Autonomous Cognitive GPR Based on Edge Computing and Reinforcement Learning

Maxwell M. Omwenga[1], Dalei Wu[1], Yu Liang[1], Li Yang[1], Dryver Huston[2], Tian Xia[3]
[1] Department of Computer Science and Engineering, University of Tennessee at Chattanooga, USA
[2] Department of Mechanical Engineering, University of Vermont, USA
[3] Department of Electrical and Biomedical Engineering, University of Vermont, USA
Email: dgh179@mocs.utc.edu; {dalei-wu, yu-liang, li-yang}@utc.edu; {dryver.huston, txia}@uvm.edu

***Abstract*—Autonomous cognitive ground penetrating radar (ACGPR), carried by drones or other robotic platforms, may perform robust and accurate subsurface object detection and recognition in varying environments based on real-time data processing and decision making. However, limited system computing resources and intelligence generating capability pose significant challenges for the operations of such systems. To address these challenges, in this paper we propose an ACGPR enabled by edge computing (EC) and reinforcement learning. Specifically, an edge computing based system architecture is presented to utilize edge resources for real-time intelligence generation. A reinforcement learning approach is developed as the decision-making model for the ACGPR to adaptively adjust its operational parameters. Simulation results show the accuracy and efficacy of the proposed ACGPR system. The framework also provides insight into the design of autonomous cognitive industrial Internet of things (IoT) supported by edge computing and machine learning.**

***Index Terms*—Autonomous cognitive GPR, reinforcement learning, edge computing, signal processing**

I. Introduction

Ground penetrating radars (GPRs) have been extensively used in many industrial applications, such as coal mining, structural health monitoring, subsurface utilities detection and localization, and autonomous driving [1], [2]. A GPR system transmits an electromagnetic wave into the ground at several spatial positions and receives the reflected signal to form GPR data, called A-scans, B-scans and C-scans with different number of dimensions [1], [3]. Through processing these types of GPR data, subsurface objects can be detected and recognized.

Although GPRs are effective in many applications, most of existing GPR systems are human-operated due to the need of experience in operation configurations based on the interpretation of collected GPR data. GPR-based subsurface survey is complicated as various sensing environment and subsurface objects have dissimilar features. In actual GPR survey, GPR sensing quality could be affected by many factors, including environmental factors, such as soil dielectric properties, environment noise, clutter, multipath effects, combined near and far field effects, and GPR operational system parameters, such as wavelength (or frequency), waveform, polarization, wave timing, and transmitter and receiving antennas direct coupling, etc. In addition, the subsurface objects have different structural features and electromagnetic (EM) properties that affect GPR

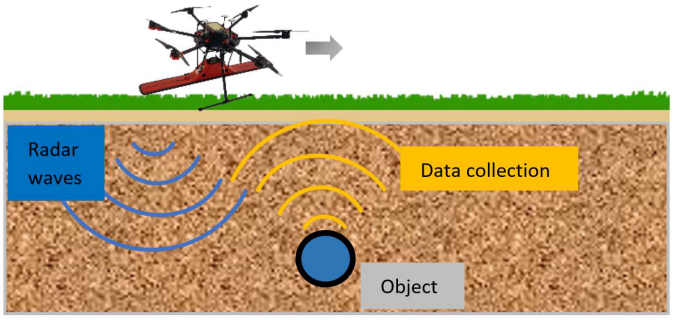

Fig. 1. A drone-borne GPR transmits radio signals into the ground and receives the signals reflected by a underground object.

EM wave propagation differently. Hence processing GPR data and extracting information of interest are challenging and involve a series of sophisticated steps. In nearly all existing GPR systems, GPR data processing is performed off-line where the data are collected and stored on field, and then post-processed on a computer after the scanning. Such a processing approach is time-consuming and lacks adaptivity. Also, some applications involve sensing tasks within hazardous and inaccessible environments. To achieve optimal sensing performance, it is desired to design an autonomous cognitive GPR (ACGPR) system that can operate adaptively under varying sensing conditions. Specifically, the system is able to adaptively move with a robotic platform and adjust its operational parameters through real-time interaction with the sensing environment.

