Appendix The appendix is organized as follows. We present details of the K-Radar dataset, the sensor suite, and criteria of conditions (weather conditions, road structures, and collection time) in Section A, B, and C, respectively. We also provide details of the annotation/calibration process and the baseline neural networks (NNs) in Section D and E, respectively. We discuss results regarding each weather condition and consideration of the K-Radar dataset as a pre-training dataset for other Radar tensor datasets in Section F and G, respectively. Finally, we introduce details of devkits and list relevant URLs to help with understanding the content of the paper in Section H and I, respectively.

A Additional details for K-Radar dataset

In this section, we present additional samples of K-Radar dataset, sequence distribution, dataset composition, license, and privacy concerns.

A.1 Additional samples of the K-Radar dataset and explanation of LPCs for each weather condition

In the sleet (Figure 8-(e)) or heavy snow (Figure 8-(g)) condition, the Lidar point cloud (LPC) measurements of some objects ahead are lost when the ego-vehicle is driving. Conversely, in the rain (Figure 8-(d)) or light snow (Figure 8-(f)) condition, LPC measurements of objects exist. The reason for this is as follows. Sleet is a mix of rain and snow that freezes when it falls from the sky in a liquid

Figure 8: Additional samples of K-Radar datasets for various weather conditions. (1) 4DRTs, (2) front view camera images, and (3) Lidar point clouds (LPCs) of three different road conditions with the same weather condition are depicted in three boxes in each column. In this example, yellow, red, and blue bounding boxes represent the sedan, bus or truck, and pedestrian classes, respectively. Objects with all LPC measurements missing due to the adverse weather are marked with purple dotted lines. More samples of K-Radar dataset can be visualized using the devkits program described in Section H.

state and comes into contact with a sensor or an ego-vehicle colder than the air (Zhou et al., 2016). In our case, the sleet freezes on the front surface of the Lidar sensor, creating a layer of frost. Thus, as shown in Figure 8-(e), the measuring signals from the Lidar cannot reach objects in the front of the ego-vehicle, which results in missing points in the LPC. In addition, heavy snow is a weather condition in which snow falls over 1 cm per hour as described in Table 8, and Figure 3 shows that a lot of snow accumulates on the front surface of the sensor after the vehicle drives forward for 5 minutes. For this reason, similar to sleet, some LPC measurements of objects in front of the ego-vehicle are missing, as shown in Figure 8-(g). Unlike sleet and heavy snow, there is only a little-to-no amount of snow accumulation on the front surface of the Lidar sensor in light snow condition, as described in Table 8. In addition, LPC measurements of objects are also partially available in rain condition, since raindrops slip over the front surface of the Lidar sensor. Therefore, the Lidar sensors can measure objects in front of the ego-vehicle as shown in Figure 8-(d) and (f). Note that we provide a video clip (Figure 9) in Section I URL 2 to show sensor measurements collected while driving forward under the heavy snow condition.

A.2 Sequence distribution

The K-Radar dataset provides a total of 35K frame data obtained in different weather conditions, road structures, and collection time. The dataset is divided into 58 sequences, where the details of each sequence are shown in Table 5.

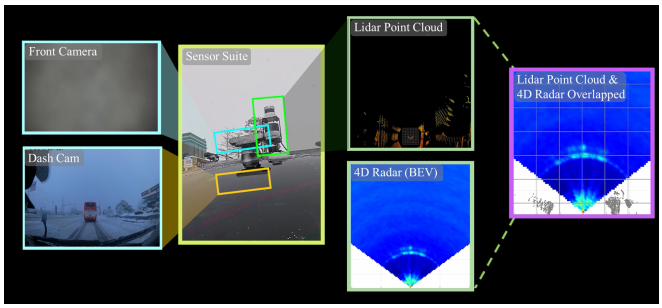


Figure 9: A snippet of the video clip that shows each sensor measurement dynamically changing during driving under the heavy snow condition. (see Section I URL 2)

A.3 Dataset composition

Each of the 58 sequences consists of 12 compressed folders, as shown in Table 6. Table 6 provides information on the folder name, data type, extension, size, and the usage of each folder.

A.4 License

The K-Radar dataset is published under the CC BY-NC-ND License, and all codes are published under the Apache License 2.0.

A.5 Privacy concerns

We confirm that all the image sequences with pedestrians, bicycles, and motorcycles do not have recognizable faces. Although everyone is wearing a mask due to COVID-19, we have taken additional precautions and blurred all faces to protect their privacy as shown in Figure 10.

Figure 10: Examples of front images showing people whose faces are blurred.

Table 5: Sequence of the K-Radar dataset; sequences 1 through 20 are obtained in Dae-jeon, and sequences 21 through 58 are obtained in Gang-won Province. 'he. snow' and 'park.lot' denotes heavy snow and parking lot, respectively.

