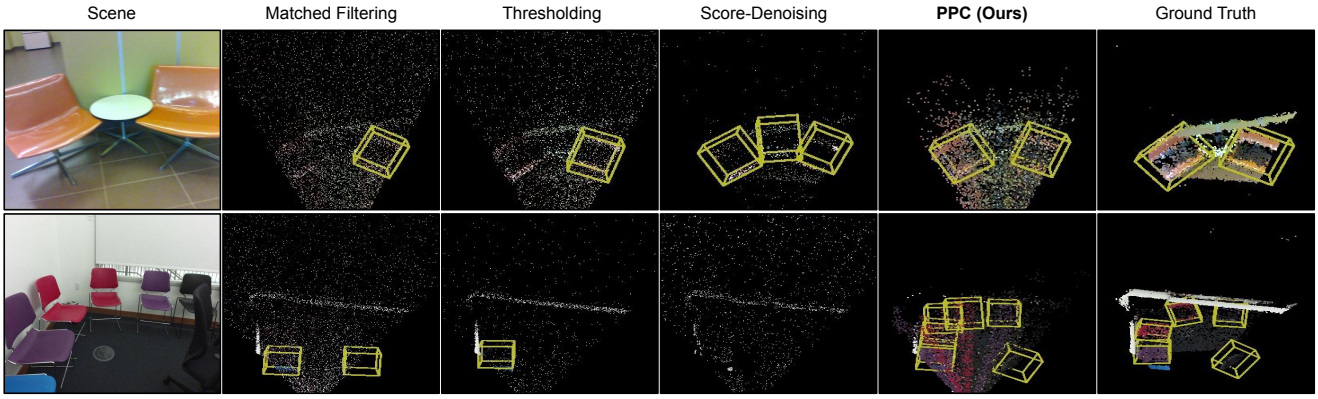


Figure 11. **3D Object Detection Results:** The figure shows scenes from the SUN RGB-D dataset under low SBR (0.01) conditions. Baselines fail to detect numerous objects (chair) due to noise, whereas our approach detects most objects in the scene.

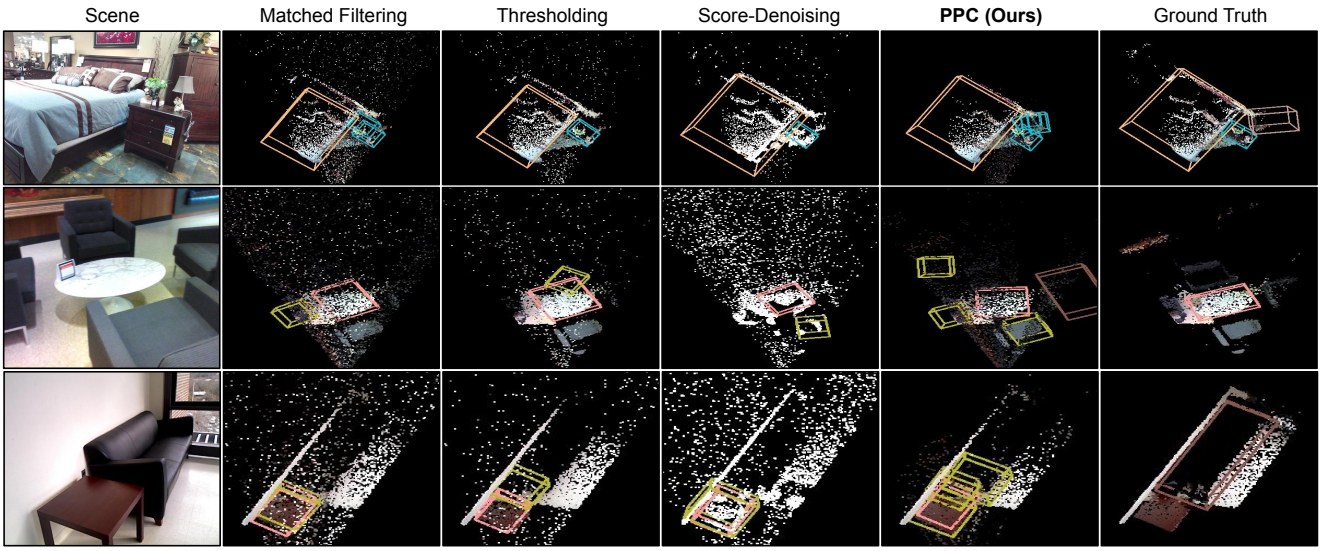


Figure 12. **3D Detection Failure Cases:** The first scene shows a scenario where PPC detects multiple boxes for the same object (night-stand). The second and third scenes show scenarios where a couch is detected as a chair by PPC and the baselines. Single-sitter couches are often detected as chairs by this model.

