sensors

MDPI

Article

Double Q-Learning for Radiation Source Detection

Zheng Liu and Shiva Abbaszadeh *

Department of Nuclear, Plasma, and Radiological Engineering, University of Illinois at Urbana-Champaign, 104 S Wright St, Urbana, IL 61801, USA; zliu86@illinois.edu

* Correspondence: sabbasza@illinois.edu

Received: 17 January 2019 ; Accepted: 22 February 2019; Published: 24 February 2019

Abstract: Anomalous radiation source detection in urban environments is challenging due to the complex nature of background radiation. When a suspicious area is determined, a radiation survey is usually carried out to search for anomalous radiation sources. To locate the source with high accuracy and in a short time, different survey approaches have been studied such as scanning the area with fixed survey paths and data-driven approaches that update the survey path on the fly with newly acquired measurements. In this work, we propose reinforcement learning as a data-driven approach to conduct radiation detection tasks with no human intervention. A simulated radiation environment is constructed, and a convolutional neural network-based double Q-learning algorithm is built and tested for radiation source detection tasks. Simulation results show that the double Q-learning algorithm can reliably navigate the detector and reduce the searching time by at least 44% compared with traditional uniform search methods and gradient search methods.

Keywords: reinforcement learning; radiation detection; source searching

1. Introduction

In the field of homeland security, detecting anomalous radiation sources in urban environments is an important yet challenging task due to the complexity of urban radiation background. When a suspicious area is determined, a radiation survey is usually carried out to search for anomalous radiation sources. To deliver a comprehensive and efficient survey, different survey approaches have been studied such as manually scanning of the area by human operated detectors and automatically scanning the area with robots under the navigation of pre-defined survey paths [1,2]. However, neither of the manual scanning method and the survey path method can achieve flexibility and efficiency at the same time. For instance, manual scanning carried out by humans is flexible to adjust survey strategies using new acquired measurements, but it requires lots of human efforts and may be dangerous for human surveyors. The pre-defined survey path method does not require human efforts during survey, but it cannot use new information gained during the survey and thus is not flexible to adjust survey paths. To address those issues, this study proposes the reinforcement learning algorithm as a data-driven approach to navigate automated robots equipped with radiation detectors for radiation source detection tasks.

Radiation exists everywhere in daily life. It arises from naturally occurring radioactive materials (NORMs) that are present in air, soil, and building materials [3,4]. Those NORMs contribute to the background radiation of the environment and are treated as a uniform radiation source. Besides those NORMs, there may also exist anomalous radiation sources in the environment such as wrongly disposed radioactive medical wastes, illicit radioactive materials, or even nuclear weapons [5,6]. They usually have high concentration with small footprint and are treated as point sources in the environment. The anomalous radiation source detection task is to identify and localize the anomalous radiation sources in an area. There are two major approaches that are commonly used to estimate the

location and intensity of anomalous radiation sources from radiation measurements: the maximum likelihood estimation-based methods [7–10] and the Bayesian estimation-based methods [11–13]. In real applications, the radiation measurements are usually short in counts due to the limited detection time and the shielding of the radiation sources. The spectral comparison ratio method [14] and Gaussian process [15] have been studied to process the count-starved radiation measurements and suppress noise.

Besides those source parameter estimation studies, people also investigated into the optimized operations of detectors such as the placement of detectors and the optimized detection time. Klimenko et al., studied the problem of optimizing detection time for a predefined survey path using methods from the sequential testing theory [16]. Cortez et al., designed radiation surveys based on variances in acquired measurements and uncertainties in the radiation field [17]. Hutchinson et al., sequentially determined the detector's placement positions using the concept of maximum entropy sampling [18]. Lazna et al., proposed a circular path planning strategy to exploit the directional characteristics of detectors [19]. Ristic et al., designed the survey path incrementally by choosing detector positions and detection times maximizing the information gain in the Renyi divergence sense [20]. Besides those conventional heuristic- or information theory-based methods, neural network-based reinforcement learning provides another emerging tool to solve the radiation source detection task.

Reinforcement learning (RL) is one of the machine learning algorithms that studies the problem of how agents ought to take actions in a given environment such that a certain goal can be achieved or rewards can be maximized [21]. With the recent development of computation hardwares and deep neural networks (DNN), the combination of RL and DNN shows great success in handling complex tasks such as playing video games, driving vehicles, playing Go, and controlling robots, to name a few [22]. By properly defining the learning problem, agents are able to learn the optimal actions under complex environments without prior human knowledge. Q-learning is one of the RL algorithms that learns the optimal action policy for agents without requiring a model of the environment [23]. Mnih et al., combined DNN with Q-learning which greatly improved the representability of Q-learning approach [24]; in their paper, the deep Q-learning (DQN) algorithm was trained to play Atari games and achieved human-level performance. Furthermore, the double Q-learning algorithm was developed to solve DQN's issue of overestimating action values and achieved a more stable and reliable learning [25]. The above implement of Q-learning approaches show the potential to apply Q-learning to automated anomalous source detection tasks.

The major contribution of this paper is formulating the radiation source detection task as a reinforcement learning problem and constructing a double Q-learning algorithm to solve the radiation source detection task. In this study, a simulation environment is setup that an agent carrying a radiation detector (this agent with detector is called "the detector" in the remaining part of this paper for brevity) searches a predefined area. This area has background radiation and an anomalous radiation source. The goal of the detector is to find the source using as short time as possible. A convolutional neural network-based double Q-learning algorithm is built to navigate the detector looking for the radiation source. This paper is organized as follows: section two introduces the radiation source detection task, the simulation environment, and the double Q-learning algorithm for radiation source detection; section three presents training results of the algorithm and analyzes its performance in searching for radiation sources; section four summarizes this paper and discusses possible future steps.

2. Materials and Methods

2.1. Radiation Source Detection Task

According to the physics of radiation emission, radiation counts measured by radiation detectors during a unit time interval can be modeled by Poisson distribution:

$$P(k;\lambda) = \frac{e^{-\lambda}\lambda^{k}}{k!} \tag{1}$$

Here λ is the intensity of the radiation, and k is the radiation counts measured by the detector. λ is contributed by two sources: the background radiation with intensity b and the anomalous radiation source with intensity I.

$$\lambda = b + \frac{I}{d^2} * \mathbb{1}_{\{not\ blocked\}} \tag{2}$$

The d^2 in Equation (2) is the distance between the detector and the radiation source. It is from the geometric efficiency correction that the detected point source's radiation intensity is proportional to the point source's solid angle viewed from the detector's detection surface. This solid angle is further approximated to be squared inverse proportional to the distance (denoted by d) between the detector and the point radiation source. As shown in Figure 1, there may exist walls in the searching area that block the radiation source's signal from the detector. Thus, there is an indicator function in Equation (2) to differentiate the blocked/not blocked case.

In this work, a radiation detection task and its simulation environment was setup according to the radiation model above. As shown in Figure 1, a searching area was setup in the simulation environment. In this area, there existed an anomalous radiation source and a radiation detector. In real life, radiation detectors usually move in all directions with arbitrary step size; in the simulation environment, the detector's moving directions were limited to up, down, left, or right, and the step size was limited to 1 meter. Theses constrains simplified the movement control while kept a valid approximation to the real life scenario. With these constrains, the detector could only move along the dashed grids as shown in Figure 1. There might also have walls in the simulation environment that could totally block the source's radiation signal. In the radiation detection task, the detector aimed at finding the anomalous radiation source as soon as possible.

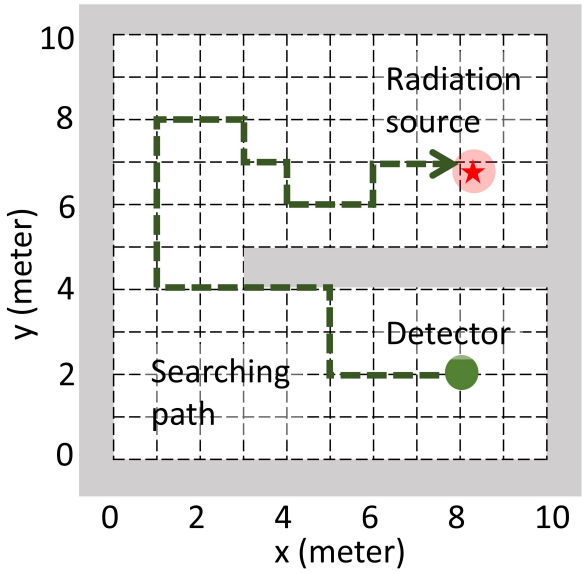

Figure 1. Illustration of the radiation detection task. The gray area denotes the searching area boundary and the walls inside the searching area. The green dashed line with arrow denotes a searching path of the detector.