Seq.	Num. Fr.	Weather Cond.	Road Stru.	Time	Seq.	Num. Fr.	Weather Cond.	Road Stru.	Time
1	597	normal	urban	night	30	470	sleet	park.lot	day
2	462	normal	highway	night	31	598	sleet	suburban	day
3	597	normal	highway	night	32	597	rain	suburban	day
4	588	normal	highway	night	33	598	rain	suburban	day
5	597	normal	urban	day	34	598	rain	suburban	night
6	594	normal	urban	night	35	597	sleet	park.lot	night
7	595	normal	alleyway	night	36	597	sleet	park.lot	night
8	567	normal	university	night	37	597	sleet	suburban	night
9	833	normal	highway	day	38	597	fog	mountain	day
10	1130	normal	highway	day	39	597	fog	mountain	day
11	1195	normal	highway	day	40	598	fog	mountain	day
12	888	normal	highway	day	41	597	fog	mountain	day
13	227	overcast	highway	day	42	598	light snow	urban	day
14	595	normal	urban	day	43	598	light snow	urban	day
15	591	normal	urban	day	44	597	fog	shoulder	day
16	578	normal	university	day	45	592	fog	shoulder	day
17	593	normal	university	day	46	598	he. snow	highway	night
18	594	normal	urban	day	47	266	he. snow	highway	night
19	592	normal	alleyway	day	48	443	light snow	highway	night
20	595	normal	urban	day	49	598	light snow	highway	night
21	597	rain	alleyway	night	50	597	sleet	highway	night
22	598	overcast	urban	night	51	597	sleet	highway	night
23	598	rain	urban	night	52	598	sleet	highway	night
24	598	rain	urban	night	53	597	sleet	highway	day
25	597	rain	urban	night	54	601	he. snow	urban	day
26	597	rain	suburban	day	55	494	he. snow	urban	day
27	598	sleet	suburban	day	56	598	he. snow	urban	day
28	597	sleet	mountain	day	57	598	he. snow	urban	day
29	597	sleet	mountain	day	58	598	he. snow	urban	day

Table 6: Dataset composition of each sequence. 'res.', 'cam.', and 'img.' denotes resolution, camera, and image, respectively.

folder name	data type	extension	size	usage
radar_tesseract	4DRT	.mat	360GB	network input, visualization
radar_xyz_cube	3DRT-XYZ	.mat	72GB	network input
os1-128	High res. LPC	.pcd	14GB	network input, visualization
os2-64	Low res. LPC	.pcd	7GB	network input, visualization
cam-front	Front cam. img.	.png	4.5GB	network input, visualization
cam-left	Left cam. img.	.png	4.5GB	network input, visualization
cam-right	Right cam. img.	.png	4.5GB	network input, visualization
cam-rear	Rear cam. img.	.png	4.5GB	network input, visualization
cam-dash	Dash cam. img.	.mp4	75MB	reference video of annotation
info_calib	Calibration values	.txt	-	calibration of 4DRT and LPC
info_condition	Conditions	.txt	-	conditional evaluation
info_label	Labels	.txt	0.5MB	training, evaluation

B Details of the sensor suite

We use waterproofed sensors with grade IP66 or higher, as mentioned in Section 3.1, for safe data collection in adverse weather conditions. Table 7 summarizes the detailed information (i.e., model

name, output data format, resolution, maximum operating distance, field of view (FOV), frames per second (FPS)) of the sensors that we install for the K-Radar data collection.

Table 7: Sensor suite details: 'Azi.', 'Ele.', 'res.' and 'CEP' denotes azimuth, elevation angle, resolution, and circular error probability, respectively.

sensors	model name	output data	resolution	max range	FOV (Azi., Ele.)	FPS
4D Radar	RETINA -4ST	64×256×107°×37° size 4D tensor	0.06m/s, 0.46m,1°,1°	118m	107°, 37°	10
long range Lidar	os2-64	131,072 3D points	0.1cm, 0.18°,0.35°	240m	360°, 22.5°	10
high res. Lidar	os1-128	262,144 3D points	0.1cm, 0.18°,0.35°	120m	360°, 45°	10
4 stereo cameras	ZED2i	8 1280×720 size images (left, right)	1280x720 pixels	n/a	110°, 70°	30
RTK-GPS	GPS500, C94-M8P3	latitude, longitude, altitude	0.025m + 1ppm CEP	n/a	n/a	1
2 IMUs	built-in Lidar	6-axis IMU data	n/a	n/a	n/a	100

C Criteria for weather conditions, road structures, and collecting time

We establish conditions for each sequence according to the criteria in Table 8, as mentioned in Section 3.2.

Table 8: Detailed criteria for each condition.

Criteria	Name	Detailed criteria
road structures	urban	Roads with four or more lanes and traffic lights, and ego-vehicle average speed is around 60 km/h
	highway	Roads without traffic lights and ego-vehicle average speed is around 100km/h
	alleyway	Roads with two to four lanes and buildings nearby
	suburban	Two- or four-lane roads with rice paddies, fields and mountains around it
	university	The inner roads of KAIST
	mountain	Sloped roads with two to four lanes in a countryside
	parking lots	Areas for stopping or parking with other vehicles around
	shoulder	Parking spaces by the side of the road
weather conditions	normal	Clear weather that does not meet the six weather conditions below
	overcast	Sunless, cloudy weather
	fog	Weather in which distant objects are dimly visible due to omni-directional fog
	rain	Rainy weather
	sleet	Precipitation that consists of both rain and snow
	light snow	Snowfall within approximately 1 cm per hour
	heavy snow	Snowfall that exceed 1 cm per hour
time zone	day	Approximately 6:00 $\sim$ 16:00
	night	Approximately 20:00 $\sim$ 4:00

D Details of annotation and calibration

D.1 Details of annotation

Annotation process for calibrated LPC measurements As mentioned in Section 3.3, it is difficult to intuitively recognize the shape of objects in BEV-2D. Therefore, we utilize the calibrated LPC (Section D.2) with a maximum calibration error of 0.5cm to enable accurate 3D bounding box annotations. We include the annotation program and code in the published devkits. The annotation program supports a resolution of 1.4 cm per pixel, resulting in a maximum annotation error of 0.7 cm. The annotation program can be used by following the two steps: (1) annotate the BEV bounding box of an object in the visualized BEV LPC, (2) annotate the height and center point of the BEV bounding box. In Figure 11, we show the GUI and usage of the annotation program, and detailed instructions can be found in the video clip that is available at Section I URL 3.

