# A Reinforcement Learning Attention Agent for Lidar-based 3D Object Detector

**Shuqing Zeng** [*1] **Jordan Chipka** [*1] **Yasen Hu** [1]

## Abstract

This paper presents a computationally efficient selective attention strategy for a lidar-based 3D object detector that takes inspiration from human fovea using a mixed-resolution scheme: a lower resolution for peripheral and a higher resolution for a mission-critical region. An end-to-end reinforcement learning agent is developed to generate human-like "attention" such that a sweet-spot is achieved by trading off efficiency (low computing power) and effectiveness (high recall rate for the detector). The proposed attention implementation is self-contained, with the lidar point cloud as the only input. The two proposed policies: deep Q network (DQN) and policy gradient (PG), are evaluated using the KITTI vision benchmark suite dataset. Results demonstrate that our approach either improves the detector's recall rate up to 20 percent under the same computing inference time or shortens the inference time up to 15 percent under the same recall rate compared with the baseline.

## 1. INTRODUCTION

Over the past years, we have seen a significant rise in autonomous driving (AD) research. An extensive sensor suite with high-definition (HD) capability is mounted on the AD vehicles to make robotic perception systems robust in sensing surrounding traffic situations. The HD sensors enable small object detection at a distance, and the wide FOV multi-modality integration in the same package reduces the plethora of sensors. However, the HD sensors introduce a bandwidth challenge for the downstream perception system. For example, an HD lidar of 3 million points per frame is equivalent to 40 times that of a Velodyne™ VLP-32, which indicates the object detection needs 40 times the computing capacity as that required by a Velodyne™ lidar if PointRCNN (Shi et al., 2019) is used. Figure 1 shows the characteristic curve of inference time vs. object detector's performance by sweeping the input lidar points' density from low (left) to high (right). The lower and further right, the better, which means we have fast execution and high accuracy. From the curve, one can see an increase of resolution significantly improves the detector's recall rate but at the cost of longer inference time. Another similar, yet less expensive, tactic is to use mixed-resolution to process the lidar point cloud at a downscaled resolution and a small region of interest (ROI) window (i.e., the red rectangle in Figure 2(a)) at a higher resolution. This approach mimics human foveal vision (Corbetta & Shulman, 2002; Almeida et al., 2017) and requires a smaller computing budget while allowing HD detections in the relevant critical regions to boost performance. Figure 2(b) shows that a sweet-spot of the trade-off between computing efficiency and detection effectiveness can be found. However, the approach may fall short if we do not ensure that the HD ROI is being taken from the most suitable area in the FOV of a lidar (e.g., places where small objects are located, etc.).

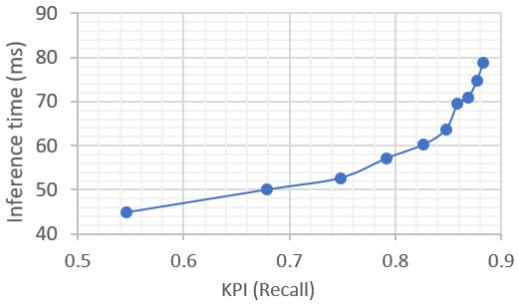
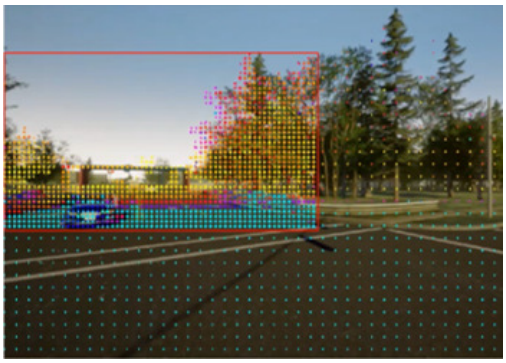



*Figure 1.* The curve of inference time vs. object detector's recall rate using KITTI dataset (Geiger et al., 2012). The curve is plotted by sweeping the input point cloud's subsampling rate from 0.1 to 1.0 with a step size of 0.1. Only 10% points are used in object detection in the leftmost of the curve, while the rightmost of the curve corresponds to 100% points used.

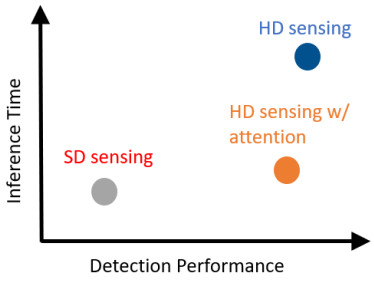
Attention techniques typically fall into two categories: bottom-up and top-down attention models. A bottom-up at-

[*]Equal contribution [1]General Motors Research and Development, 30565 William Durant Boulevard, Warren, Michigan, 48092-2031, USA. Correspondence to: Shuqing Zeng <shuqing.zeng@gm.com>, Jordan Chipka <jbc274@cornell.edu>.

*Proceedings of the 39$^{th}$ International Conference on Machine Learning*, Baltimore, Maryland, USA, PMLR 162, 2022. 