2.2. Q-Learning with Convolutional Neural Network

In this work, we applied the double Q-learning approach [25] from reinforcement learning to train the detector searching for radiation sources. The radiation source detection task was formulated as a finite discrete Markov decision process (MDP), in which the radiation detector interacted with the

environment through a sequence of states (s), actions (a), and rewards (R). The goal of the detector was to move to the grid node closest to the source as soon as possible. This was achieved by training a convolutional neural network (CNN) to approximate the optimal action value function $Q^*(s,a)$

$$Q^*(s,a) = max_\pi E[\sum_{t'=0} \gamma^{t'} R_{t'} | s, a, \pi], \tag{3}$$

which is the maximum expected cumulative future reward starting from state s, action a, discounted by γ, maximized over action policy π, and accumulated for future time steps t' (see Appendix A).

As shown in Figure 2, the CNN takes the state s_t as input and outputs four different values (x_1, x_2, x_3, x_4) corresponding to four different actions (a_1=up, a_2=down, a_3=left, a_4=right). Each value represents the expected cumulative future reward starting from state s_t and taking the corresponding action, for example $x_i = Q(s_t, a_i), \forall i \in \{1,2,3,4\}$. Since the detector could only move along dashed grids (Figure 1), the state s_t was constructed by several matrices, which preserved the detector's searching history.

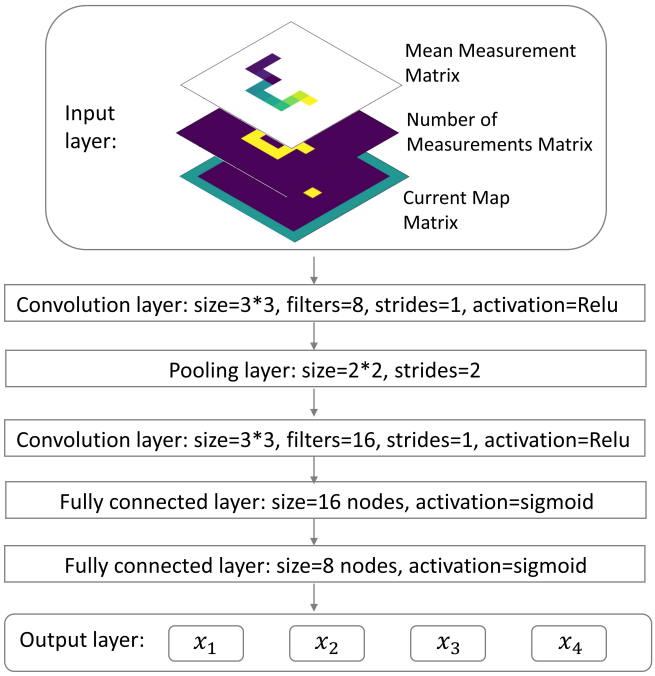

Figure 2. Schematic illustration of the convolutional neural network (CNN). The input of this CNN contains three images. The first image shows the mean radiation measurement for each of the visited positions. The second image shows the number of measurements for each position. The third image shows the detector's current position in the searching area. The output layer is a fully connected layer with 4 nodes, and each of the nodes represents its corresponding action's expected cumulative future reward given the input as current state. For the convolution layers or the pooling layer, the 'size' parameter specifies the size of the convolution or pooling kernel; the 'filters' parameter specifies the channel number of the convolution kernel; the 'strides' parameter specifies the stride of the sliding window for each dimension of input; the 'activation' parameter specifies the activation function applied to the output of the convolution results. For the pooling layer, we applied max pooling. For the fully connected layers, the 'size' parameter specifies the number of nodes in the layer, and the 'activation' parameter specifies the activation function applied to the output of the fully connected layer. The 'Relu' activation function is defined as $Relu(x) = max(x, 0)$, and the 'Sigmoid' activation function is defined as $Sigmoid(x) = \frac{e^x}{1+e^x}$. These definitions and names are consistent with the names used by the 'tf.layers.conv2d' function in Tensorflow [26].

If the simulation area is m meters wide and n meters long, a Mean Measurement Matrix of size $m \times n$ can be created so that the (i, j) element of this matrix represents the mean value of radiation measurements acquired at position (i, j). Similarly, a Number of Measurements Matrix of size $m \times n$ can be created so that the (i, j) element of this matrix stores how many measurements have been taken at position (i, j). A Current Map Matrix can also be created to record the position of the detector. Being different from previous two matrices, the Current Map Matrix has size $(m + 2) \times (n + 2)$ to incorporate boundaries of the simulation area. In this matrix, all boundaries and walls are denoted by 1, all accessible positions are denoted by 0, and the current position of the detector was denoted by 2. As shown in Figure 3, at each time step t these three matrices can be represented by three images. In our algorithm, the state s_t was constructed as a stacking of these three images: the Mean Measurement Matrix and the Number of Measurements Matrix were first padded by zeros to let them have size $(m + 2) \times (n + 2)$, and then these three images were stacked together.

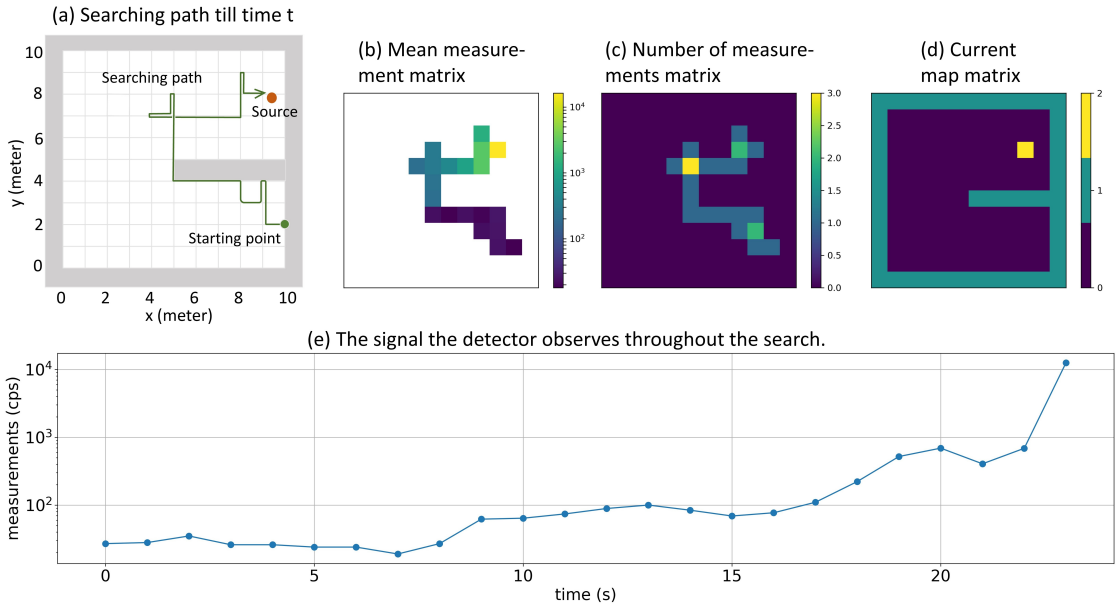

Figure 3. Representing the searching history using Mean measurement matrix, Number of measurements matrix, and Current map matrix. Plot (**a**) shows an example of a searching history. In plot (**b**), color represents the intensity of mean radiation measurements for each position. In plot (**c**), color represents the number of measurements been taken so far at each position. In plot (**d**), yellow represents the detector's current position, green represents boundaries and walls, and blue represents accessible positions. Plot (**e**) represents the signal that the detector observed throughout the search.

In our implementation, the simulated environment had an area of 10 m by 10 m, and we considered a typical hand-hold scintillation detector (a thallium activated cesium iodide detector with scintillator size $2 \times 1 \times 0.5$ inch). The average background radiation level of the UIUC campus measured by this detector was 25 cps (counts per second); thus, b was set to be 25 cps in Equation (2). Plot (e) of Figure 3 shows an example of the signal the detector observed in one search. When training the reinforcement learning algorithm, the radiation source should be strong enough so that the detector can learn how to search for the source. Thus, I in Equation (2) was uniformly sampled from (3000, 7000) cps to simulate different anomalous radiation sources with different intensities. The double Q-learning algorithm was trained by iteratively conducting a large number of episodes, in which the detector interacted with the environment until it reached a certain terminal state, and the environment was reset for the next episode. In an episode, the simulation environment was randomly initialized so that the wall's position, the radiation source's position and intensity, and the detector's position were properly defined. In each time step inside the episode, the radiation detector chose one direction to move and then collected one second radiation measurement. This episode would terminate if the

detector moves to the grid node closest to the source, or the total number of time steps is larger than a predefined time limit. In this study, the time limit was chosen to be 100 because it is the time needed to traverse the entire simulation area. The double Q-learning algorithm was trained for 1 million episodes. The reward in each time step was defined as follows:

$$R_t = \begin{cases} 0.5, & \text{if the detector moves closer to the source.} \\ -1.5, & \text{otherwise.} \end{cases} \tag{4}$$

Since there might exist walls, the Euclidean distance could not be directly used; the shortest-path distance was instead used to determine whether the detector moved closer or further to the source. This reward was set to be asymmetrical (0.5 vs. -1.5) to encourage the detector finding the source as soon as possible. The detailed training configuration is presented in Appendix B, and code is available upon request.