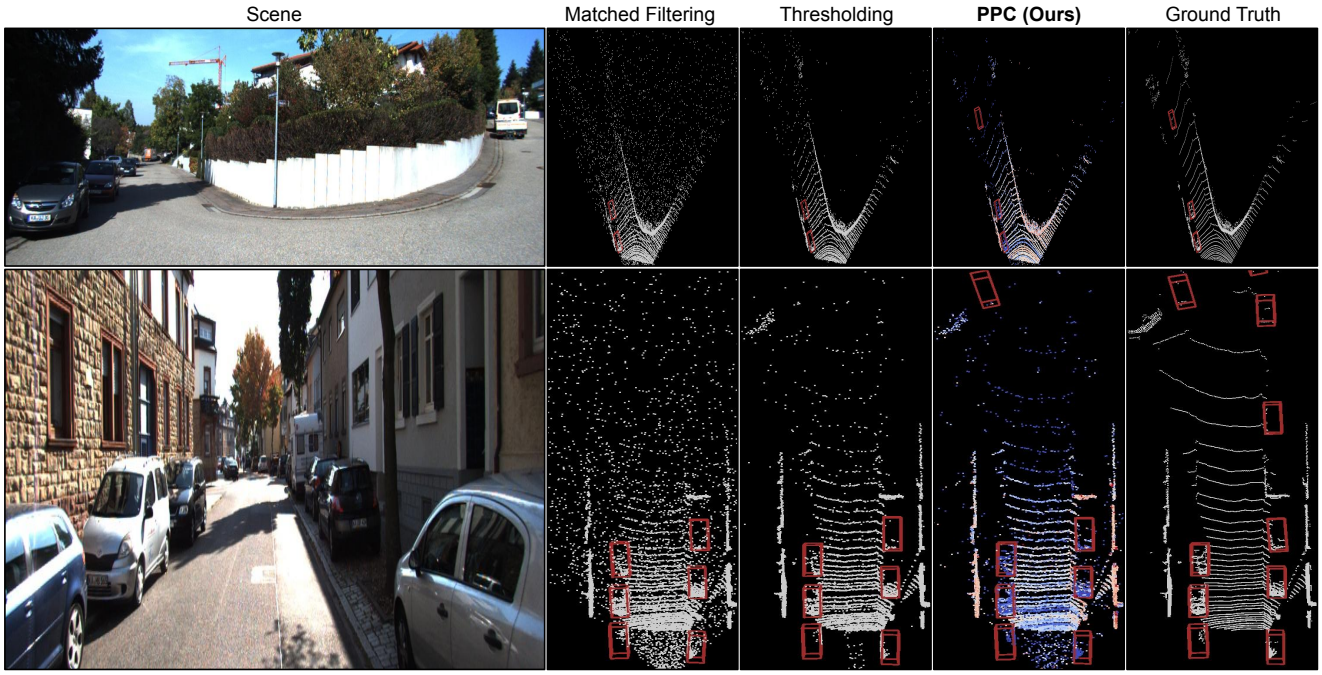


Figure 13. **3D Object Detection Results:** Figure shows scenes from the KITTI dataset under medium SBR (0.05) conditions. Baselines fail to detect farther cars. PPC is more robust for distant objects.