(a)




(b)

*Figure 2.* (a) The mixed-resolution scheme for the lidar sensor: a standard-definition (SD) resolution for background and a high-definition (HD) resolution for the most relevant region, a.k.a, attention. (b) The attention-enabled mixed-resolution scheme is a trade-off between computing efficiency and detection performance.

tention model uses features or characteristics from the scene to derive its attention. A classic illustration of bottom-up attention involves a human subject having their attention drawn to a single horizontal bar in a scene filled with vertical bars (Treisman & Gelade, 1980). Top-down attention models, on the other hand, are based on prior knowledge and are based on goals, expectations, and/or rewards (Yarbus, 2013).

Quickly and automatically detecting relevant areas of a scene through bottom-up attention models is an appealing capability for machine vision (Milanova & Mendi, 2012). Numerous studies in recent years have used bottom-up attention models for the tasks of object segmentation, object recognition, image captioning, and visual question answering (Chen et al., 2017; Xu et al., 2015; Singh et al., 2018; Chen et al., 2015). These methods typically use convolutional neural networks and/or recurrent neural networks to identify the attention region – which is not favorable for safety-critical systems with real-time constraints due to additional computing needs. Another common bottom-up approach is to identify salient regions in the image (Cheng et al., 2014; Anderson et al., 2018; Harel et al., 2007). However, for autonomous driving, salient regions are not necessarily the most relevant regions of the scene. Salient regions often identify the most conspicuous objects in the scene; however, these objects do not require HD sensing as they could also be detected with low-resolution sensing. Instead, relevant regions in a roadway scene can include small distant objects such as an upcoming traffic light or pedestrians crossing at a forthcoming crosswalk.

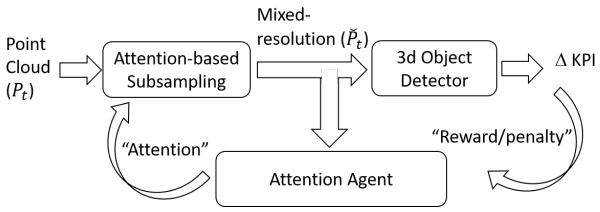



*Figure 3.* The proposed agent view of the attention-based sensing scheme. The attention agent is rewarded or penalized depending on whether the object detector's key performance index (KPI) improves.

Although the majority of attention models use a bottom-up approach, it is widely accepted that top-down factors play a crucial role in attention guidance (Henderson & Hollingworth, 1999). A recurrent attention model (RAM) and deep recurrent attention model (DRAM) were proposed to mimic human attention and have demonstrated promising results for the tasks of image classification and digit recognition (Mnih et al., 2014; Ba et al., 2014). Additional saliency-based techniques have also taken a top-down approach to derive attention and predicting human gaze (Hou et al., 2017; 2018; Xia et al., 2020). Finally, spatial transformers have been used as an attention mechanism for digit classification tasks due to their advantage of being fully differentiable (Jaderberg et al., 2015; Mendi & Milanova, 2010).

Taking inspiration from recent advances in deep reinforcement learning (RL) (Li, 2017), as well as aspects from both bottom-up and top-down attention models, we present a lightweight RL agent that intelligently places the ROI window in the most relevant area and demonstrate its benefit on the task of object detection on roadway scenes. This approach is a hybrid between bottom-up and top-down attention because it uses low-level features as its input, but is trained to achieve a high-level goal through the agent's reward function, shown in Figure 3. To the best of our knowledge, this may be the first study of applying attention mechanism to 3D lidar based object detection for the trade-off between computing efficiency and effectiveness (high detection rate).

As a summary, we list the novelties of the approach as

below:

- Attention behavior emerges based on the reward received from the external environment in an end-to-end manner.
- Self-contained, directly use the lidar's point cloud as the only input to determine the attention signal.
- The perception's (i.e., object detection) performance characteristics are explored via reinforcement training to place the ROI window to locations such that the overall detection performance is boosted.
- Hybrid between top-down and bottom-up attention.
- The scheme is simplistic and resilient, applicable to different sensor modalities (i.e., lidar point cloud, radar points, and camera image).
- An optimized technique to address the trade-off between computing efficiency and detection effectiveness.

## 2. METHODOLOGY

We present RL based attention technique in which the required computing resources are substantially reduced while the perception performance is retained. Figure 3 shows the agent view for the proposed approach in which human-like foveal behavior is emulated in a bi-directional process. In this scheme, the point cloud $P_t$ is captured from the lidar data stream and passed on to the attention-based subsampling mechanism. The subsampling mechanism takes the attention signal as input and downscales the full FOV frame to a background resolution while keeping the resolution inside the attention window intact. It then passes along the generated mixed-resolution data $\check{P}_t$ to the attention agent and the object detector. The attention agent then identifies the most relevant region inside the FOV and passes that back to the attention-based subsampling module in the form of an attention signal for the next time cycle. Finally, object detection is performed on the mixed-resolution point cloud $\check{P}_t$, and the objects are generated.

During the training phase, the final objects are compared with the ground truth to evaluate whether the quality of the generated attention signal has improved. The agent is rewarded if the detector's key performance index (KPI) improves. Otherwise, the agent is penalized.