2.3. Evaluation

Three simulation experiments were carried out to evaluate the performance of the Q-learning algorithm. The first experiment was the radiation source searching test (Section 3.2), the second experiment was the detector trapping test (Section 3.3), and the third experiment was the radiation source estimation test (Section 3.4).

In the first experiment, the goal of the detector was to find the source as soon as possible. A successful case of finding a source was defined as the detector moving to the grid node closest to the source within 100 steps. For example, when source is at $(5.2, 7.8)$, the search is successful if the detector moves to the grid $(5, 8)$ within 100 steps. The metrics were the average searching time and the failure rate of finding the source. In this experiment, two different searching areas were tested as shown in Figure 4. One searching area does not have walls inside, and the other searching area has one wall inside. For both of the searching areas, the detector was always initialized at the lower left corner of the searching area. In each of the searching areas, we tested ten different source intensities from 50×2^0 cps to 50×2^9 cps. For each source intensity, we repeated the search 200 times with different random source positions. The performance of the Q-learning algorithm was compared with the gradient search algorithm (Appendix C).

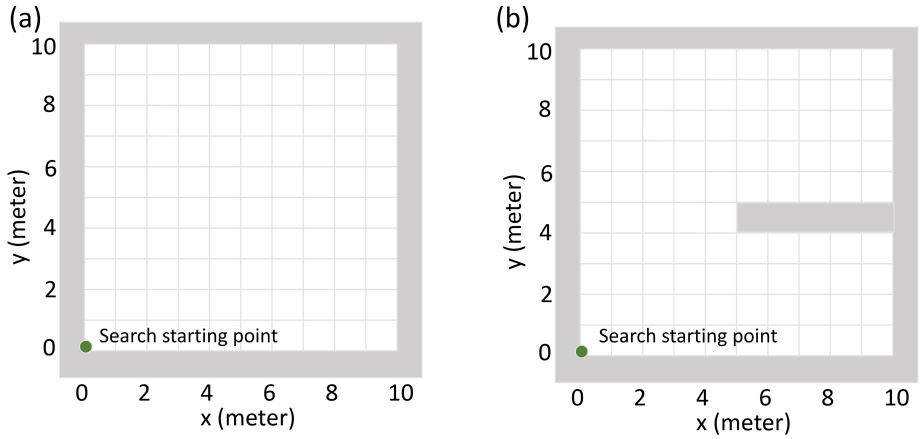

Figure 4. Illustration of the searching areas in the first experiment. (**a**) The searching area that does not have walls inside. (**b**) The searching area that has one wall inside. In both of the searching areas, the detector always starts searching from the lower left corner.

In the second experiment, we compared the detector trapping behavior between the Q-learning algorithm and the gradient search algorithm. As shown in Figure 5, there were two different kinds of simulation areas to be tested. In both of the areas, the detector started from the lower left corner,

and the source was placed at the upper left corner. In order to find the source, the detector needed to move from the lower part to the upper part of the simulation area. The plot (a) in Figure 5 shows the first kind of simulation area, in which the wall was attached to the left edge of the area and had various lengths (0 m, 2 m, 4 m, 6 m, 8 m). The longer the wall is, the more radiation signal it will block. All these wall configurations were already seen by the Q-learning algorithm in the training stage. The plot (b) in Figure 5 shows the second kind of simulation area. This wall configuration was not seen by the Q-learning algorithm in the training stage. For each of the wall configurations, 20 repeated tests were performed. For each test, we computed the relative trapped time, which was defined as the percentage of time the detector stayed in the lower part of the simulation area ($y \in [0, 4]$) during the whole searching process. The larger the relative trapped time is, the severer the detector is trapped.

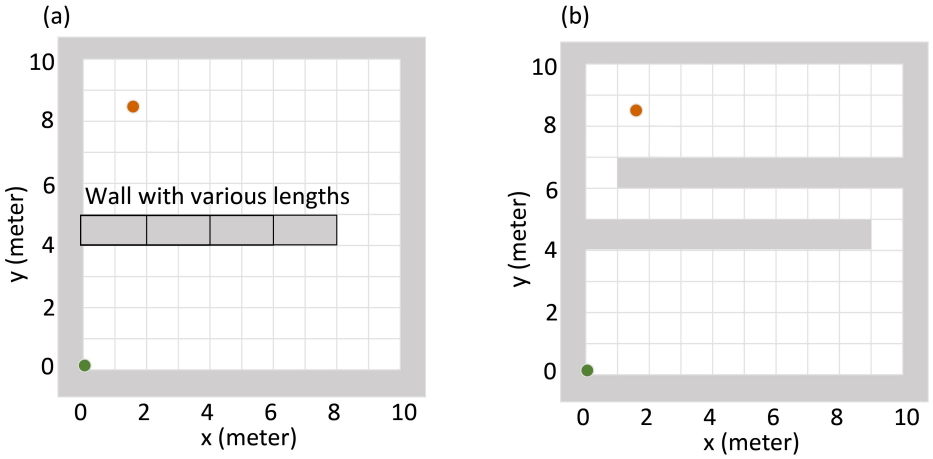

Figure 5. Illustration of the simulation areas in the detector trapping test. Green dots represent the initial point of the detector, and red dots represent the position of the radiation source. Grey areas represent the walls which can block the source's radiation entirely. Plot (**a**) shows the first kind of simulation area, in which the wall is attached to the left edge of the simulation area and has different lengths (0 m, 2 m, 4 m, 6 m, 8 m). The longer the wall is, the harder it is for the detector to move from lower part to the upper part. All these wall configurations have been seen by the Q-learning algorithm in the training stage. Plot (**b**) shows the second kind of the simulation area. This wall configuration has not been seen by the Q-learning algorithm in the training stage.

In the third experiment, the goal of the detector was to estimate the accurate location and intensity of the radiation source. Similar to the first experiment, the detector was guided by a navigation algorithm to look for the radiation source. The navigation algorithms under comparison were the Q learning algorithm, the gradient search algorithm, and the uniform search algorithm [27] with three different searching densities (see Appendix C). After the search, we estimated the radiation source's location and intensity using measurements collected so far. The closer the detector is to the source, the more informative radiation measurements can be acquired by the detector, and consequently a more accurate estimation about the source can be made. There are a variety of algorithms to estimate the radiation source's location and intensity [7–13]. For simplicity, we used the maximum likelihood estimation (MLE) approach to do the estimation. Suppose $\{(x_1, y_1, m_1), (x_2, y_2, m_2), \ldots, (x_t, y_t, m_t)\}$ are the measurements acquired till time t, the MLE estimator of source position (x_s^*, y_s^*) and source intensity μ_s^* is calculated as follows:

$$(x_s^*, y_s^*, \mu_s^*) = argmax_{(x_s, y_s, \mu_s)} \sum_{i=1}^{t} (m_i \times log(\lambda_i) - \lambda_i), \tag{5}$$

$$\lambda_i = b + \frac{\mu_s}{(x_i - x_s)^2 + (y_i - y_s)^2} * \mathbb{1}_{\{not\ blocked\}}. \tag{6}$$

In this experiment, the simulation area did not have walls inside, and nine different source positions were tested (Figure 6). The source intensity was set to be 500 cps. For each source position, 20 repeated tests were performed.

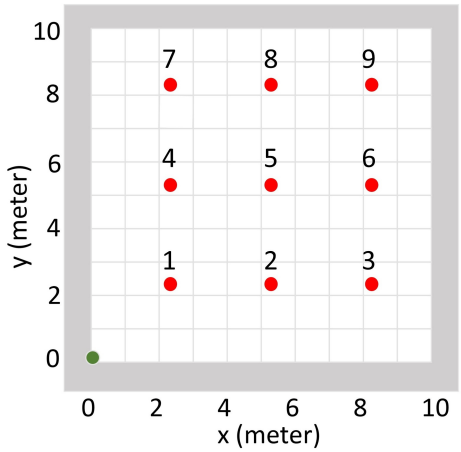

Figure 6. Experiment setup for the radiation source estimation test. The simulation area does not have walls inside. The green dot illustrates the starting point of the detector, and the red dots illustrate nine different testing source positions indexed from 1 to 9.