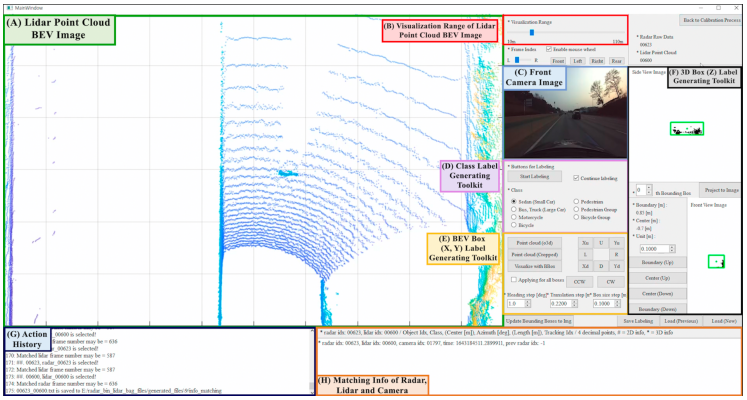


Figure 11: A snippet of the video clip that shows annotation process. (see Section I URL 3)

Annotation process in the absence of LPC measurements of objects As mentioned in Section 3.3, the annotation program we provide has a function to overlap the calibrated BEV-2D to the LPC so that annotations can be created even in the absence of LPC measurements of objects for various reasons such as adverse weather conditions. To annotate objects in the absence of LPC measurements, the human annotator processes 3D bounding box annotation by referring to overlapped BEV-2D and dash camera images of the ego-vehicle. The human annotator then verifies the height and size information of the 3D bounding box with BFS-2D, as shown in Figure 6-(b). We note that the height of the vehicle is set to a pre-defined value after checking the type of the vehicle through the dash cam image. Figure 12 illustrates the GUI of the annotation program in the absence of LPC measurements, and more detailed instructions can be found in the video clip available at Section I URL 4.

D.2 Details of the calibration between 4D Radar and Lidar

Accurate calibration of the 4DRT and LPC is crucial to utilize the 3D bounding box annotated in the LPC as a label of the 4DRT. We utilize visualized BEV-2D and LPC as well as spatial information (sensor placement location) of all sensors to precisely calibrate 4DRT and LPC. We develop a program that matches temporal offset (i.e., frame error) and spatial offset (i.e., 2D translation, yaw) through near-field (within about 30m) objects which are clearly visualized (calibration clue shown in Figure 13), as shown in Figure 14. The GUI program in Figure 14 supports a resolution of 1 cm per pixel, resulting in a maximum calibration error of 0.5 cm. We have not considered the pitch angle difference between 4D Radar and Lidar, since we fix the sensors precisely perpendicular to the ground, resulting in no theoretical difference in the pitch angle. We note that the video clip containing the calibration process (shown in Figure 14) is available in Section I URL 3, and the video clip containing the calibration result (shown in Figure 15) is available in Section I URL 5. Through the calibration process, we note that the vehicle approaching from the other side matches correctly as shown in the calibration result (shown in Figure 15).

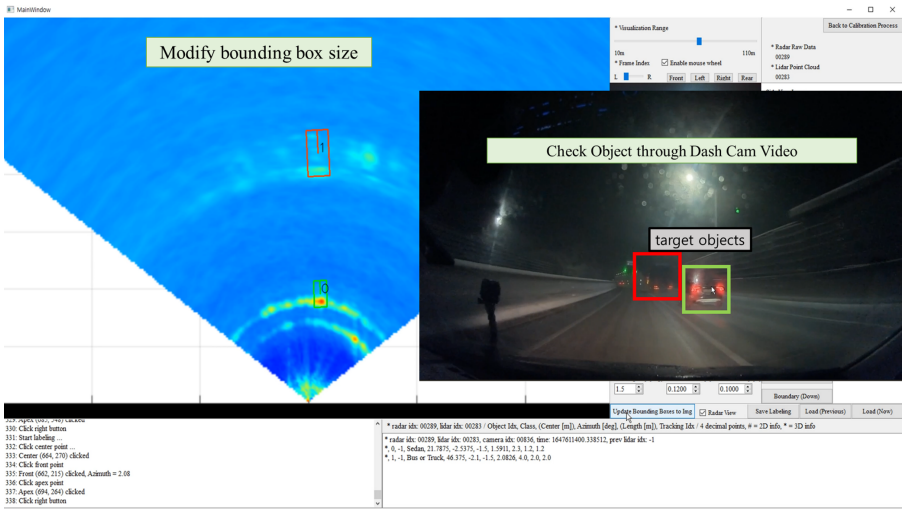


Figure 12: A snippet of the video clip that shows the annotation process in the absence of LPC measurements of objects. (see Section I URL 4)