### 2.1. RL-BASED ATTENTION AGENT

The first step to developing the RL-based attention agent is establishing its action space. The action of the attention agent $A_t^*$ is simply the selected location of the attention window's centroid (i.e., x and y locations in image coordinates) within the forward-facing camera frame. The lidar points are transformed to/from the camera's image plane to determine which points are within the attention region. To simplify the action space, we discretized the FOV into $N_x \times N_y$ blocks. Then, a fixed-size attention window $R_{N_x} \times R_{N_y}$ is determined based on the given action, $A_t^*$, taking discrete values from $(N_x - R_{N_x} + 1) \times (N_y - R_{N_y} + 1)$ blocks. Figure 4 depicts the discretized grid and the attention agent's action space within the forward-facing camera frame.

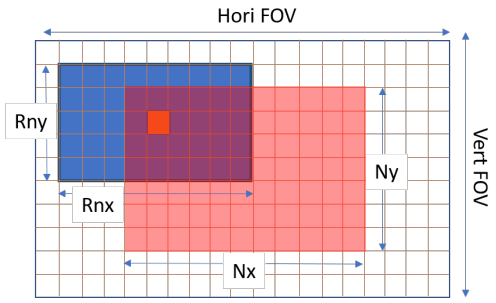



*Figure 4.* Depiction of the discretized grid, attention region (blue), the centroid of attention region (orange), and action space (red). The action space does not encompass the entire FOV to ensure the attention region does not ever extend beyond the FOV boundary.

The attention/perception module shown in Figure 5 is mainly comprised of a PointNet-based object detection (OD) network (Shi et al., 2019) and a deep neural network implemented the attention agent. Figure 5 shows a detailed view of how the attention and perception modules fit into the overall workflow. The attention and perception modules share the same backbone network to compute the semantic feature tensor $X_t$. Object detection is performed sequentially through the stages of region proposal generation, point cloud region pooling, and final 3d bounding box refinement and classification confidence regression. The output of the attention agent is either the anticipated reward or the probability distribution of each possible action $A_t$ from the action space $(N_x - R_{N_x} + 1) \times (N_y - R_{N_y} - 1)$ blocks in Figure 4. In DQN, the `argmax` of this matrix is then taken to obtain the attention signal, $A_t^*$. In policy gradient (PG), $A_t^*$ is the random draw from the probability distribution of the action space. Finally, based on this attention signal, the mixed-resolution point cloud $\check{P}_t$ is subsampled from the original point cloud $P_t$, passed to the OD network, and the final detected objects are outputted for the downstream functional modules.

The attention agent's architecture consists of two branches, which are fully connected neural networks comprised of one hidden layer. One branch is used to determine the attention window's x location, while the other branch is used to determine the attention window's y location. Figure 8 depicts the architecture of the DQN. The input to the DQN is the

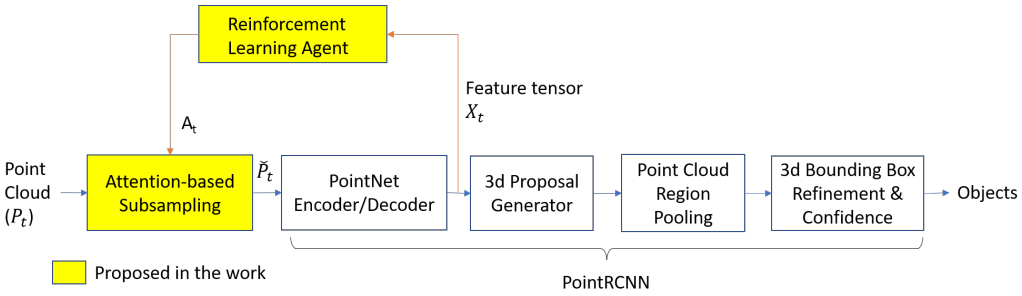



*Figure 5.* Detailed view of the attention/perception module and how it fits in the overall workflow. Note that the attention agent and object detectors share the backbone network providing the semantic feature $X_t$.

feature tensor from the object detection backbone. This feature tensor is then flattened and passed to two distinct, fully connected networks. The first network outputs an array with size $N_x$. This output's maximum value represents the predicted column for the attention window's centroid to maximize the reward. The second network outputs an array with size $N_y$. This output's maximum value represents the predicted row for the attention window's centroid, maximizing either the probability or the reward.

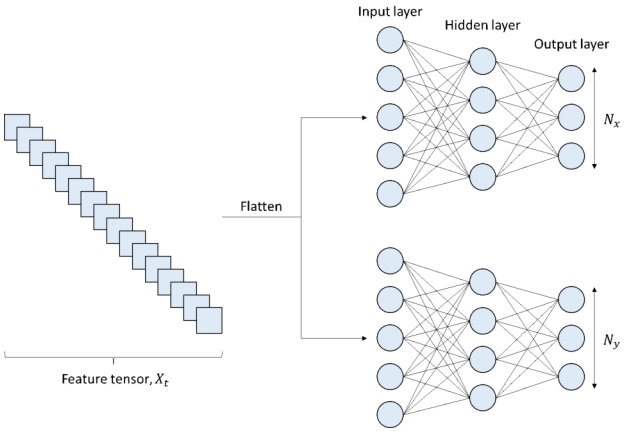



*Figure 6.* The reinforcement learning agent's network architecture consists of two fully connected branches – one to predict the attention window's x location, and the other to predict the attention window's y location.