3. Results and Discussion

3.1. Training Result of the Double Q-Learning Algorithm

The training progress of the double Q-learning algorithm was monitored through a series of checkpoints. Between every 100 training episodes there was a checkpoint. In each checkpoint, 30 randomly initialized episodes were performed with CNN parameters fixed, and the average, minimum, and maximum reward in these episodes were recorded (Figure 7).

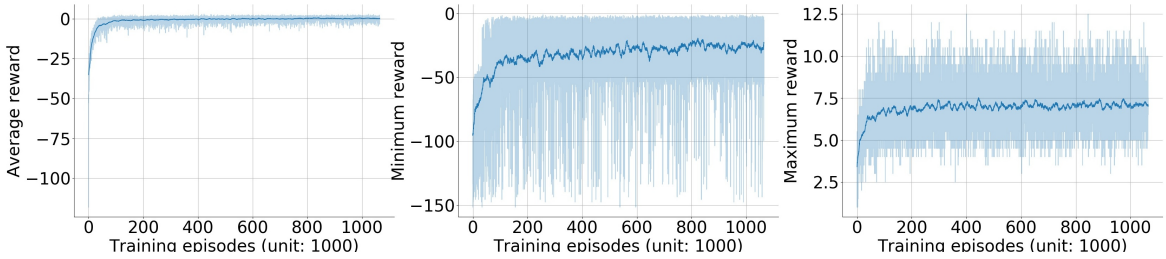

Figure 7. Training curves of the episode reward. One episode is one round of the searching task, starting from initializing the simulation environment and ending with the detector triggering the termination condition (either finding the source or reaching the maximum time step inside an episode). Between every 100 training episodes, 30 randomly initialized episodes were evaluated with CNN parameters fixed. Plot (**a**) shows the average testing reward of the 30 episodes. Plot (**b**) shows the minimum testing reward of the 30 episodes. Plot (**c**) shows the maximum testing reward of the 30 episodes. In these three plots, dark blue lines represent the averaged testing results with window size of 100 episodes, while light blue shadows represent raw testing results.

The average reward converged to near zero after 400 k episodes of training. Note that the per-step reward function was asymmetric (Equation (4)), an episode reward with zero mean meant that for every three correct actions (move closer to the source) the detector would take one wrong action (move further away from the source) on average. The minimum reward had majority between −50 and 0 with mean value converging to −25. It delivered a much bigger fluctuation than the mean

reward. This fluctuation was caused by rear scenarios that were not learned well by the algorithm. The maximum reward converged to 7 with much smaller fluctuation than the minimum reward.

3.2. Radiation Source Searching Test

To further analyze the Q-learning algorithm's performance in radiation source searching tasks, we selected the Q-learning model learned at the 842,500th episode and tested its performance under two different searching areas as shown in Figure 4. This model was selected because it achieved the biggest minimum reward during the training process (the middle plot of Figure 7). Metrics under evaluation were the average searching time and the failure rate of finding a radiation source. The Q-learning model was also compared with the gradient search approach.

As shown in Figure 8, the Q-learning algorithm outperformed the gradient search method in both simulation areas with walls and without walls, in all tested source intensities. Because walls could block radiation and introduce obstacles in detector's movements, adding walls made the searching time longer for both of the two algorithms. For simulation areas without walls, two algorithms' searching times reduced when the source intensity increased, and the Q-learning algorithm spent at least 25% less searching time than the gradient search method. For simulation areas with walls, the two algorithm's searching times did not depend on the source intensity, and the searching time of the Q-learning algorithm was 50% less than the searching time of the gradient search method.

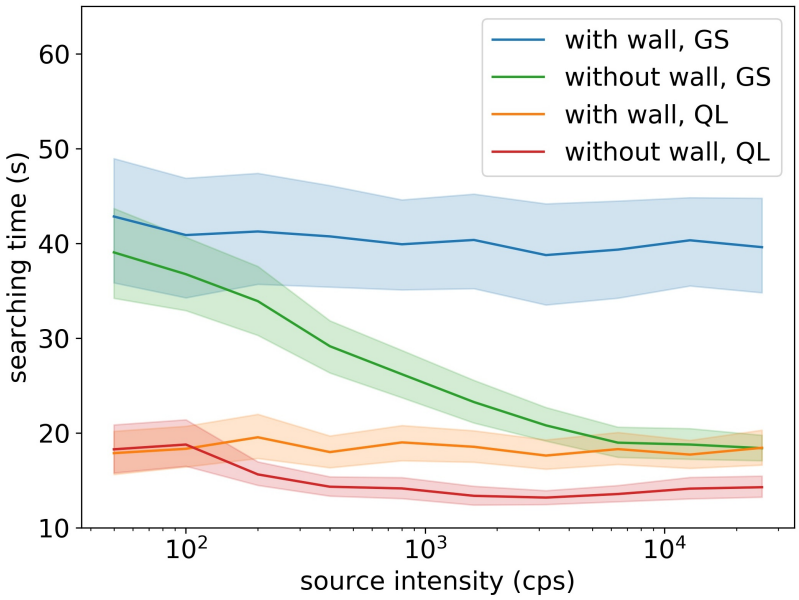

Figure 8. Average searching time under different source intensities. The shaded area is the 95% confidence interval estimated by bootstrapping. The 'GS' stands for the gradient search method, and the 'QL' stands for the Q-learning method. The Q-learning method used less searching times than the gradient search method in all simulation conditions.

Figure 9 shows that the failure rate of the gradient search algorithm and the Q-learning algorithm significantly dropped when the source intensity increased. Without walls, the failure rate of the two algorithms were the same and converged to below 2% for sources stronger than 800 cps. With walls, the failure rate of the Q-learning algorithm slightly increased and converged to 5% for sources stronger than 800 cps, but the gradient search method's failure rates were all above 30%. The gradient search method relied solely on radiation gradient to search for sources. If the detector and the source were in different sides of the wall, there would be no radiation gradient in the detector side, and the detector would just randomly move until it moved to the other side of the wall. This behavior significantly reduced the performance of the gradient search method, especially when the searching area had

obstacles. On the contrary, the Q-learning method was able to design its optimal searching path based on the obstacles in the searching area. If the detector found the source did not exist in this side of the wall, it would take the shortest path to the other side of the wall and search for sources. This explained why the Q-learning algorithm performed equally well in areas with walls and without walls.

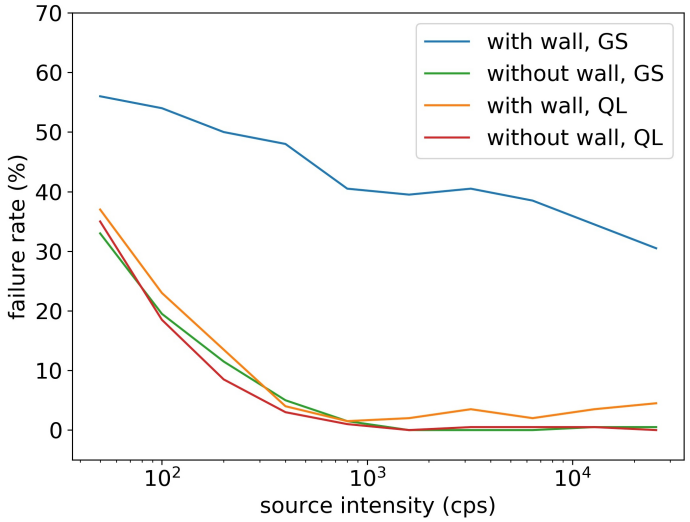

Figure 9. Failure rate of the Q-learning algorithm (QL) and the gradient search algorithm (GS) in the radiation source searching test, for simulation areas with walls and without walls. Without walls, the two algorithms had the same failure rate; with walls, the Q-learning algorithm achieved much smaller failure rate than the gradient search method.

3.3. Detector Trapping Test

The previous experiment reveals the Q-learning algorithm's ability to avoid being trapped by the walls. In this experiment, we further analyzed the detector trapping issue of the Q-learning algorithm and the gradient search algorithm on two different simulation areas shown in Figure 5.