There has been some work done on the study and development of autonomous GPRs [4]–[6]. Cornick et al. [4] describe a localizing GPR system fused with GPS, LiDAR and camera hooked at the bottom of an autonomous vehicle for autonomous ground vehicle localization. The system allows real-time creation of single-track maps with online data processing, as well as real-time localization of the vehicle to a prior map. In [5] the authors developed an autonomous robotic system employing GPR probing of glacier surfaces for void detection in ice. Supervised machine learning with pre-trained models was applied to automatically classify data into crevasse and crevasse-free classes. Foessel et al. [6] described a sled-

978-1-7281-2977-8/19/$31.00 ©2019 IEEE
DOI 10.1109/ICII.2019.00066

mounted GPR integrated with position and latitude instrument for autonomous search for antarctic meteorites. Although the aforementioned systems have used robotic systems to move GPR scanners, GPR moving and operational parameters were not adaptively adjusted on the fly.

The concept of cognitive radar was first proposed in [7] where recursive cognitive cycles are performed through a feedback mechanism to dynamically tune radar operational parameters and continuously improve sensing performance. The cognition is based on real-time GPR data processing, which requires significant computing and storage capability of the system. However, typical ACGPR systems, such as drone-borne GPR, as shown in Fig. 1, have limited computing, storage, and power resources.

Edge computing (EC) provides promising support for the implementation and operation of ACGPRs. Edge computing aims at pushing the computation, communication, and storage resources from the remote data center to the edge of network [8]–[10]. The name "edge" typically signifies the point at which traffic enters or exits the network, for example, data gateways to collect data from sensors or user devices. EC is expected to reduce the computing latency and the burden of backbone, and enable analytics and intelligence generation to occur at the source of the data. In an edge-computing-enabled ACGPR system, the proximity of edge servers to the ACGPR sensor may satisfy requirement for real-time or low-latency transmission of data and control feedback. Also, for the purpose of resource conservation, the ACGPR sensor may offload some of the computation tasks to an edge server. In contrast, traditional remote cloud computing services are not suitable for ACGPR due to the intermittent network connectivity and long communication latency. There has been some work done on the research of edge computing for the support of real-time and/or autonomous systems [11], [12].

Although edge computing makes it very promising to develop an ACGPR , there are still several significant research challenges that need to be addressed. One of the major challenges is the online intelligence generation in the context of edge computing to enable a resource-constrained GPR to autonomously and adaptively perform sensing tasks in unknown environment. Existing robotic or drone-borne GPRs are human-controlled with prior knowledge of the environment [13]. In reality, however, these human-controlled systems lack flexibility, scalability, and efficiency. Also, the knowledge regarding the environment is normally limited or unavailable. Therefore, a methodology allowing GPR to learn to make decisions on missions through exploring unknown environment is needed.

As a computational methodology for automatic decision-making of intelligent agents in uncertain environments, reinforcement learning (RL) has progressed tremendously in the past decade [14]. RL is mainly concerned with how RL agents ought to take actions in an environment so as to maximize some notion of cumulative reward. The full potential of RL requires an agent to directly interact with the environment to attain a flow of real-world experience. RL methods have been successfully applied on the operation of autonomous systems [15], [16].