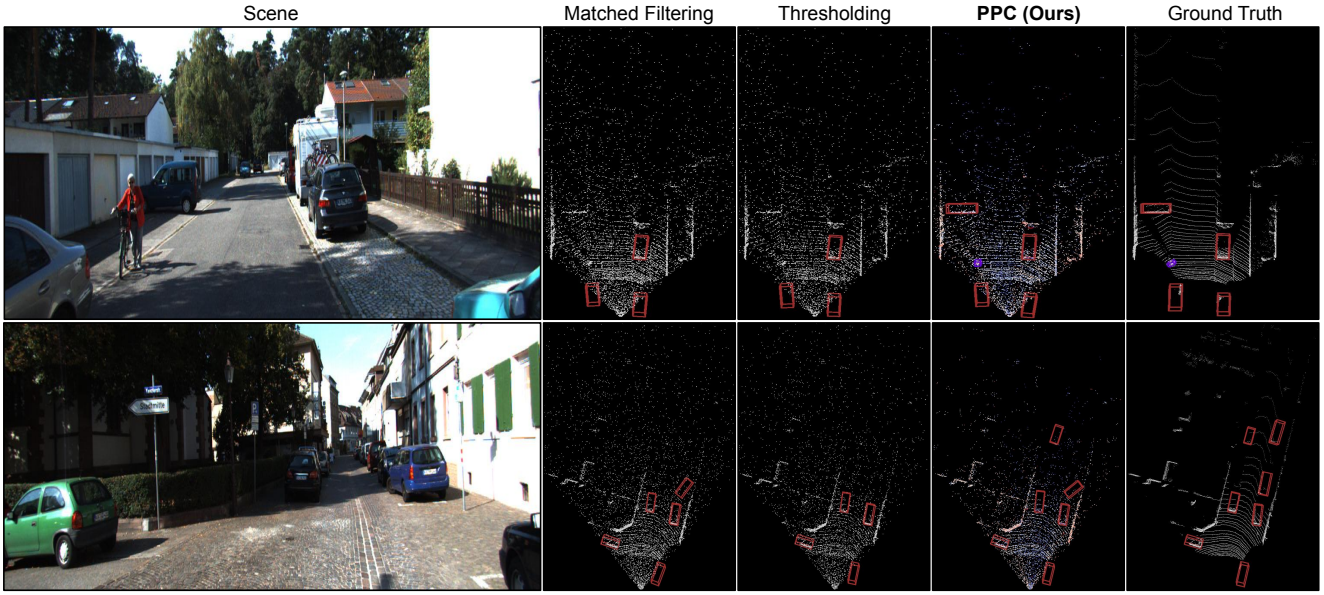


Figure 14. **3D Object Detection Results:** Figure shows scenes from the KITTI dataset under low SBR (0.02) conditions. The first scene shows a scenario where baselines fail to detect objects like farther cars and the pedestrian. PPC is more robust for distant objects.