Figure 10 shows the average relative trapped times (curves) and searching path examples (drawings) for the Q-learning algorithm and the gradient search algorithm in simulation areas illustrated by plot (a) of Figure 5. The Q-learning model under evaluation was taken from the 842,500th episode. As the wall length increased from 0 m to 8 m, the relative trapped time of the gradient search method increased from 0.5 to 0.9. From the searching path examples we can find that the gradient search method was easily trapped by long walls and wasted most of the searching time in the wrong side of the wall. When there was no wall, the Q-learning algorithm's relative trapped time was 0.27. When there were walls, its relative trapped times increased to 0.46 and were independent of the wall length. This demonstrates that the Q-learning algorithm was not trapped by walls. Example searching paths from the Q-learning algorithm show efficient searching strategies.

In the previous detector trapping test, one possible reason for the Q-learning algorithm not being trapped by walls is that the tested simulation areas are similar to the ones that have already been seen by the Q-learning algorithm during the training stage. In order to evaluate the detector trapping performance in new simulation areas, we tested the Q-learning algorithm and the gradient search method in the simulation area illustrated by plot (b) of Figure 5. This simulation area has different geometry compared to the ones being used to train the Q-learning algorithm. The overlapped two walls blocked all the source's radiation from the initialization point of the detector. In this test, the relative trapped time was defined as the percentage of the time the detector stayed in the lower 60% of the searching area. The gradient search algorithm's mean relative trapped time was 0.999, and plot (a) of Figure 11 shows a failure search path. For this geometry, the detector in the lower part of the simulation area could not receive any signal from the radiation source in the upper part of the simulation area, and the gradient method would just randomly pick one direction to move since

it could not detect any meaningful radiation gradient. The gradient search algorithm was heavily trapped by this geometry, and almost all of its searches failed. The Q-learning model taken from the 842,500th episode had a mean relative trapped time of 1, which means that this Q-learning model was also entirely trapped by the new geometry. Plot (b) of Figure 11 shows a failure search path of this Q-learning model. We can see that this detector was trapped in the corner and did not search for the source at all. It is because the trained Q-learning algorithm was highly fitted to the training simulation areas and did not generalize well for the simulation area with new geometry. To teach the Q-learning algorithm how to search in this new geometry, we added this new geometry into the training stage and trained 6000 additional episodes starting from the 842,500-episode model. After this additional training, the new Q-learning algorithm was not trapped by walls any more and achieved a relative trapping time of 0.63. Plot (c) of Figure 11 shows a successful search path from the new Q-learning algorithm. In this search, the algorithm firstly realized that there was no source in the lower part of the area; then, it took the shortest path to the upper part of the area and finally found the source there.

This experiment demonstrates that the Q-learning algorithm is able to efficiently design its searching path based on searching area geometries and recent radiation measurements. For learned search area geometries, the Q-learning algorithm will not be trapped and can find the source efficiently; for new search area geometries, the Q-learning algorithm will fail. This issue can be solved by adding additional training episodes for the new search area geometries. In our experiment, the 6000 additional training episodes took less than 30 min on a Tesla K80 GPU. In real applications, the searching area geometry is usually available in advance (such as through satellite pictures from Google Maps), and the Q-learning model can be easily updated according to the target searching area geometry in short time.

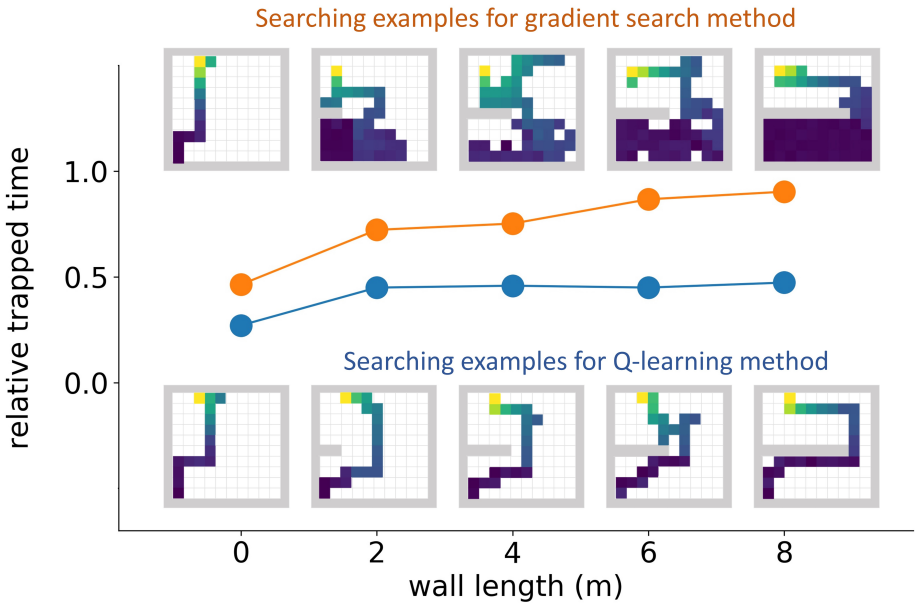

Figure 10. Detector trapping test 1. In this trapping test, simulation areas with different wall lengths were tested. The relative trapped time was defined as the percentage of the time the detector stayed in the lower 40% part of the simulation area during the whole searching process. The orange curve represents the mean relative trapped time of the gradient search method, and the blue curve represents the mean relative trapped time of the Q-learning method. The example searching paths for both algorithms in different wall lengths are also drawn in the figure. For all the searching path examples, the detector initialization point was at the lower left corner, the radiation source was at the upper left corner, and the color of the path represents the mean radiation intensity (blue is low, and yellow is high). The Q-learning algorithm was not trapped by walls, while the gradient search method was trapped by walls.

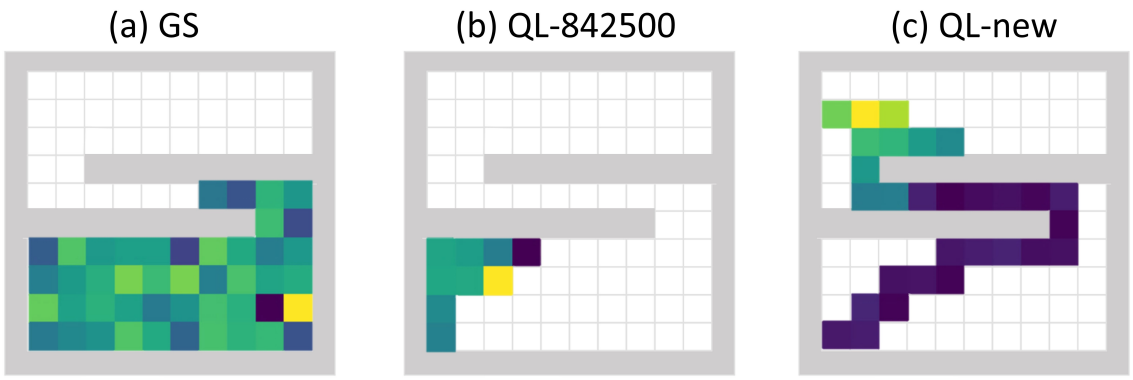

Figure 11. Detector trapping test 2. In this trapping test, a simulation area with two overlapped long walls was tested. The relative trapped time was defined as the percentage of the time the detector stayed in the lower 60% part of the simulation area during the whole searching process. For all the searching path examples, the detector initialization point was at the lower left corner, the radiation source was at the upper left corner, and the color of the path represents the mean radiation intensity (blue is low, and yellow is high). Plot (**a**) shows an example search path from the gradient search method that was failed to find the source. Plot (**b**) shows an example search path from the Q-learning model taken from the 842,500th episode, and this search was also failed to find the source. Plot (**c**) shows an example search path from the Q-learning model that was trained for additional 6000 episodes on the new simulation area. After this additional training, the new Q-learning model was able to efficiently search for sources in the new geometry.

3.4. Radiation Source Estimation Test

In this experiment, we tested the Q-learning algorithm, the uniform search algorithm, and the gradient search algorithm's performance of estimating the source's position and intensity. Similar to the previous section, the Q-learning model under evaluation was taken from the 842,500th episode.

Table 1 shows the mean estimation error for different searching methods. The simulation area was 10 m by 10 m. All of the five searching methods achieved source localization errors less than 15 cm (relative errors less than 1.5%) and source intensity estimation errors less than 7%. Averaging over all the 9 source positions shown in Figure 6, the mean searching time was 15.82 s for the Q-learning algorithm and 33.63 s for the gradient search algorithm. The Q-learning approach was 50% quicker than the gradient search algorithm and at least 44% quicker than the uniform search algorithms. Compared to the gradient search method, the Q-learning algorithm reduced the localization error by 35%, obtained similar source intensity estimation error, and used less searching time. Compared to the uniform search 1 approach, the Q-learning approach used less detection time and achieved a smaller localization error. The trade-off was a slightly bigger intensity estimation error (6.3% vs. 5%). The uniform search 2 and 3 approaches outperformed the Q-learning method, but they used much longer searching time and collected much more measurements. For source searching tasks with limited searching time, the Q-learning method would be a competitive alternative method compared to the uniform search method and the gradient search method, since it significantly reduced the searching time while maintained a low estimation error.