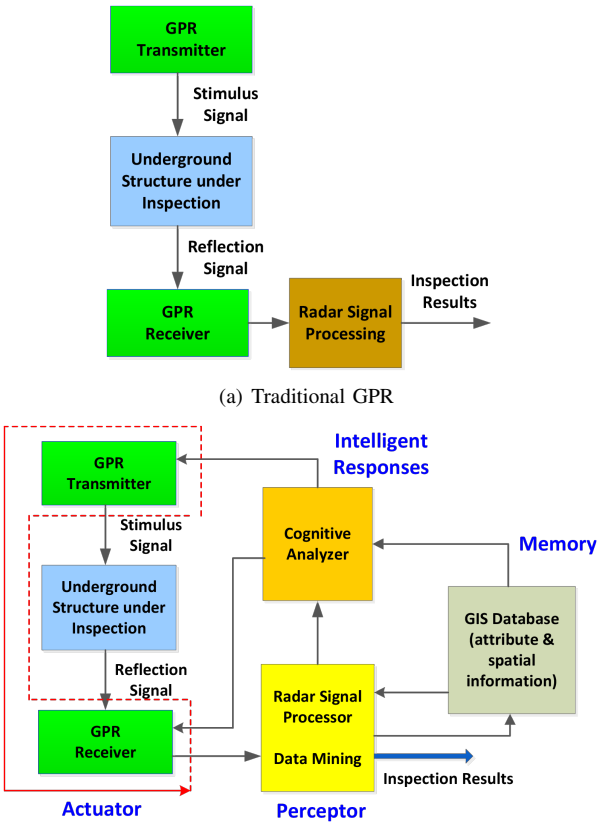

Fig. 2. The operational flows of a traditional GPR (a) and a cognitive GPR (b).

This paper is focused on the development of an edge computing and reinforcement learning framework that enables autonomous cognitive GPR (ACGPR). First, an edge computing architecture for ACGPR is developed. Functions of different modules of the architecture are explained. Second, a Q-learning approach with proper reward function is developed to learn a policy that directs the ACGPR's actions in an unknown environment. To the best of our knowledge, this is the first work based on edge computing and reinforcement learning for the development of ACGPR.

II. System Architecture

The tuning of operational system parameters of a ACGPR, such as wavelength (or frequency), waveform, polarization and wave timing, can lead to considerable improvement in the quality and scope of gathered information. Fig. 2(a) shows the conventional mode of ACGPR operation where an expert sets the operational parameters into an optimal configuration based on prior experience with similar past situations. This is an iterative time-consuming process. The conventional mode is not suitable for continuous long-time operations, especially in a complex environment inaccessible to humans.

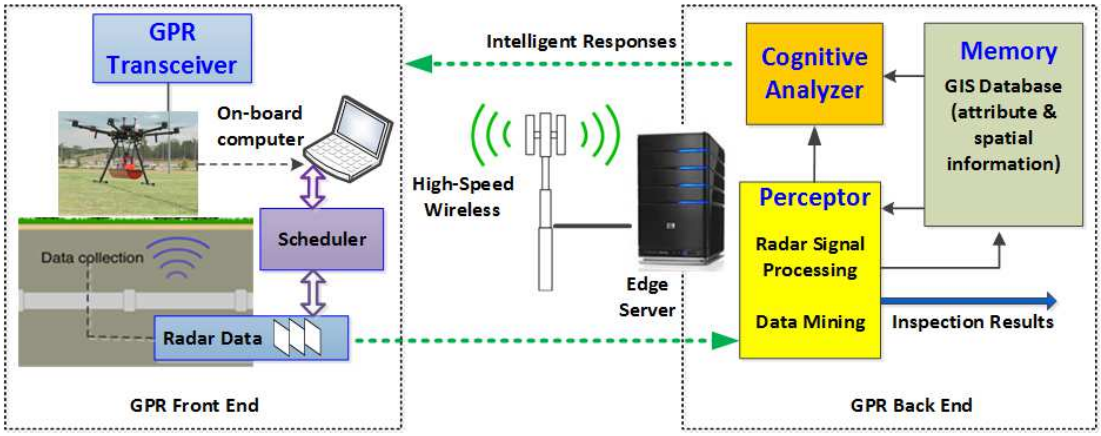

Fig. 3. The architecture of the proposed edge computing (EC)-enabled autonomous cognitive GPR (ACGPR).