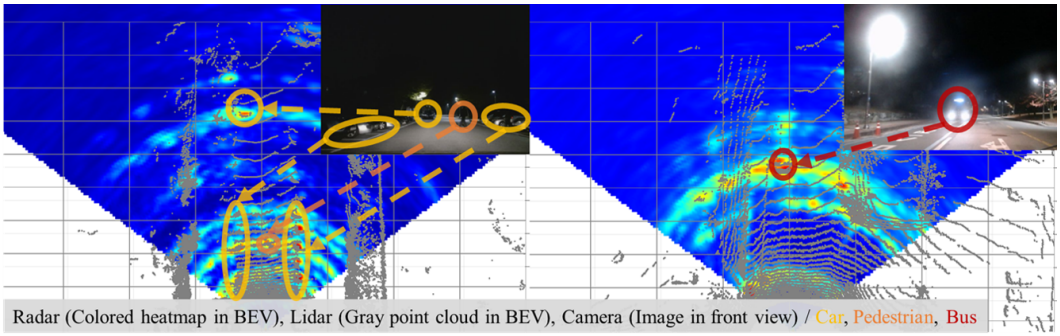


Figure 13: Examples of calibration clues.

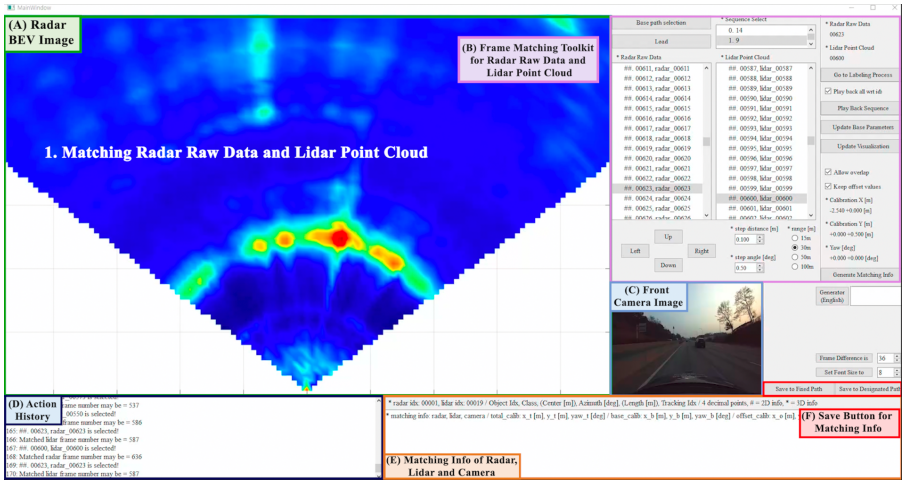


Figure 14: A snippet of the video clip that shows 4DRT/LPC calibration process through BEV-2D and LPC visualization. (see Section I URL 3)

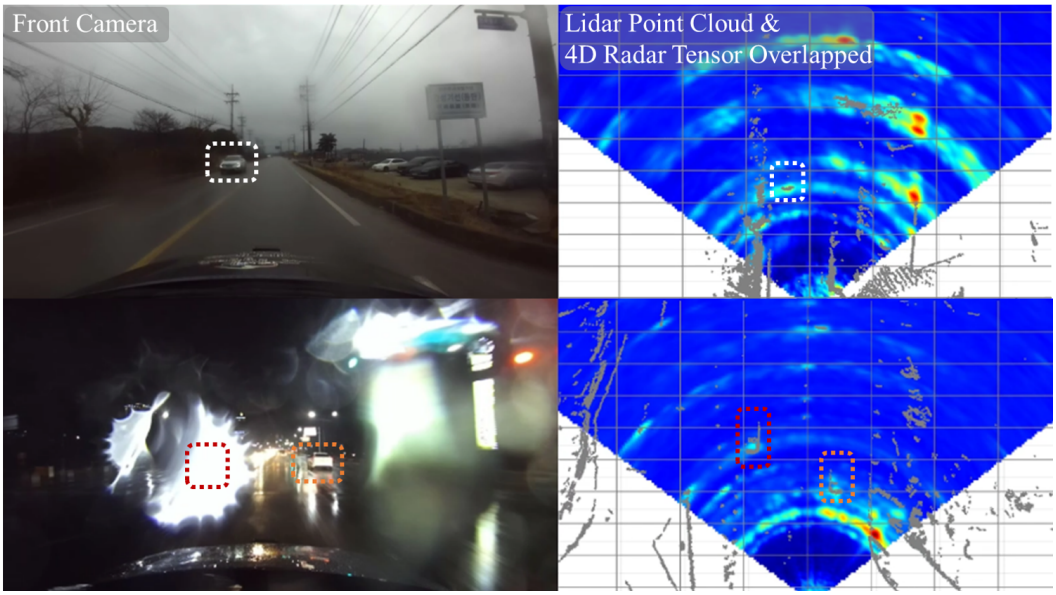


Figure 15: A snippet of the video clip that shows calibration results images for two different roads. (see Section I URL 5)