Table 1. Radiation source estimation test.

Approach *	Mean Estimation Error			Mean Searching Time (s) **								
	$x_s(m)$	$y_s(m)$	$\mu_s(\%)$	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9
QL	0.057	0.064	6.3	7.85	16.70	22.35	6.90	15.70	17.75	15.65	16.75	22.70
GS	0.090	0.101	6.5	11.2	17.2	31.3	17.85	21.15	52.1	38.8	37.0	76.05
U1	0.082	0.134	5.0					28				
U2	0.024	0.026	3.1					37				
U3	0.013	0.011	1.7					54				

* QL: Q-learning; GS: gradient search; U1: uniform search 1; U2: uniform search 2; U3: uniform search 3 (Appendix C.2); ** $t_i (i \in [1,9])$ represents the mean searching time for the i'th source shown in Figure 6.

4. Conclusions and Future Works

In this paper, the radiation source detection task was formulated as a reinforcement learning problem and a CNN-based double Q-learning algorithm was developed to navigate the detector searching for the radiation source. Simulation environments were set up to test the algorithm's performance. In the radiation source searching test, the Q-learning method used at least 25% less searching time and achieved a lower failure rate than the gradient search algorithm. In the detector trapping test, the Q-learning algorithm was less prone to the detector trapping issue than the gradient search method. In the radiation source estimation test, all of the uniform search methods, the gradient search method, and the Q-learning method achieved relative localization error lower than 1.5% and relative intensity estimation error lower than 7%, but the Q-learning approach reduced the mean searching time by at least 44% compared to other methods.

In the future, we are going to implement this Q-learning algorithm on a drone-based radiation detection platform, and field tests will be carried out. In real applications such as nuclear security and nuclear decommissioning, the detection platform's computation power and energy supply are usually limited. For security concerns, the detection platform may need to be isolated from Internet connection. These limitations require the navigation algorithm to run locally on light-weight mobile computation devices such as mobile phones. The Q-learning algorithm's computation burden is mostly from the training stage, which could be done in other powerful computers. Once the training is finished, the algorithm can be deployed efficiently on light-weight mobile devices for navigation.

The current implementation of the double Q-learning algorithm has fixed detection time (1 s for each time step), fixed step length (1 m for each action), and limited moving angles (4 directions). In the future, we will explore the finer controlling of the detector such as adding detection time as another controllable parameter, accepting different step lengths, and supporting more moving angles. The current Q-learning algorithm was designed and trained based on the one-source searching scenario; consequently, it could only handle one-source searching tasks. However, it has the potential to be applied into multiple-source searching tasks. When more than one source present in the environment, the current algorithm can still move towards one of the sources, but it does not know what to do after finding the first source. In order to search for multiple sources, the Q-learning algorithm needs to remember the found sources and use their positions and intensities to adjust new radiation measurements.

Author Contributions: Conceptualization, Z.L. and S.A.; methodology, Z.L.; software, Z.L.; validation, Z.L.; formal analysis, Z.L.; investigation, Z.L. and S.A.; resources, S.A.; data curation, Z.L.; writing—original draft preparation, Z.L.; writing—review and editing, Z.L. and S.A.; visualization, Z.L.; supervision, S.A.; project administration, S.A.; funding acquisition, S.A.

Funding: This study was supported by the Defense Threat Reduction Agency under the grant HDTRA 1-14-1-0011, and by the Department of Energy National Nuclear Security Administration under Award Number(s) DE-NA0002576 through the Consortium for Nonproliferation Enabling Capabilities.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Q-Learning and Radiation Source Detection

In this section, the double Q-learning algorithm for radiation source detection task is described in details. We firstly describe the radiation source detection model, then introduce the Bellman equation in Q-learning, and finally explain how a convolution neural network (CNN) is constructed and trained according to the Bellman equation.

Appendix A.1. Radiation Source Detection Model

The setup of radiation source detection task is described in Section 2.1: the detector aims to find the source within the given search-time limit T. Because we also assume that the detector will move one step in each second, this search-time limit T is also the upper limit for the number of steps the detector can take to look for the source. At time t ($0 \leq t \leq T$), the detector acquires one radiation measurement k_t at position (x_t, y_t), takes one action a_t, and receives one reward R_t. The measurement k_t is sampled based on the Poisson model being described in Equations (1) and (2), the action a_t is chosen from the action set {move up, move down, move left, move right}, and the step reward R_t is determined by Equation (4). The state of the detector at time t is denoted by s_t, which is constructed by all the historical measurements and their positions up to time t: $s_t = f((x_0, y_0, k_0), \ldots, (x_t, y_t, k_t))$. In this paper, we construct the state s_t as a collection of three different matrices: the mean measurements matrix, the number of measurements matrix, and the current map matrix. Those matrices are represented as figures in the input layer of the CNN shown in Figure 2. With this construction, we can easily verify the Markov property of the states:

$$P(s_{t+1}|a_t, s_t, s_{t-1}, \ldots, s_0) = P(s_{t+1}|a_t, s_t) \tag{A1}$$

Appendix A.2. Bellman Equation in Q-Learning

In this paper, the Q-learning algorithm aims to find the radiation source as soon as possible. This is achieved by learning an action strategy to maximize a cumulative future reward:

$$G_t = \sum_{t'=0}^{T-t} \gamma^{t'} R_{t+t'}. \tag{A2}$$

γ is a discounting factor between zero and one that reduces the importance of far-future rewards, and G_t is the cumulative future reward starting at time t. In the radiation source detection task, the detector chooses its action a_t according to a probability-based policy π. $\pi(a_t|s_t)$ specifies the probability of the detector to take action a_t when it observes state s_t. With a certain policy π, the action-value function $Q_\pi(s, a)$, also known as Q-function, describes the expected cumulative future reward starting from state s, action a, and using policy π:

$$Q_\pi(s, a) = E[G_t|S_t = s, A_t = a, \pi] \tag{A3}$$

The optimal action-value function $Q^*(s, a)$ is obtained by applying an optimal policy so that the action-value function $Q_\pi(s, a)$ is maximized:

$$Q^*_\pi(s, a) = max_\pi E[G_t|S_t = s, A_t = a, \pi]. \tag{A4}$$

$Q^*(s, a)$ represents the expected cumulative future reward starting from state s and action a, and using an optimal policy. Bring Equation (A2) into Equation (A4), we will get

$$
\begin{aligned}
& Q^*(s,a) \\
&= max_\pi \, E_{\{(S_{t+1},A_{t+1}),(S_{t+2},A_{t+2}),\dots\}}[R_t + \gamma G_{t+1} | S_t = s, A_t = a, \pi] && \text{(A5)} \\
&= E_{S_{t+1}}[R_t | S_t = s, A_t = a] \\
&\quad + \gamma max_\pi E_{(S_{t+1},A_{t+1})}[\, E_{\{(S_{t+2},A_{t+2}),\dots\}}[G_{t+1} | S_{t+1}, A_{t+1}, \pi] \, | S_t = s, A_t = a, \pi] && \text{(A6)} \\
&= E_{S_{t+1}}[R_t | S_t = s, A_t = a] \\
&\quad + \gamma \, max_\pi \, E_{(S_{t+1},A_{t+1})}[\, Q^*(S_{t+1}, A_{t+1}) \, | S_t = s, A_t = a, \pi] && \text{(A7)} \\
&= E_{S_{t+1}}[R_t | S_t = s, A_t = a] \\
&\quad + \gamma \, E_{S_{t+1}}[\, max_{a'} \, Q^*(S_{t+1}, a') \, | S_t = s, A_t = a] && \text{(A8)}
\end{aligned}
$$

Finally, we obtain the Bellman equation [28] from the last line of above equation:

$$Q^*(s,a) = E_{S_{t+1}}[R_t + \gamma \, max_{a'} Q^*(S_{t+1}, a') \, | S_t = s, A_t = a] \tag{A9}$$

The intuition of the Bellman equation is that the optimal policy will always choose the action that can maximize the expected cumulative future reward for the next time step. According to this Bellman equation, Watkins and Dayan proposed the following iterative updating algorithm known as Q-learning and proved its convergence [23]:

$$Q(S_t, A_t) \leftarrow (1 - \alpha) \, Q(S_t, A_t) + \alpha [R_t + \gamma max_a Q(S_{t+1}, a) - Q(S_t, A_t)]. \tag{A10}$$

α is a parameter between 0 and 1 controlling the learning rate. The convergence of the above updating rule requires that all possible pairs of (a, s) are repeatedly visited during the training process.