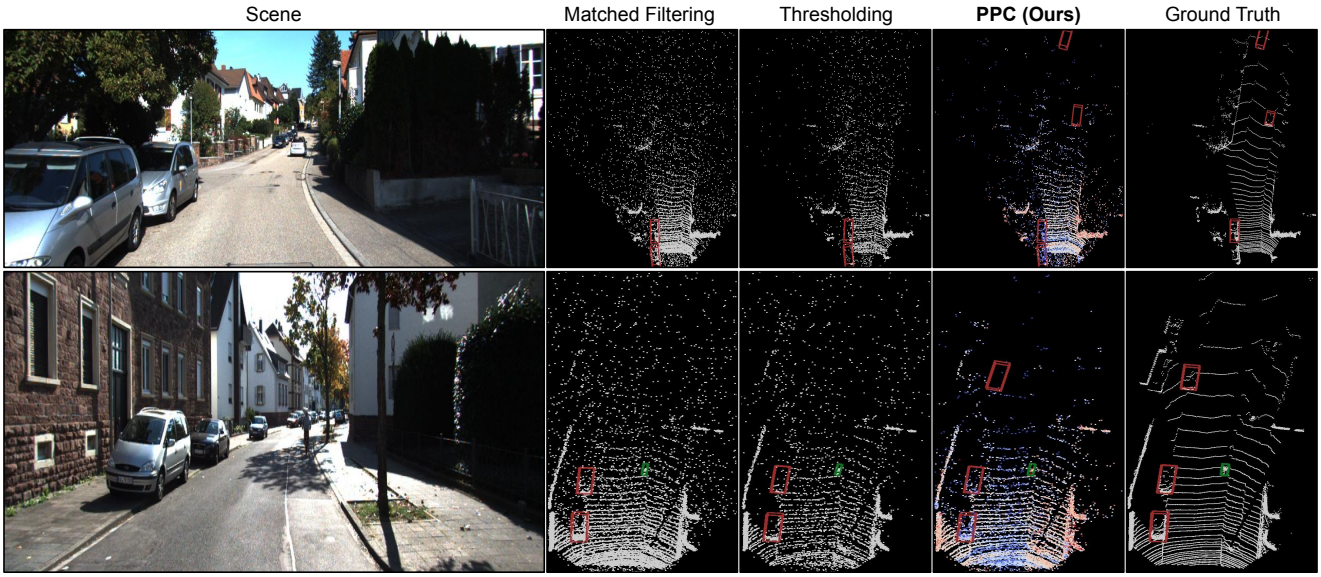


Figure 15. **3D Object Detection Results:** Figure shows scenes from the KITTI dataset under low SBR (0.01) conditions. Baselines struggle with farther cars, whereas PPC detects most objects.

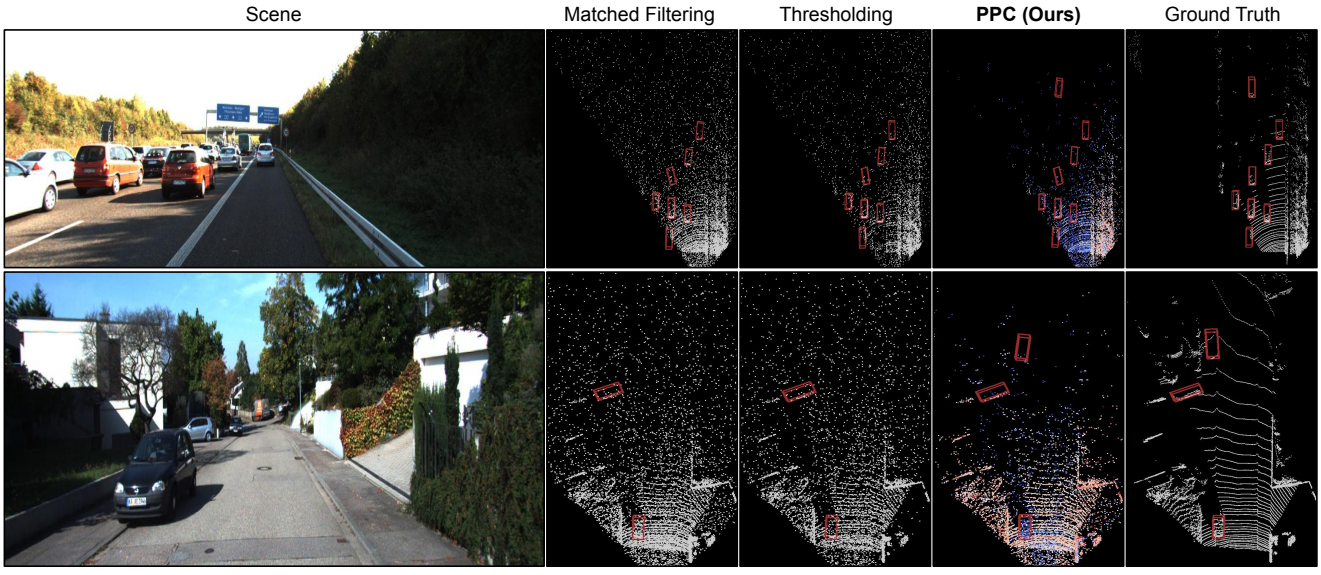


Figure 16. **3D Object Detection Results:** Figure shows scenes from the KITTI dataset under low SBR (0.005) conditions. Baselines fail to detect dark and farther objects like black cars. PPC is more robust for distant objects.