D.3 Details of the calibration between Lidar and camera

The calibration of the Lidar and camera is to determine a total of three parameters: 1) extrinsic parameters to define the relative position of the Lidar coordinate system (i.e., reference frame) and the camera coordinate system, 2) lens distortion parameters to correct camera distortion, and 3) intrinsic parameters to match each pixel in the pixel coordinate system with the points in the camera coordinate system. As shown in Figure 16, we scan a 3D model of the ego-vehicle with the sensor suite using a Lidar scanner provided in iPhone 12 Pro (Luetzenburg et al., 2021). We extract the coarse position of each sensor from the scanned 3D vehicle model. Second, we extract the lens distortion parameters and the intrinsic parameters of the camera using the camera calibration process provided by ROS (Stanford Artificial Intelligence Laboratory et al.). The previous two processes extract approximate calibration parameters, which may include calibration errors. Therefore, we construct a GUI program that can modify each parameter finely, as shown in Figure 17, and fine-tune the parameters so that the measurements of the camera and the LIDAR at the close and far objects match accurately, as shown in Figure 18.

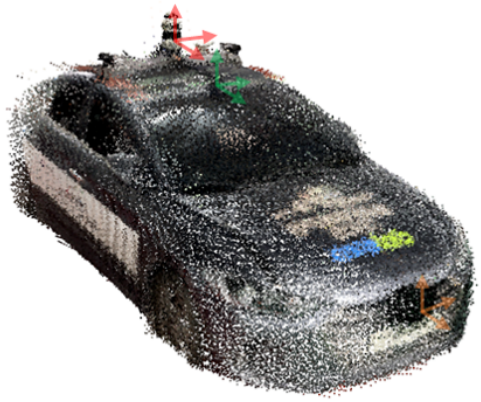


Figure 16: The scanned 3D model of the ego-vehicle with sensor suite.

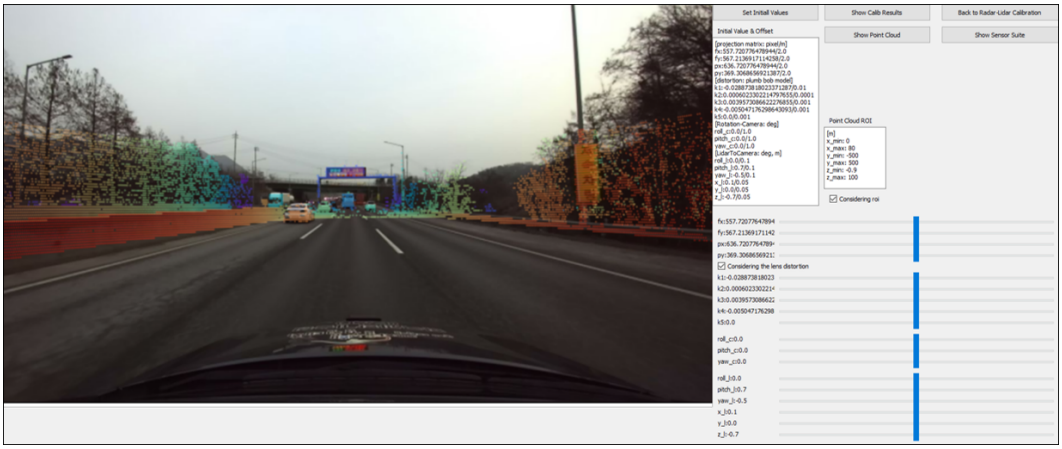
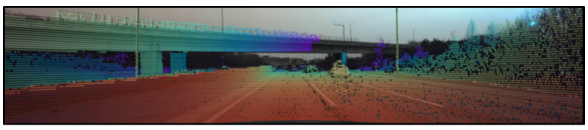


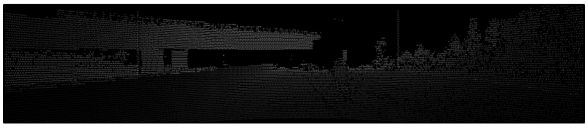
Figure 17: The GUI program to fine-tune the calibration parameters between Lidar and camera.

Figure 18: Examples of calibration result between the camera and Lidar, where colored points on the front images show the projected points of the corresponding LPCs.

In addition, we obtain the ground-truth depth value of the corresponding pixel through the points projected onto the camera image. Because the LPC is sparse, as shown in Figure 19-(b), the dense depth map is provided through interpolation, as shown in Figure 19-(c). We note that these depth maps can magnify the utilization of our dataset for the depth estimation tasks (Mertan et al., 2022), which is one of the most widely studied fields in computer vision.



(a) A front camera image with projected points of the LPC.



(b) A sparse depth map before interpolation.

(c) A dense depth map after interpolation.

Figure 19: An example of generating a depth map based on calibration result.

E Details of baseline NNs

In this section, we describe the common structures of RTNH and RTN, neck, and head of the baseline NNs, and the structures of 3D-SCB and 2D-DCB, which are the backbone of RTNH and RTN, respectively.

E.1 Neck and head

As mentioned in Section 3.4, both RTNH and RTN extract multiple feature maps (FMs) of different resolutions. Neck transforms the FMs into the same size by applying TransposeConv2D and concatentes the transformed FMs (Lin et al., 2017a). The size of the concatenated FM is $C_{FM} \times Y_{FM} \times X_{FM}$. C_{FM}, Y_{FM}, and X_{FM} represent the number of channels of the concatenated FM, the number of grids for the left and right widths, and the number of grids for the front distance, respectively. The head predicts the bounding boxes from the concatenated FM using an anchor-based method as in Ren et al. (2015), and its structure is shown in Figure 20.