Appendix A.3. Neural Network for Q-Learning

Usually, interesting problems have a large number of states and actions, and thus the updating rule (Equation (A10)) is difficult to implement. Instead, a function approximator $Q(s, a; \theta)$ can be learned according to the Bellman equation. In this study, we use CNN as the function approximator, and θ represents the parameters (weights) of the CNN. In the Deep Q Network (DQN) paper [24], the target value is defined as

$$y = R_t + \gamma \, max_a Q(S_{t+1}, a; \theta') \tag{A11}$$

The θ' is the parameter of the CNN from previous iterations and hold as constant. The loss function is defined according to the Bellman equation:

$$
\begin{aligned}
L(\theta) &= E_{s,a}[\, (Q(s,a;\theta) - E_{S_{t+1}}[y | S_t = s, A_t = a])^2 \,] && \text{(A12)} \\
&= E_{s,a,S_{t+1}}[\, (Q(s,a;\theta) - y)^2 \,] - E_{s,a}[Var_{S_{t+1}}(y | S_t = s, A_t = a)] && \text{(A13)}
\end{aligned}
$$

The second term in Equation (A13) represents the variance of the target value which does not depend on θ and is thus ignored. The gradient of the estimation loss with respect to θ is

$$\nabla_\theta L(\theta) = E_{s,a,S_{t+1}}[\, (Q(s,a;\theta) - y) \nabla_\theta Q(s,a;\theta) \,] \tag{A14}$$

At this point, θ can be learned using gradient descent approach:

$$\theta_{i+1} = \theta_i - \alpha \nabla_\theta L(\theta_i) \tag{A15}$$

Experience replay [24] is used to calculate the expectation in the gradient (Equation (A14)) during the training process. In every training step, training data (S_t, A_t, R_t, S_{t+1}) are stored in a memory buffer. For each iteration, samples are uniformly selected from the memory buffer, and the CNN is

updated through the batch stochastic gradient descent using those samples. This technique insures the independence of the training data between different training iterations and thus avoids oscillations or divergence in θ [24].

Double Q-learning [25] is used in this paper to further improve the performance. In DQN, Equation (A11) can be rewritten as following:

$$y = R_t + \gamma Q(S_{t+1}, argmax_a Q(S_{t+1}, a; \theta'); \theta') \tag{A16}$$

Because the action a is greedily selected from Q with parameter θ' and then evaluated using Q with the same parameter θ', it may lead to the overestimation of the action-value function. Double Q-learning approach addresses this issue by decoupling the selection from evaluation [25]:

$$y^{DoubleQ} = R_t + \gamma Q(S_{t+1}, argmax_a Q(S_{t+1}, a; \theta); \theta') \tag{A17}$$

Appendix B. Training Details

When training the model, we applied the ϵ-greedy [24] policy with ϵ reduced from 1 to 0.1 linearly over the first one million steps. During training, the loss calculated according to the mean squared-error (Equation (A12)) might be very big and have large gradients. This large gradients might make the training process unstable. To control the fluctuation of gradients and stabilize the training process, we replaced the original loss in Equation (A12) with the following Huber loss:

$$L(y, \hat{y}) = \begin{cases} \frac{(y-\hat{y})^2}{2}, & \text{if} |y - \hat{y}| \leq 1. \\ |y - \hat{y}| - \frac{1}{2}, & \text{otherwise.} \end{cases} \tag{A18}$$

The double Q-learning algorithm used in this study is shown as follows:

Algorithm A1 Double Q-learning with CNN.

Initialize reward function Q with weights θ
Initialize target reward function $\hat{Q}$ with weights $\hat{\theta}$
Initialize replay memory D
for episode = 1,M **do**
  Initialize states $\phi_1 = (s_1, n_1, m_1)$
  for t = 1,T **do**
    Select action a_t using ϵ-greedy policy based on Q
    Execute action a_t, obtain reward r_t and state s_{t+1}
    Update states $\phi_{t+1} = (s_{t+1}, n_{t+1}, m_{t+1})$
    Store transition $\{\phi_t, a_t, r_t, \phi_{t+1}\}$ in D
    Sample random minibatch of transitions $\{\phi_j, a_j, r_j, \phi_{j+1}\}$ from D
    Set

$$y_j = \begin{cases} r_j \text{ if episode terminates at step } j+1 \\ r_j + \gamma \hat{Q}(\phi_{j+1}, \underset{a}{argmax} Q(\phi_{j+1}, a)) \text{ otherwise} \end{cases}$$

    Perform a gradient descent step on HuberLoss$(y_j, Q(\phi_j, a_j; \theta))$ for θ
    Every C steps reset $\hat{Q} = Q$
    Terminate the episode if source is found
  end for
end for

s_t, n_t, and m_t represent the mean measurement matrix, number of measurements matrix, and current map matrix at step t respectively. The total number of training episodes was $M = 1 \times 10^6$, the step limit for each episode was $T = 100$, the discount factor was $\gamma = 0.9$, and the θ' was updated every 10 training steps.

Appendix C. Other Source Searching Algorithms

In this section, we introduce the gradient search algorithm and the uniform search algorithm.

Appendix C.1. Gradient Search Algorithm

The gradient search algorithm is a light-weighted algorithm that searches for sources by moving along the radiation gradient-rising direction. In this paper, the gradient of radiation was defined as $\frac{d\bar{m}}{dl}$. $d\bar{m}$ was the change of mean radiation measurement, and dl was the change of the measurement position. The gradient search algorithm calculated the radiation gradients of four possible moving directions (up, down, left, right) as (g_1, g_2, g_3, g_4). Then, these gradients were converted to a moving probability vector by the softmax function: $(p_1, p_2, p_3, p_4) = softmax(g_1/q, g_2/q, g_3/q, g_4/q)$. For example, p_1 represents the probability for the detector to move in the 'up' direction. q is a tunable parameter controlling how much randomness we want to add into the decision process. A large q leads to a more evenly distributed random choice over four moving directions, while a small q assigns more probability in the moving direction with the largest radiation gradient.

Since the radiation measurements are noisy, the calculated radiation gradients are not always pointing to the correct direction. Thus, it is not always preferred to move along the biggest-gradient direction. Instead, we want to explore other directions. This will help the detector avoid being trapped in local loops. In order to get the best trade-off between moving along the biggest gradient direction and exploring other directions, we conducted two experiments to find the optimal q settings for simulation areas without walls (plot(a) of Figure A1) and with walls (plot (b) of Figure A1). For each of the simulation areas, we tried six log-scaled q values: {0.03, 0.07, 0.16, 0.35, 0.80, 1.78}. For each q value, we repeated the experiment 200 times with fixed source intensity as 1000 cps. As shown in Figure A1, the best q was 0.16 for areas without walls, and the best q was 0.35 for areas with walls. Those two optimal values were used throughout all other experiments in this paper.

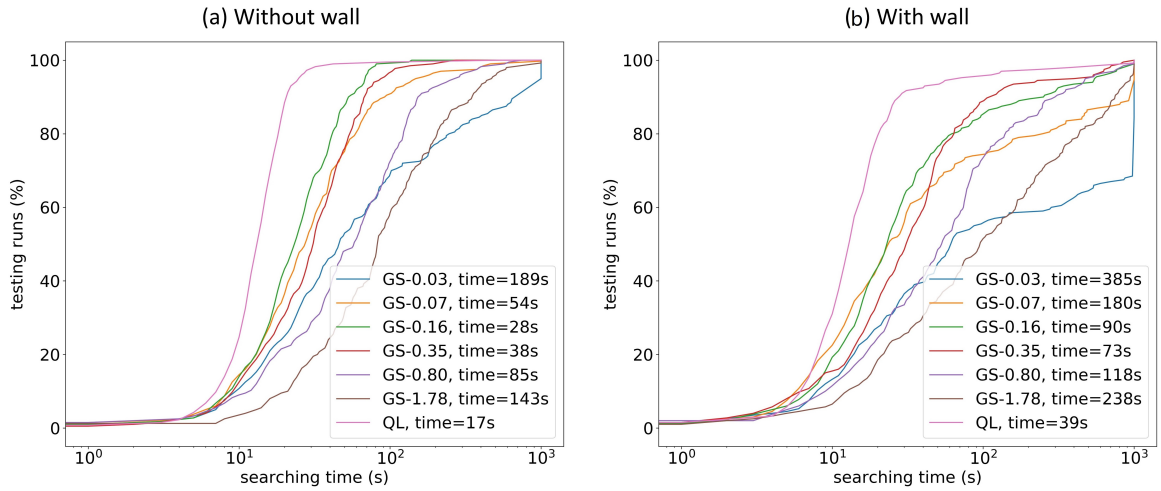

Figure A1. Parameter tuning for the gradient search method. These two plots represent the percentage of testing runs that are finished within specific searching times. Plot (**a**) shows the results from simulation area without walls, and plot (**b**) shows the results from simulation area with walls. 'GS' stands for the gradient search method, the number after 'GS' represents the q value. 'time' represents the mean searching time for that specific setting. 'QL' stands for the Q-learning method. Without walls, the best q is 0.16; with walls, the best q is 0.35.