One can note that a computationally efficient light-weight architecture implements the RL-attention agent whose add-on cost is negligible compared to the object detection pipeline in Figure 5.

## 2.2. TRAINING PROCESS

This work evaluates two reinforcement learning algorithms: deep Q network (DQN) and policy gradient network (PG) (Li, 2017). Despite DQN and PG's differences in training procedures, their inference network architecture is similar, as shown in Figure 6, with some minor differences in the final output layer. In DQN, the outcome is the anticipated reward of each possible action, while in PG, the outcome is the probability of each possible action.

### 2.2.1. DQN AGENT

We use standard techniques for training the DQN, but with a few minor adjustments. The DQN is trained to approximate a function that can predict Q-values, which are a measure of the agent's expected reward, given an action $A_t$, state $X_{t-1}$, and network parameters $\theta$. The policy which governs the agent's actions, $\pi_\theta(A_t, X_{t-1})$, is an epsilon greedy policy around the function $A_t^* = \text{argmax}_{A_t} Q(A_t, X_{t-1})$. This policy is used to improve training stability and increase the likelihood of convergence.

The DQN was trained using the point cloud from the KITTI dataset (Geiger et al., 2012). The point-cloud frames in the KITTI dataset are not sequential, and hence each frame could be considered independent from the next. Therefore, a replay memory was not needed to randomly sample data to ensure de-correlated batches. Also, as a result of temporally independent data, the discount factor, $\gamma$, was set to be 0. The discount factor is a constant between 0 and 1 that determines how the agent adjust future rewards – i.e., a discount factor of 0 will reward the agent simply based on the instantaneous reward. At the same time, a value closer to 1 will give more significance to expected rewards in the future. Since our DQN essentially operates on a frame-by-frame basis, our agent is only concerned with maximizing the reward for the current frame, and therefore, the discount factor is set to 0.

With a discount factor of 0 ($\gamma = 0$), the original training rule (as shown in Eq. 1) simplifies to Eq. 2.

$$Q^\pi(X, A) = r + \gamma Q^\pi(X', \pi(X')) \quad (1)$$

$$Q^\pi(X, A) = r \quad (2)$$

$$\delta = Q^\pi(X, A) - r \quad (3)$$

In the above equations, $Q^\pi(X, A)$ are the Q-values accord-

ing to policy $\pi$, given state $X$ and action $A$. The reward is represented as $r$, and the expected future state is $X'$. The difference between the two sides of the equality in Eq. 3 is the error, $\delta$, that we minimize during training. Since the DQN has two branches, one for the attention window's x position and the other for its y position, the network produces two sets of Q-values for each batch during training. Therefore, we get two error values, $\delta_x$ and $\delta_y$. Using an L1 loss function, we show the final training loss in Eq. 4, where $B$ is the batch size.

$$L = \sum_{i=0}^{B} |\delta_x| + |\delta_y| \tag{4}$$

A simple reward function is implemented to train the RL agent. This reward function compared the number of true positive detections before applying the attention window, TP, to the number of true positive detections after applying the attention window, TP′. If using the attention window resulted in more true positive detections, then the reward is +1. If using the attention window resulted in the same amount of true positive detections, the reward is 0. Finally, if using the attention window resulted in less true positive detections, the reward is -1. This reward function is summarized in Eq. 5 below.

$$r = \begin{cases} -1 & \text{if } \text{TP}' - \text{TP} < 0 \\ 0 & \text{if } \text{TP}' - \text{TP} = 0 \\ 1 & \text{if } \text{TP}' - \text{TP} > 0 \end{cases} \tag{5}$$

We design the reward function to maximize true positive detections of any object simply. The results show that the RL agent learned to seek small, distant vehicles using its attention ROI window. This behavior is due to vehicles being the dominant class in the KITTI dataset. However, a more sophisticated reward function can easily be tailored to give more significance to other classes of more interest. Furthermore, additional post-processing techniques could be implemented to provide more importance to other regions based on other available information sources, such as maps, path planning, etc.

Additional details regarding the training process, such as the DQN's behavior before and after convergence, can be found in Appendix.

#### 2.2.2. Policy Gradient (PG) Agent

Standard techniques are used for the PG agent. The agent predicts the probability distribution (i.e., multinomial distribution) of action $A_t$ over the agent's action space from the belief state $X_{t-1}$ of the previous time cycle using the function approximator (see Figure 5) to approximate $\pi_\theta\left(A_t|X_{t-1}\right)$ where $\theta$ is the parameters (weights) of the PG network.

## 3. RESULTS

### 3.1. EXPERIMENTAL SETUP

The RL-based attention mechanism was tested on both sequential and non-sequential data. For non-sequential data, only a single point cloud frame was used throughout the attention workflow. In other words, the attention window is applied to the same point cloud frame that provided the input feature tensor to the network architecture in Figure 5. However, for sequential data, this process spans two frames, so that the frame at time $t-1$ is used to generate the attention window that is applied to the frame at time $t$. $A_t$ is the action taken by the attention agent (i.e. the location of the attention window), $P_t$ is the input point cloud at time $t$, $\check{P}_t$ is the corresponding subsampled frame with mixed resolution (i.e., lower background resolution for peripheral and higher resolution in attention window), $X_t$ is the derived feature tensor, KPI′ and KPI are the key performance index with and without attention, and the reward $r_t$ is the delta between the two KPIs, i.e., $r_t = \text{KPI}' - \text{KPI}$.