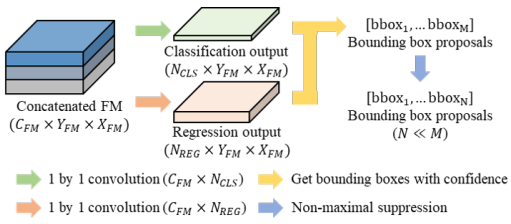


Figure 20: Head structure.

We apply 1 by 1 convolution to the concatenated FM to extract classification and regression output for each grid, as shown in Figure 20. We use two anchor boxes with yaw angles of 0° and 90° for each class, resulting in $N_{CLS} = 2$(Anchor)+1(Background)$= 3$. In addition, we assign a total of eight parameters for each anchor: center point (x_c, y_c, z_c), length, width, height (x_l, y_l, z_l), $cos(yaw)$, and $sin(yaw)$ (Simony et al., 2018) of the bounding box, resulting in $N_{REG} = 8(N_{CLS} - 1) = 16$. We then extract M bounding box proposals from the classification and regression outputs. In training process, proposals with an intersection over union (IOU) of 0.5 or more with respect to the ground-truth are classified as positive bounding boxes, and proposals with an IoU of less than 0.2 are classified as negative bounding boxes. We apply the focal loss (Lin et al., 2017b) to cope with the problem of class imbalance between positive bounding boxes and negative bounding boxes, and apply the smooth L1 loss between the regression value and the target value. During inference, an index with the largest logit value from the classification output is inferred as the proposal's class, and a confidence threshold of 0.3 is applied, so that low-confidence predictions are regarded as backgrounds. Thereafter, non-maximal suppression is applied to remove overlapping bounding boxes and finally a total of N bounding boxes are obtained.

E.2 3D-SCB

As mentioned in Section 3.4, we extract FMs using 3D sparse conv blocks to reduce the usage of GPU memory, while still using height information from the 4DRT. A 3D sparse conv block consists of a total of three consecutive 3D convolution layers. We set the first 3D convolution layer as 3D sparse convolution layer (Liu et al., 2015) and the remaining 3D convolution layer as 3D submanifold convolution layer (Graham et al., 2018). The output of the 3D sparse conv block is a sparse FM with four dimensions (channel, height, width, length) of different resolutions. Each sparse FM is transformed into its dense tensor counterpart, and then TransposeConv2D is applied to the three-dimensional dense FMs, resulting in dense FMs represented in BEVs with height information encoded. Finally, all dense FMs are concatenated to produce the final concatenated FM, which is the input of the head.

E.3 2D-DCB

We construct a 2D dense conv backbone (2D-DCB) with 2D conv blocks, as mentioned in Section 3.4, to extract FMs without encoding the height information. We utilize ResNet50 (He et al., 2016) and ResNext101 (Xie et al., 2017) as the 2D conv blocks whose performance has been validated on tasks such as classification (Mahajan et al., 2018) and object detection (Qiao et al., 2021). We compare object detection performance for two variations, as shown in Table 9, and in Section 4.2,

we show the results of 2D-DCB-ResNext101, which has higher performance among the two, as the representative result of RTN.

Table 9: Performance of two variants of RTN.

backbone	AP_{3D} [%]	AP_{BEV} [%]	GPU RAM [MB]
2D-DCB-ResNext101	**40.12**	**50.67**	520
2D-DCB-ResNet50	39.86	49.37	**257**

F Qualitative results of RTNH and PointPillars with additional discussion in various conditions

We show the object detection results of RTNH and PointPillars (Lang et al., 2019) under various weather conditions in Figure 21 and 22 with 3D bounding box labels, BEV-2D, front camera image, and LPC. We also show images from the dash camera, since some of the outdoor camera measurements are unreliable due to the adverse weather conditions.

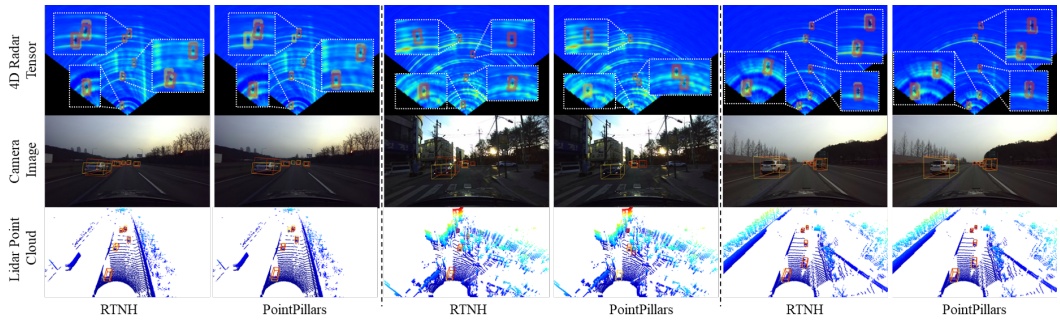


Figure 21: 3D object detection results of RTNH (4DRT) and PointPillars (LPC) in a road environment where multiple vehicles exist. We use yellow and red boxes to represent the ground truths and predictions, respectively.