Appendix C.2. Uniform Search Algorithm

The uniform search algorithm is a traditional source searching strategy that uses a zip-zap pattern to scan the entire searching area [27]. In this paper, we used three uniform search algorithms with low,

medium, and high searching densities represented by plot (a), (b), and (c) in Figure A2. The searching density controls the trade-off between searching time and searching coverage.

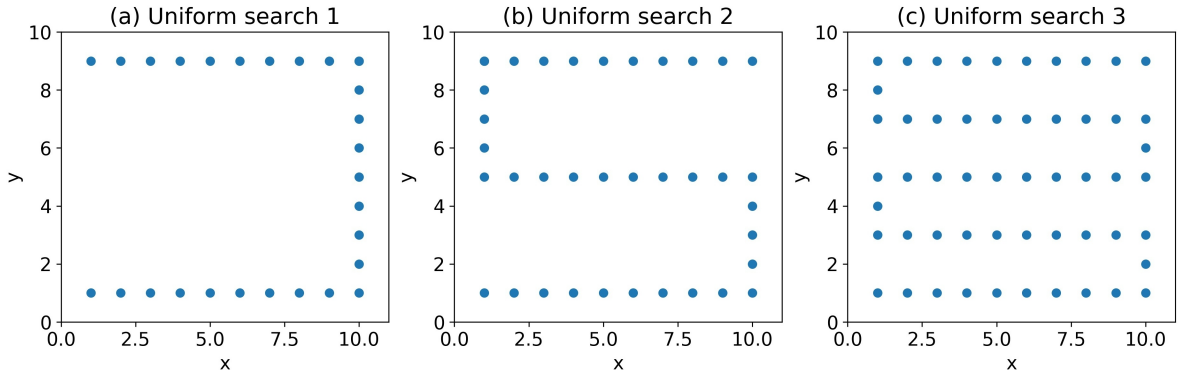

Figure A2. The uniform search algorithms with different searching densities. Each blue dot represents a position to take one measurement. From (**a**) to (**c**), their total search times are 28 s, 37 s, and 54 s respectively.

References

1. Zavala, M. Autonomous Detection and Characterization of Nuclear Materials Using Co-Robots. Ph.D. Thesis, Georgia Institute of Technology, Atlanta, GA, USA, 2016.
2. Liu, A.H. Simulation and Implementation of Distributed Sensor Network for Radiation Detection. Ph.D. Thesis, California Institute of Technology, Pasadena, CA, USA, 2010.
3. Philips, G.W.; Nagel, D.J.; Coffey, T. *A Primer on the Detection of Nuclear and Radiological Weapons*; Technical report; National Defence University, Center For Technology and National Security Policy: Washington, DC, USA, 2005.
4. International Atomic Energy Agency. Naturally Occurring Radioactive Material. Available online: https://www.iaea.org/topics/radiation-safety-norm (accessed on 19 February 2019).
5. United States Nuclear Regulatory Commission. Special Nuclear Material. Available online: https://www.nrc.gov/materials/sp-nucmaterials.html (accessed on 19 February 2019).
6. U.S. Department of Health and Human Services. Radiological Dispersal Devices (RDDs). Available online: https://www.remm.nlm.gov/rdd.htm (accessed on 19 February 2019).
7. Gunatilaka, A.; Ristic, B.; Gailis, R. On localisation of a radiological point source. In *Information, Decision and Control*; IEEE: Piscataway, NJ, USA, 2007; pp. 236–241. [CrossRef]
8. Chandy, M.; Pilotto, C.; McLean, R. Networked sensing systems for detecting people carrying radioactive material. In Proceedings of the INSS 5th International Conference on Networked Sensing Systems, Kanazawa, Japan, 17–19 June 2008; pp. 148–155. [CrossRef]
9. Deb, B. Iterative estimation of location and trajectory of radioactive sources with a networked system of detectors. *IEEE Tran. Nucl. Sci.* **2013**, *60*, 1315–1326. [CrossRef]
10. Bai, E.W.; Heifetz, A.; Raptis, P.; Dasgupta, S.; Mudumbai, R. Maximum likelihood localization of radioactive sources against a highly fluctuating background. *IEEE Trans. Nucl. Sci.* **2015**, *62*, 3274–3282. [CrossRef]
11. Liu, A.H.; Bunn, J.J.; Chandy, K.M. Sensor networks for the detection and tracking of radiation and other threats in cities. In Proceedings of the 10th International Conference onInformation Processing in Sensor Networks (IPSN), Chicago, IL, USA, 12–14 April 2011; pp. 1–12.
12. Morelande, M.; Ristic, B.; Gunatilaka, A. Detection and parameter estimation of multiple radioactive sources. In Proceedings of the IEEE 10th International Conference on Information Fusion, Québec, QC, Canada, 9–12 July 2007; pp. 1–7.
13. Morelande, M.R.; Ristic, B. Radiological source detection and localisation using Bayesian techniques. *IEEE Trans. Signal Process.* **2009**, *57*, 4220–4231. [CrossRef]
14. Pfund, D.M.; Runkle, R.C.; Anderson, K.K.; Jarman, K.D. Examination of count-starved gamma spectra using the method of spectral comparison ratios. *IEEE Trans. Nucl. Sci.* **2007**, *54*, 1232–1238. [CrossRef]

15. Alamaniotis, M.; Mattingly, J.; Tsoukalas, L.H. Kernel-based machine learning for background estimation of NaI low-count gamma-ray spectra. *IEEE Trans. Nucl. Sci.* **2013**, *60*, 2209–2221. [CrossRef]
16. Klimenko, A.V.; Priedhorsky, W.C.; Hengartner, N.W.; Borozdin, K.N. Efficient strategies for low-statistics nuclear searches. *IEEE Trans. Nucl. Sci.* **2006**, *53*, 1435–1442. [CrossRef]
17. Cortez, R.A.; Papageorgiou, X.; Tanner, H.G.; Klimenko, A.V.; Borozdin, K.N.; Lumia, R.; Priedhorsky, W.C. Smart radiation sensor management. *IEEE Robot. Automat. Mag.* **2008**, *15*, 85–93. [CrossRef]
18. Hutchinson, M.; Oh, H.; Chen, W.H. Adaptive Bayesian sensor motion planning for hazardous source term reconstruction. *IFAC Pap. Online* **2017**, *50*, 2812–2817. [CrossRef]
19. Lazna, T.; Gabrlik, P.; Jilek, T.; Zalud, L. Cooperation between an unmanned aerial vehicle and an unmanned ground vehicle in highly accurate localization of gamma radiation hotspots. *Int. J. Adv. Robot. Syst.* **2018**, *15*, 1729881417750787. [CrossRef]
20. Ristic, B.; Morelande, M.; Gunatilaka, A. Information driven search for point sources of gamma radiation. *Signal Process.* **2010**, *90*, 1225–1239. [CrossRef]
21. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; MIT Press: Cambridge, MA, USA, 1998; Volume 1.
22. Arulkumaran, K.; Deisenroth, M.P.; Brundage, M.; Bharath, A.A. A brief survey of deep reinforcement learning. *arXiv* **2017**, arXiv:1708.05866.
23. Watkins, C.J.; Dayan, P. Q-learning. *Mach. Learn.* **1992**, *8*, 279–292. [CrossRef]
24. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529. [CrossRef] [PubMed]
25. Van Hasselt, H.; Guez, A.; Silver, D. *Deep Reinforcement Learning with Double Q-Learning*; AAAI: Menlo Park, CA, USA, 2016; Volume 16, pp. 2094–2100.
26. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. 2015. Available online: tensorflow.org (accessed on 19 February 2019).
27. Ziock, K.; Goldstein, W. The lost source, varying backgrounds and why bigger may not be better. In *Unattended Radiation Sensor Systems for Remote Applications*; AIP Publishing: College Park, MD, USA, 2002; Volume 632, pp. 60–70.
28. Bellman, R. A Markovian decision process. *J. Math. Mech.* **1957**, *6*, 679–684. [CrossRef]