4. More 3D Detection Architectures

In this section, we evaluate our PPC approach using a variety of 3D object detection architectures. First, we evaluate a camera-LiDAR fusion approach, ImVoteNet [10]. Table 4 includes the comparison on the SUN RGB-D dataset, which shows significant improvement for all SBR levels. Second, we evaluate using a recent LiDAR-only transformer-based architecture Uni3DETR [15]. Table 5 includes mAP comparison on the SUN RGB-D dataset, which shows performance improvement for low SBR levels. Lastly, we evaluate using a Pillar-based architecture, PointPillars [6]. Table 6 includes mAP for car, pedestrian, and cyclist categories on moderate difficulty of KITTI val split, calculated with 11 recall positions. Our method shows significant improvements under low SBR conditions for pedestrian and cyclist categories.

PPC shows significant gains under low SBR for all detection architectures, which shows its versatility for a wide range of 3D detection models. The gain is large for point-net or transformer-based architectures (Uni3DETR, VoteNet, and ImVoteNet) as they suffer the most from the low SBR noise. The gain is significant but comparatively smaller for voxel or pillar-based architectures (PointPillars and PV-RCNN). Intuitively, this could be because the spurious points with large depth errors are away from the surface, and are not part of the same voxel or pillar as the surface points. Hence, their impact on the performance is also smaller.

4.1. Additional 3D inference tasks

Our approach is easy to extend to other 3D inference tasks, *e.g.* point cloud classification and point segmentation, with minimal modifications. Table below shows preliminary results for point cloud classification using PointNext [11] backbone on ScanObjectNN [14] dateset. PPC outperforms the baselines under low SBR conditions.

Avg. SBR	0.1	0.01
Matched Filtering	62.3	49.8
Thresholding	64.4	53.4
PPC (Ours)	70.5	58.4

Table 7. Classification overall accuracy on ScanObjectNN dataset.

Avg. SBR	Clean		0.1		0.05		0.02		0.01	
	AP@25	AP@50	AP@25	AP@50	AP@25	AP@50	AP@25	AP@50	AP@25	AP@50
Matched Filtering	63.37	35.51	53.89	27.64	53.23	24.67	37.54	10.99	33.17	7.98
Thresholding	64.25	36.17	59.57	33.44	58.82	32.45	42.43	18.17	39.51	12.61
PPC (Ours)	**64.36**	**36.94**	**61.51**	**35.69**	**60.19**	**31.38**	**53.21**	**25.37**	**46.84**	**20.14**

Table 4. **3D Detection Comparison using camera-LiDAR fusion architecture**: AP@0.25 and AP@0.50 results on the SUN RGB-D dataset using ImVoteNet show significant gains for PPC for all SBR levels.

Avg. SBR	Clean		0.1		0.05		0.02		0.01	
	AP@25	AP@50	AP@25	AP@50	AP@25	AP@50	AP@25	AP@50	AP@25	AP@50
Matched Filter	64.98	48.28	61.52	45.09	60.82	43.92	51.12	34.09	45.29	27.97
Thresholding	64.50	47.94	61.08	44.71	61.19	44.29	51.97	34.87	48.13	28.64
PPC (Ours)	**65.53**	**49.35**	**62.58**	**46.71**	**61.98**	**48.28**	**56.46**	**38.03**	**51.21**	**31.16**

Table 5. **3D Detection Comparison using LiDAR-only tranformer-based architecture**: AP@0.25 and AP@0.50 results on the SUN RGB-D dataset using Uni3DETR show significant gains for PPC under low SBR conditions.

Avg. SBR	Clean			0.05			0.02			0.01			0.005		
	Car	Ped	Cyc	Car	Ped	Cyc	Car	Ped	Cyc	Car	Ped	Cyc	Car	Ped	Cyc
Matched Filtering	77.08	**52.78**	64.49	68.25	49.52	58.96	64.13	47.67	46.45	54.43	41.61	41.76	45.03	32.46	31.06
Thresholding	**77.34**	52.09	64.81	68.06	49.63	59.09	63.87	47.92	46.96	54.18	40.88	42.18	45.11	32.79	31.89
PPC (Ours)	77.19	52.12	**65.21**	**69.12**	**50.23**	**62.44**	**65.63**	**49.27**	**48.09**	**56.39**	**45.77**	**44.46**	**47.24**	**38.74**	**34.89**

Table 6. **3D Detection Comparison using LiDAR-only pillar-based architecture**: mAP results for car, pedestrian, and cyclist categories on moderate difficulty of KITTI val split calculated with 11 recall positions for PointPillars. Our method shows significant gains under low SBR conditions.