Figure 21 shows the object detection results of RTNH and PointPillars for the road environments where multiple vehicles exist under weather conditions without precipitation (e.g., normal, overcast). As shown in Figure 21, RTNH produces similar or more robust detection results (robust to miss detection) compared to PointPillars. The comparisons summarized in Table 4 and Figure 21 show that 4D Radar has similar or more robust detection performance to Lidar in various road environments where multiple vehicles exist. This indicates that 4D Radar can be sufficiently used alone as a perception sensor in autonomous driving.

In addition, notice that the general AP for the normal condition can be lower than the overcast condition on K-Radar because of the following two reasons. One reason is that 4D Radar and Lidar are not affected by lighting condition, so that the detection performance of 4D Radar and Lidar for overcast condition cannot be lower than that for normal condition. Another reason is that the normal condition in K-Radar has more difficult situation than the overcast condition; the normal condition contains various situations including many vehicles parked along the side of alleyways, while the overcast condition in K-Radar does not have many vehicles on clear urban roads of two lanes, as shown in Table 5.

Figure 22 shows the object detection results of RTNH and PointPillars under weather conditions with precipitation (e.g., sleet, light snow, heavy snow). As mentioned in Section A.1, Lidar can produce reliable LPC measurements in rain and light snow condition, but not in sleet and heavy snow conditions, since the sensor surfaces are covered by frost or snow. This can be seen in the LPC in Figure 22, and also by comparing the PointPillars results in Table 4.

From the results of Figure 22 and Table 4, we demonstrate that 4D Radar is a more robust sensor than Lidar in the adverse weathers. We note that the inputs of RTNH and PointPillars are 4DRT and LPC, respectively. As mentioned in 4.3, we do not claim that RTNH is a better neural network

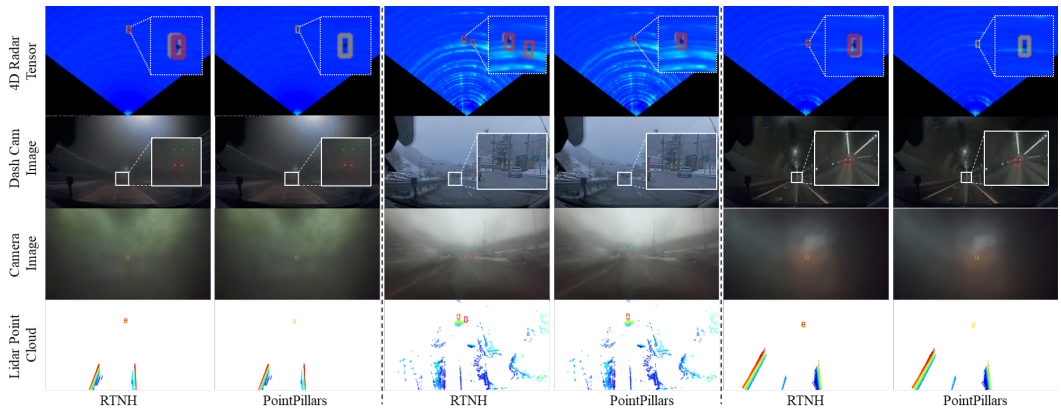


Figure 22: 3D object detection results of RTNH (4DRT) and Point Pillars (LPC) under weather conditions with precipitation. From the left, sleet, light snow, and heavy snow. We use yellow and red boxes to represent the ground truths and predictions, respectively.

architecture compared to PointPillars. Instead, we demonstrate the robustness of 4D Radar in all weather conditions including adverse weathers.

G Consideration of the K-Radar dataset as a pre-training dataset for other Radar tensor datasets

Pre-training on large scale datasets is well known to help neural networks converge faster (He et al., 2019). Therefore, we may consider using K-Radar as a pre-training dataset for other Radar tensor-based object detection datasets, or conversely, using other datasets as a pre-training dataset for K-Radar.

However, we want to note that the pre-training on K-Radar does not directly guarantee a strong improvement on RADIATE. This is because the characteristics of K-Radar and RADIATE are inherently different.

First, the power measurements in K-Radar and RADIATE have different distributions due to the different type of Radars used. When a neural network is trained on a dataset and applied to process target data of different distribution, there will be a poorly degraded performance in the target domain, as we see in Lidar object detection networks trained and evaluated on different type of point clouds (e.g., Velodyne and Ouster) (Wang et al., 2020).

Second, the resolution of RADIATE (0.175m) is higher than K-Radar (0.46m). This mismatch of resolution can also adversely affect the detection performance as usually seen in Lidar object detection networks trained on NuScenes (32-channels) and evaluated on KITTI (64-channels) (Wang et al., 2020).

Third, the data distributions are significantly different. K-Radar data is collected in South Korea where cars drive on the right, while RADIATE is collected in the U.K where cars drive on the left.

The above reasons apply to other Radar tensor-based datasets as well as RADIATE. For these reasons, it is difficult to expect performance improvement by using K-Radar as a pre-training dataset for other Radar tensor-based datasets and vice versa.

H Details of devkits

To facilitate the experiments on various neural network structures, we provide modularized neural network training codes that can manage each experiment with a single configuration file. We also provide GUI-based programs for visualization and neural network inference, as shown in Figure 23, to facilitate inference on large amounts of data. We provide a video clip on how to use the program,

which can be found through Section I URL 6, and all codes for devkits can be downloaded from Section I URL 1.