The work uses the KITTI Velodyne™ dataset containing about 7500 independent frames for both the training and test datasets. We trained the DQN agent and PG agent using the frames in the training dataset. In this study, the discretized action space is defined as shown in Figure 6 where $N_x = 70$ and $N_y = 26$, and the fixed-size attention window size $R_{N_x} = 20$ and $R_{N_y} = 8$. Therefore the potential action takes a discrete value from the $51 \times 19$ red grid.

### 3.2. NON-SEQUENTIAL DATA

The following performance evaluation is reported using the separated test dataset. The experimental setup consists of comparing the PointRCNN vehicle detector's performance at four subsampling policies: baseline, random policy, DQN attention policy, and PG attention policy, shown in Figure 7. The baseline is simply a uni-resolution scheme where we subsample the input point cloud with an inclusion probability sweeping from 0.1 to 1 (all included). The random policy is another baseline where the mixed-resolution scheme is used, but the attention window's centroid is randomly placed in the lidar FOV. Figure 7 (c) and (d) show the result under DQN and PG attention policies, respectively.

The recall rate at IOU 50% is used as the performance metric for this study. Inference time was also used as a secondary performance metric. The performance curves are plotted in Figure 8 from left to right by sweeping the sampling probabilities from 0.1 to 1, respectively. Like in Figure 1, the lower and further right, the better, which means we have both fast execution and high accuracy. Figure 8 shows that the random policy is slightly better than the baseline since the mixed resolution provides a bit more points. The DQN and PG policies' performance is similar

| Baseline | | | | |
|---|---|---|---|---|
| Sampling probability | Points | Recall | FPS | Inference time |
| 0.1 | 1.6k pts | 0.496 | 22.3 | 44.84 |
| 0.2 | 3.3k pts | 0.629 | 20.0 | 50.00 |
| 0.3 | 4.9k pts | 0.698 | 19.0 | 52.63 |
| 0.4 | 6.6k pts | 0.742 | 17.5 | 57.14 |
| 0.5 | 8.1k pts | 0.777 | 16.6 | 60.24 |
| 0.6 | 9.8k pts | 0.798 | 15.7 | 63.69 |
| 0.7 | 11.5k pts | 0.809 | 14.4 | 69.44 |
| 0.8 | 13.1k pts | 0.819 | 14.1 | 70.92 |
| 0.9 | 14.7k pts | 0.827 | 13.4 | 74.63 |
| 1 | 16.4k pts | 0.833 | 12.7 | 78.74 |

(a) Baseline

| Random | | | | |
|---|---|---|---|---|
| Sampling probability | Points | Recall | FPS | Inference time |
| 0.1 | 3.2k pts | 0.554 | 21.4 | 46.73 |
| 0.2 | 4.7k pts | 0.662 | 19.9 | 50.25 |
| 0.3 | 6.2k pts | 0.721 | 18.7 | 53.48 |
| 0.4 | 7.6k pts | 0.757 | 17.5 | 57.14 |
| 0.5 | 9.1k pts | 0.786 | 16.8 | 59.52 |
| 0.6 | 10.5k pts | 0.803 | 15.9 | 62.89 |
| 0.7 | 12.0k pts | 0.818 | 14.9 | 67.11 |
| 0.8 | 13.5k pts | 0.822 | 14.4 | 69.44 |
| 0.9 | 14.9k pts | 0.828 | 13.8 | 72.46 |
| 1 | 16.4k pts | 0.833 | 13.2 | 75.76 |

(b) Random policy

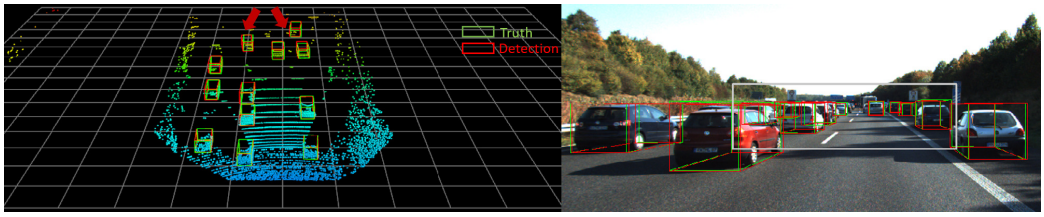
| Attention_DQN | | | | |
|---|---|---|---|---|
| Sampling probability | Points | Recall | FPS | Inference time |
| 0.01 | 2.7k pts | 0.569 | 21.5 | 46.51 |
| 0.05 | 3.2k pts | 0.701 | 20.7 | 48.31 |
| 0.1 | 3.9k pts | 0.756 | 20.2 | 49.50 |
| 0.2 | 5.3k pts | 0.790 | 18.8 | 53.19 |
| 0.3 | 6.7k pts | 0.806 | 17.7 | 56.50 |
| 0.4 | 8.1k pts | 0.818 | 16.7 | 59.88 |
| 0.5 | 9.5k pts | 0.822 | 16.1 | 62.11 |
| 0.6 | 10.8k pts | 0.824 | 15.2 | 65.79 |
| 0.7 | 12.2k pts | 0.831 | 14.4 | 69.44 |
| 0.8 | 13.6k pts | 0.831 | 13.9 | 71.94 |
| 0.9 | 15.0k pts | 0.833 | 13.4 | 74.63 |
| 1 | 16.4k pts | 0.833 | 12.8 | 78.13 |