5. Comparison with Denoised 3D Temporal Histograms

An effective approach for removing noise in 3D sensing systems described in this work is to denoise the 3D temporal histograms. Current state-of-the-art denoising methods for temporal histograms denoising [8] show high performance on depth reconstruction tasks. Hence, we also evaluate our approach and baselines using denoised temporal histograms. We use a 3D-CNN denoising model [8] to denoise the temporal bins, which are then used to construct point clouds for inference. We compare our method with the baselines under low SBR (0.02) conditions in Table 8. As expected, all methods perform better with denoised temporal histograms. Our method shows a further gain in AP@0.25 of about 3-4%, which shows that 3D inference can benefit from our PPC approach with denoised temporal histograms as well.

Histogram Denoising Method	Thresholding	PointClean Net	Score-Denoising	PathNet	**PPC (Ours)**
-	29.58	18.24	26.35	25.45	**38.49**
3D-CNN [8]	50.30	51.03	50.85	51.07	**54.25**
Gaussian Filter	38.79	40.12	43.36	43.25	**50.93**

Table 8. **3D Object Detection Results**: Comparison of AP@0.25 results using denoised temporal histograms.

We also compare the effectiveness of our method with a non-learning-based histogram denoising method. We use a 5x5 Gaussian filter in the spatial dimension to denoise histograms. Matched filtering is still used over the temporal dimension. Table 8 shows that our method has significant gains with Gaussian denoised histograms, and the performance is comparable to the results with 3D-CNN denoising.

Should we denoise 3D temporal histograms for inference? Denoising methods like [8] require compute and memory-intensive 3D-CNN operations, which makes it infeasible for real-time applications. It is thus not suitable for sensor on-chip processing. It also requires a read-out of full 3D temporal histograms, which has a significantly high data-bandwidth cost (compared to only reading out a point cloud as considered in earlier approaches in the main paper). However, it is an effective approach for denoising in non-real-time applications.

The Gaussian filter is a computationally cheap, non-learning-based operation and is feasible with sensor on-chip processing. Table 9 shows the inference time of our complete recognition pipeline (including denoising temporal histograms, point cloud processing, and inference) with different histogram denoising methods that are discussed. This suggests that, given a computational budget, a simple histogram denoising approach like a Gaussian filter is a good candidate for 3D recognition.

Histogram Denoising Method	Runtime (ms)
-	95
3D-CNN	7200
Gaussian Filter	98

Table 9. **Runtime Time for 3D Detection:** Comparison of per-scene runtime time of our method using different histogram denoising methods.

5.1. Comparison with Compressed 3D Timing Histograms

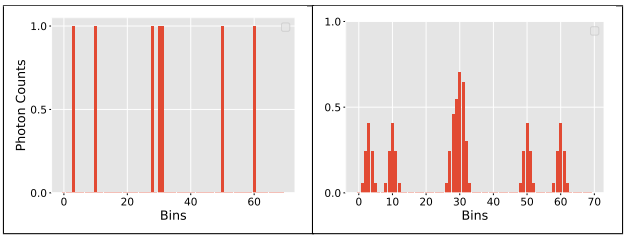
Recently, compression techniques have been proposed to read out compressed representations [4] of the temporal histograms to reduce data bandwidth requirements. We also show the performance of our approach on such decompressed histograms in Table 10. We use a lightweight Truncated Fourier (k=32) representation from [4] for evaluation. Our approach is effective even under the data loss incurred due to compression and shows significant gains over the Thresholding baseline for 3D detection.

	Thresholding	**PPC (Ours)**
Decompressed Histograms	16.50	**29.77**

Table 10. **3D Object Detection Results**: Comparison of AP@0.25 results using decompressed histograms.