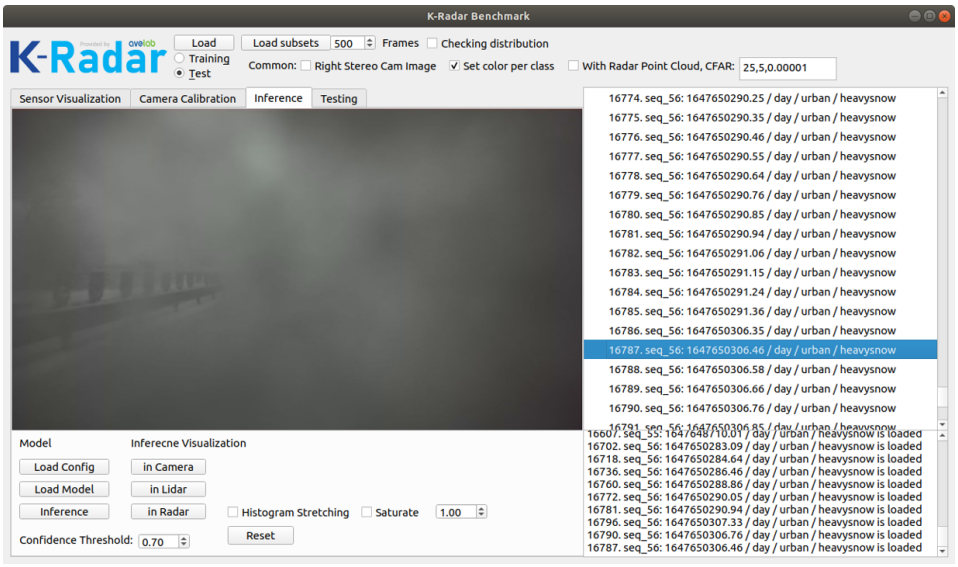


Figure 23: A snippet of the video clip that shows GUI-based program for visualization and neural network inference. (see Section I URL 6)

I Relevant URLs

(1) Publication of datasets and complete devkits code (learning, evaluation, reasoning, visualization, labeling programs): `https://github.com/kaist-avelab/K-Radar`

(2) The video clip showing each sensor measurement dynamically changing during driving under the heavy snow condition: `https://www.youtube.com/watch?v=TZh5i2eLp1k&t=103s`

(3) The video clip showing the 4DRT/LPC calibration and annotation process: `https://www.youtube.com/watch?v=ylG0USHCBpU&t=152s`

(4) The video clip showing the annotation process in the absence of LPC measurements of objects: `https://www.youtube.com/watch?v=ILlBJJpm4_4&t=8s`

(5) The video clip showing calibration results: `https://www.youtube.com/watch?v=U4qkaMSJOds&t=10s`

(6) The video clip showing the GUI-based program for visualization and neural network inference: `https://www.youtube.com/watch?v=MrFPvO1ZjTY&t=3s`

(7) The video clip showing the information regarding tracking for multiple objects on the roads: `https://www.youtube.com/watch?v=8mqxf58_ZAk`

References

Yue Zhou, Shengjie Niu, Jingjing Lü, and Yuehua Zhou. The effect of freezing drizzle, sleet and snow on microphysical characteristics of supercooled fog during the icing process in a mountainous area. *Atmosphere*, 7:143, 11 2016. doi: 10.3390/atmos7110143.

Gregor Luetzenburg, Aart Kroon, and Anders Bjørk. Evaluation of the apple iphone 12 pro lidar for an application in geosciences. *Scientific Reports*, 11, 11 2021.

Stanford Artificial Intelligence Laboratory et al. Robotic operating system. URL `https://www.ros.org`.

Alican Mertan, Damien Jade Duff, and Gozde Unal. Single image depth estimation: An overview. *Digital Signal Processing*, page 103441, 2022.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017a.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

Martin Simony, Stefan Milzy, Karl Amendey, and Horst-Michael Gross. Complex-yolo: An euler-region-proposal for real-time 3d object detection on point clouds. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017b.

Baoyuan Liu, Min Wang, Hassan Foroosh, Marshall Tappen, and Marianna Penksy. Sparse convolutional neural networks. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 806–814, 2015. doi: 10.1109/CVPR.2015.7298681.

Benjamin Graham, Martin Engelcke, and Laurens Van Der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9224–9232, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European conference on computer vision (ECCV)*, pages 181–196, 2018.

Siyuan Qiao, Liang-Chieh Chen, and Alan Yuille. Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10213–10224, 2021.

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.

Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4918–4927, 2019.

Yan Wang, Xiangyu Chen, Yurong You, Li Erran Li, Bharath Hariharan, Mark Campbell, Kilian Q Weinberger, and Wei-Lun Chao. Train in germany, test in the usa: Making 3d object detectors generalize. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11713–11723, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] See Section 1.
   (b) Did you describe the limitations of your work? [Yes] See Section 5.1.
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] Our paper conforms to the ethics review guidelines.
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Section 3.4, 4.2, and 4.3.
   (b) Did you include complete proofs of all theoretical results? [Yes] See Section 4 and Appendix F.
3. If you ran experiments (e.g. for benchmarks)...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See Appendix H and I.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 3.2 and 4.1.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 4.1.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A] We only use our proposed dataset, and cite the corresponding paper for the existing neural networks.
   (b) Did you mention the license of the assets? [Yes] See Appendix A. We mention the license of our proposed dataset.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] See Appendix I.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]