(c) DQN attention policy

| Attention_Policy_Gradient | | | | |
|---|---|---|---|---|
| Sampling probability | Points | Recall | FPS | Inference time |
| 0.01 | 1.6k pts | 0.521 | 22.1 | 45.25 |
| 0.05 | 2.2k pts | 0.685 | 21.3 | 46.95 |
| 0.1 | 2.9k pts | 0.745 | 20.4 | 49.02 |
| 0.2 | 4.3k pts | 0.772 | 18.9 | 52.91 |
| 0.3 | 5.9k pts | 0.800 | 17.8 | 56.18 |
| 0.4 | 7.4k pts | 0.811 | 17.5 | 57.14 |
| 0.5 | 8.9k pts | 0.816 | 16.9 | 59.17 |
| 0.6 | 10.4k pts | 0.818 | 15.8 | 63.29 |
| 0.7 | 11.9k pts | 0.823 | 14.5 | 68.97 |
| 0.8 | 13.4k pts | 0.824 | 14.0 | 71.43 |
| 0.9 | 14.9k pts | 0.828 | 13.4 | 74.63 |
| 1 | 16.4k pts | 0.833 | 12.8 | 78.13 |

(d) PG attention policy


(e) Baseline




(f) DQN attention

*Figure 7.* (a)-(d) The vehicle detector performance vs. execution time by sweeping point density. Points denote the number of points sampled from the input point cloud. The detector performance is reported using the recall rate at IOU 50%. FPS is frames per second, and the unit of inference time is milliseconds. (e) Detected vehicles by the baseline without attention. (f) Detected vehicles using DQN attention policy. The gray rectangular box denotes the attention window where higher point density is kept.

but substantially superior to an agent with random policy or no attention case. Figure 8 demonstrates that our approach either improves the detector's recall rate up to 20 percent under the same inference time or shortens the inference time up to 15 percent under the same recall rate.

Figures 7(e)-(f) show an illustrative example. The left part of the image shows a perspective view of the point cloud. In the right part, the point cloud detections are overlaid in the registered forward-view camera image. Green and red bounding boxes denote the ground truth and actual detected vehicles, respectively. As shown in subfigure (e), we use 6.6k points as input to the baseline configuration, and red bounding boxes plot the detected vehicles after an inference

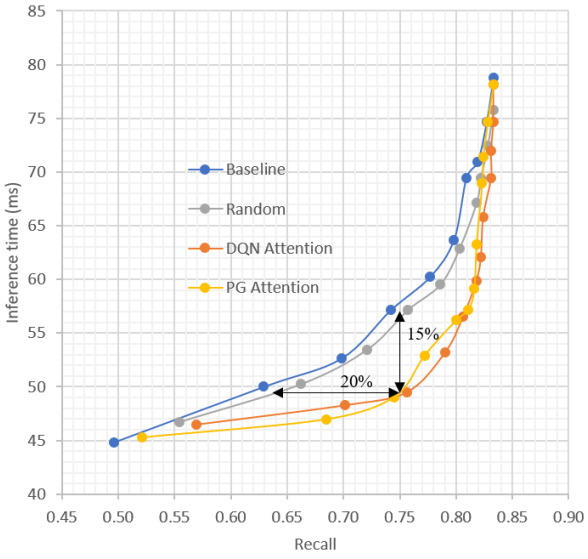



*Figure 8.* Performance curves: inference time of the four policies: baseline, random, DQN attention, and PG attention as a function of the detector performance (recall) for the KTTI dataset. The performance curves are plotted from left to right with an increasing point density.

time of 57 milliseconds. Notice that two remote vehicles are not detected, annotated by two bold red arrows. Meanwhile, in subfigure (f), the DQN attention policy is applied with a total of 5.3k points and 53-millisecond inference time, the two remote vehicles are reliably detected. This specific example illustrates that we can better detect vehicles by using fewer points than the baseline if the proposed attention policy is applied. Thus, the perception module runs faster.

### 3.3. SEQUENTIAL DATA

A similar test evaluation study was conducted on the following a sequential lidar dataset 2011_09_29_drive_0004. This drive was in the dataset's Road category and was on a rural 3-lane road with moderate traffic, no intersections. See Figure 9 for the results. In the sequential dataset, better performance boosts are observed than that of the non-sequential dataset. For example, in Figure 9, we can see either improving the detector's recall rate up to 36% under the same inference time or shortening the inference time up to 37% under the same recall rate.