6. Point Clouds Under Extremely Low SBR Conditions

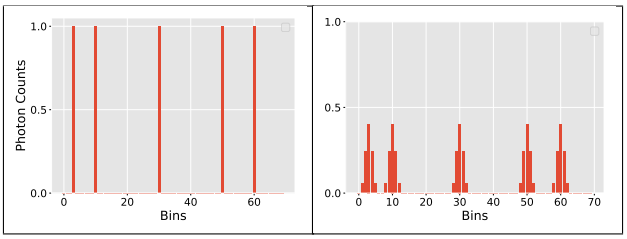
We use matched filtering output from the timing histograms to find the peak and compute the probability attribute for each point. Under extreme conditions of low signal, there may not be any bin with more than one detected photon. The matched filtering output can still provide a dominant peak corresponding to the signal, and hence a correct point in the point cloud, if there are 'clusters' of photon detections in nearby bins. Figure 17 illustrates this using a histogram and its matched filtering output.



(a) Raw Histogram (b) Matched Filtering Output

Figure 17. An example of a raw histogram and matched filtering output for a LiDAR measurement under extremely low SBR.

Under ultra-low SBR, or when there is no return signal, there could be multiple sparsely located bins with single photon detection. Figure 18 shows an example to illustrate this scenario. In such cases, there is no peak corresponding to more than one photon in the matched filtering output. We use a minimum height threshold to ignore such points in the 3D point cloud.



(a) Raw Histogram (b) Matched Filtering Output

Figure 18. An example of a raw histogram and matched filtering output for a LiDAR measurement under extremely low SBR.

References

[1] MMDetection3D Contributors. MMDetection3D: OpenMMLab next-generation platform for general 3D object detection. https://github.com/open-mmlab/mmdetection3d, 2020. 5

[2] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012. 5

[3] OneFive GmbH. Katana-05 hp laser. https://www.laserlabsource.com/Solid-State-Lasers/solid-state-and-fiber-lasers/Picosecond-Laser-532nm-5--J-Onefive, 2024. 1

[4] Felipe Gutierrez-Barragan, Atul Ingle, Trevor Seets, Mohit Gupta, and Andreas Velten. Compressive single-photon 3d cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17854–17864, 2022. 11

[5] Robert K. Henderson, Nick Johnston, Francescopaolo Mattioli Della Rocca, Haochang Chen, David Day-Uei Li, Graham Hungerford, Richard Hirsch, David Mcloskey, Philip Yip, and David J. S. Birch. A 192 × 128 time correlated spad image sensor in 40-nm cmos technology. *IEEE Journal of Solid-State Circuits*, 54(7):1907–1916, 2019. 1

[6] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12697–12705, 2019. 10

[7] Shitong Luo and Wei Hu. Score-based point cloud denoising. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4583–4592, 2021. 5

[8] Jiayong Peng, Zhiwei Xiong, Xin Huang, Zheng-Ping Li, Dong Liu, and Feihu Xu. Photon-efficient 3d imaging with a non-local neural network. In *European Conference on Computer Vision*, pages 225–241. Springer, 2020. 11

[9] Adaps Photonics. Ads6311 spad array. https://www.adapsphotonics.com/en/product-55669-218658.html, 2024. 1

[10] Charles R Qi, Xinlei Chen, Or Litany, and Leonidas J Guibas. Imvotenet: Boosting 3d object detection in point clouds with image votes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4404–4413, 2020. 10

[11] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *Advances in neural information processing systems*, 35:23192–23204, 2022. 10

[12] Marie-Julie Rakotosaona, Vittorio La Barbera, Paul Guerrero, Niloy J Mitra, and Maks Ovsjanikov. Pointcleannet: Learning to denoise and remove outliers from dense point clouds. In *Computer Graphics Forum*, pages 185–203. Wiley Online Library, 2020. 5

[13] Shuran Song, Samuel P Lichtenberg, and Jianxiong Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 567–576, 2015. 5

[14] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Duc Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *International Conference on Computer Vision (ICCV)*, 2019. 10

[15] Zhenyu Wang, Ya-Li Li, Xi Chen, Hengshuang Zhao, and Shengjin Wang. Uni3detr: Unified 3d detection trans-

former. In *Advances in Neural Information Processing Systems*, pages 39876–39896. Curran Associates, Inc., 2023. 10

[16] Zeyong Wei, Honghua Chen, Liangliang Nan, Jun Wang, Jing Qin, and Mingqiang Wei. Pathnet: Path-selective point cloud denoising. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(6):4426–4442, 2024. 5