The results demonstrate that the proposed attention mechanism can significantly boost detection performance while keeping the same processing time or reducing the processing time while maintaining the same detection performance, as shown in Figure 8 and Figure 9. This performance boost or computing efficiency is achieved by sampling the input point cloud using a mixed resolution. We expect the agent

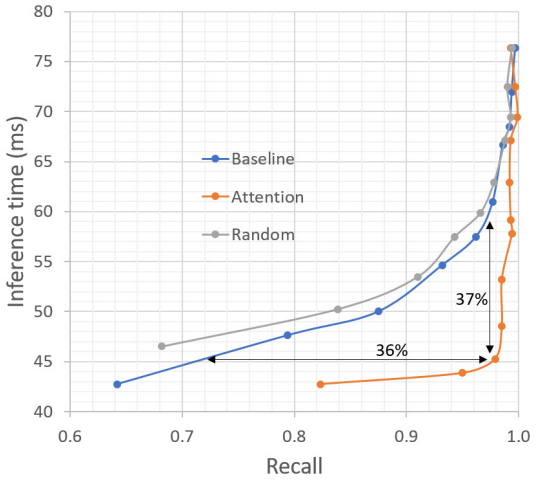



*Figure 9.* Performance curves: inference time of the three policies: baseline, random, and DQN attention as a function of the detector performance (recall).

to place the attention window to a region inside the FOV such that faraway vehicles are enclosed in the window.

(a)

(b)

*Figure 10.* Two snapshots of the attention window. The attention window, the gray rectangular box, is overlaid in the FOV of the forward-view camera.

Figure 10 shows two cases of where the attention windows are located. Note that the attention window is placed to regions such that the HD density will boost the detection performance. Please note that the attention window generation is self-contained; namely, the attention is derived from the attention agent's understanding of the scene from point cloud input. From the figure, the agent predicts the area enclosing small vehicles and places the attention window to cover the area since the agent has been rewarded most in the training.

Another perspective we need to discuss is the comparison between the random policy and the attention agent policies. Although the random case employs mixed-resolution and has the same input size as the proposed attention, Figure 8 and Figure 9 show that the random case's performance is similar to that of the baseline and significantly inferior to that of the attention agents.

## 4. CONCLUSIONS

Reinforcement learning (RL) was used to train a detector that process only relevant lidar points by emulating human attention: lower resolution for peripheral and higher HD resolution to cover critical areas (e.g., task-relevant and containing small objects). The evaluation results identified the sweet-spot of the trade-off between computing efficiency and detection effectiveness. Key contributions for the proposed attention module include:

- Hybrid between bottom-up and top-down attention
- Simple, resilient, emerged from the reward received in an end-to-end manner, and self-contained with input only from perception
- Reduced bandwidth requirement while not compromising perception performance
- Negligible add-on cost

## APPENDIX

Figure 11(a) shows the average loss for each epoch during training. The loss function is defined in Eq. 3 as the L1 norm and is descending along with the increasing epochs, and we expect these values to converge to the same number as training progresses. The value converges to around 0.3. Figure 11(b) shows the average Q value for each epoch during training.

At the beginning of training, the agent's behavior appears randomly. Figure 11(c) shows the Q-values of the attention window's locations for a given batch at the first training epoch. In this image, whiter regions indicate the actions with higher Q values that correspond to the attention region being placed in these locations more frequently – the whiter the region, the more frequently the attention window was placed there. As seen in the figure was relatively evenly dispersed around the lidar's field-of-view.

After convergence, the DQN agent's behavior appears much more consistent. Figure 11(d) shows the locations of the attention window for the same batch after convergence. This image shows that the attention window is fixated along the range image's horizon line. This behavior is sensible because small, distant objects are most likely to appear near the horizon line of the lidar's FOV. Furthermore, we see that the attention window most often seems to be slightly left-of-center. The DQN learns to place its attention window in that specific location due to the large number of small, distant vehicles that often appear in the oncoming traffic lane.

## References


Almeida, A. F., Figueiredo, R., Bernardino, A., and Santos-Victor, J. Deep networks for human visual attention: A hybrid model using foveal vision. In *Iberian Robotics conference*, pp. 117–128. Springer, 2017.

Anderson, P., He, X., Buehler, C., Teney, D., Johnson, M., Gould, S., and Zhang, L. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6077–6086, 2018.

Ba, J., Mnih, V., and Kavukcuoglu, K. Multiple object recognition with visual attention. *arXiv preprint arXiv:1412.7755*, 2014.

Chen, K., Wang, J., Chen, L.-C., Gao, H., Xu, W., and Nevatia, R. Abc-cnn: An attention based convolutional neural network for visual question answering. *arXiv preprint arXiv:1511.05960*, 2015.

Chen, L., Zhang, H., Xiao, J., Nie, L., Shao, J., Liu, W., and Chua, T.-S. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5659–5667, 2017.

Cheng, M.-M., Mitra, N. J., Huang, X., Torr, P. H., and Hu, S.-M. Global contrast based salient region detection. *IEEE transactions on pattern analysis and machine intelligence*, 37(3):569–582, 2014.

Corbetta, M. and Shulman, G. L. Control of goal-directed and stimulus-driven attention in the brain. *Nature reviews neuroscience*, 3(3):201–215, 2002.

Geiger, A., Lenz, P., and Urtasun, R. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. IEEE, 2012. doi: 10.1109/CVPR.2012.6248074.

Harel, J., Koch, C., and Perona, P. Graph-based visual saliency. 2007.

Henderson, J. M. and Hollingworth, A. High-level scene perception. *Annual review of psychology*, 50(1):243–271, 1999.

Hou, Q., Cheng, M.-M., Hu, X., Borji, A., Tu, Z., and Torr, P. H. Deeply supervised salient object detection with short connections. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3203–3212, 2017.

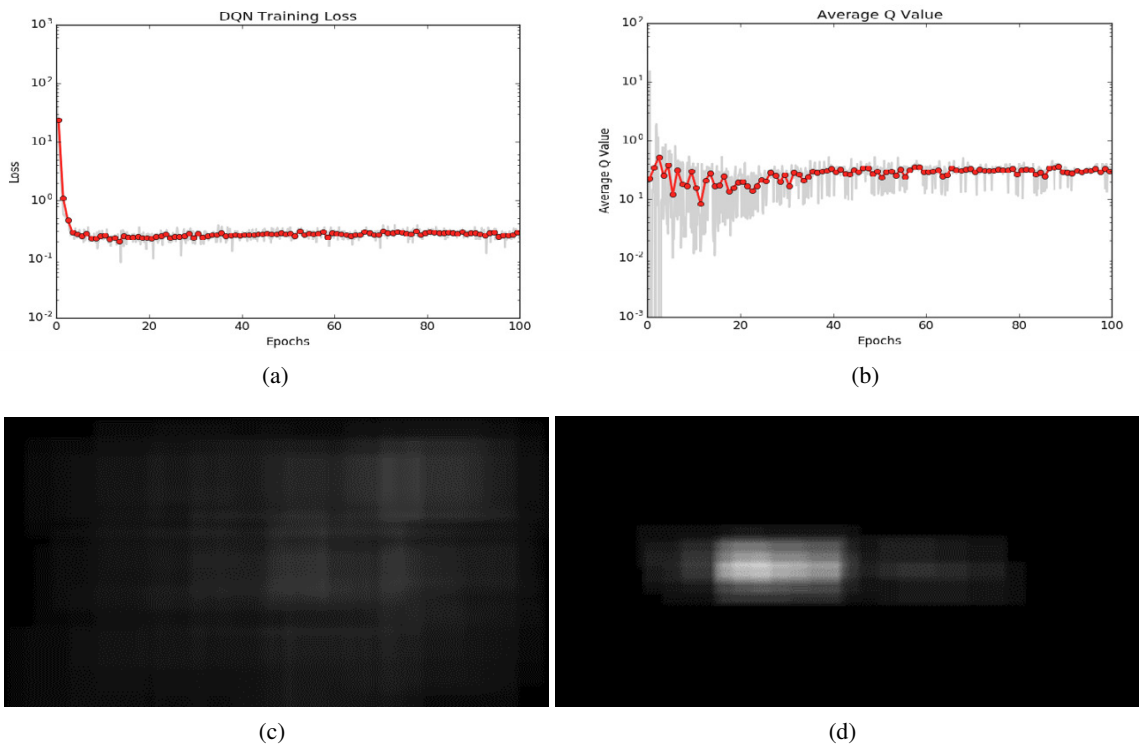



*Figure 11.* The learning curves of the DQN agent: a) The average loss ($\delta$ in Eq. 3) varies with training epochs, and b) The average Q value for each epoch during training. c) The image of the Q-values of the attention window's locations for a given batch at the first training epoch. The whiter the location, the higher the Q value for the agent to place the attention ROI at the location. d) The image of the Q-values for the same batch after convergence.


Hou, Q., Liu, J.-J., Cheng, M.-M., Borji, A., and Torr, P. H. Three birds one stone: A general architecture for salient object segmentation, edge detection and skeleton extraction. *arXiv preprint arXiv:1803.09860*, 2018.

Jaderberg, M., Simonyan, K., Zisserman, A., and Kavukcuoglu, K. Spatial transformer networks. *arXiv preprint arXiv:1506.02025*, 2015.

Li, Y. Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274*, 2017.

Mendi, E. and Milanova, M. Contour-based image segmentation using selective visual attention. *Journal of Software Engineering and Applications*, 3(8):796, 2010.

Milanova, M. and Mendi, E. Attention in image sequences: Biology, computational models, and applications. In *Advances in Reasoning-Based Image Processing Intelligent Systems*, pp. 147–170. Springer, 2012.

Mnih, V., Heess, N., Graves, A., and Kavukcuoglu, K. Recurrent models of visual attention. *arXiv preprint arXiv:1406.6247*, 2014.

Shi, S., Wang, X., and Li, H. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–779. IEEE, 2019. doi: 10.1109/CVPR.2019.00086.

Singh, J., Ying, V., and Nutkiewicz, A. Attention on attention: Architectures for visual question answering (vqa). *arXiv preprint arXiv:1803.07724*, 2018.

Treisman, A. M. and Gelade, G. A feature-integration theory of attention. *Cognitive psychology*, 12(1):97–136, 1980.

Xia, Y., Kim, J., Canny, J., Zipser, K., Canas-Bajo, T., and Whitney, D. Periphery-fovea multi-resolution driving model guided by human attention. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1767–1775, 2020.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., and Bengio, Y. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pp. 2048–2057. PMLR, 2015.

Yarbus, A. L. *Eye movements and vision*. Springer, 2013.