The concept of cognitive GPR was proposed in [7] where intelligence is generated on the fly to adaptively adjust the operational parameters based on data analysis and feedback control. As shown in Fig. 2(b), a cognitive GPR consists of an adaptive GPR transceiver, a perceptor module, a memory module, and a cognitive analyzer. The operation of the cognitive GPR follows a perception-action cycle: first, the GPR transceiver collects the reflected wave data about subsurface objects and sends them to the preceptor. Then, the preceptor processes and analyzes the data to extract signature patterns and format a perception of subsurface conditions. The memory module has a geographic information system (GIS) database containing attributes about environment conditions and spatial locations. The cognitive analyzer carries out computational learning based on both the processing results from the perceptor and the prior knowledge about the environment from the memory module to produce intelligent response for the control of radar transceiver reconfigurations. During this process, collected GPR data can also be integrated with other data acquired by IoT devices such as positioning sensors and soil moisture sensors. Once receiving the intelligent feedback from the cognitive analyzer, the adaptive GPR transceiver adjusts its operational parameters.

Although the concept of cogitive GPR is promising for detection and recognition of subsurface objects, it is challenging to develop an ACGPR with resource-limited mobile robotic platforms, such as unmanned aerial vehicles (UAV) and unmanned ground vehicles (UGV). Both the perceptor module and the cognitive analyzer need sufficient computing capacity to perform real-time data analysis. A large memory capacity is required to maintain and update the prior knowledge available to the system.

In view of the advantage of edge computing that brings computing and storage resources to edge devices, we propose an edge computing based system architecture for ACGPR, as shown in Fig. 3 [17]. The system mainly includes two parts: the front end and the back end. The front end is mobile and includes a GPR transceiver for launching and receiving electromagnetic waves, a microcomputer for local computation, and a wireless access point for communicating with the edge server. The back end of the ACGPR resides at the edge server and includes the perceptor module, the memory module, and the cognitive analyzer.

The perception-action cycle of the edge computing enabled ACGPR incorporates the communication between the front end and the back end, and can be described as follows. The GPR transceiver at the front end captures the reflected wave data about underground objects. There are several types of computation tasks that need to be performed over the GPR data, such as data preprocessing, data compression, region-of-interest identification, and object detection and recognition. Based on the resource constraints and the delay performance requirement, a scheduler running within the microcomputer at the GPR front will decide whether each task should be performed locally at the front end or offloaded to the edge server at the back end [17]. Following the decision, the microcomputer either executes a task locally or offloads the task. With the related information from the GPR front end, the cognitive analyzer at the edge server performs machine learning and generates control command for the GPR transceiver recalibration. The control command will be wirelessly sent back to the GPR front-end. As a result, the operational parameters of the GPR is adjusted in a self-adaptive manner, and will be continuously updated as the GPR scanner explores the environment.

For a detailed discussion about the computation tasks offloading method, the interested readers are referred to reference [17]. This paper will focus on the implementation of the cognitive analyzer module based on reinforcement learning.

III. The Cognitive Analyzer Based on Q-Learning

The cognitive analyzer is a critical module of the proposed ACGPR. It produces intelligent responses to control the GPR movement and its operational configurations based on the collected GPR data and prior knowledge about GPR measurement. This section presents a reinforcement learning approach to the implementation of the cognitive analyzer. Specifically

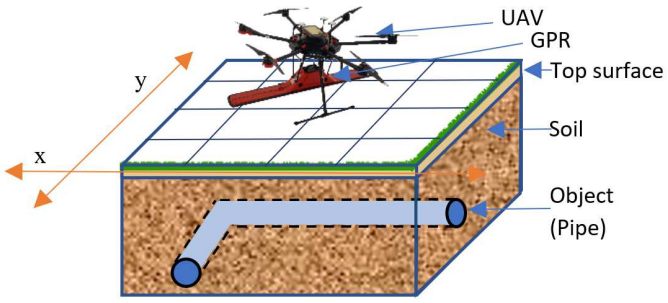

Fig. 4. An autonomous cognitive GPR, performing subsurface sensing for a buried L-shaped pipe.

a near optimal ACGPR sensing algorithm based on ϵ-greedy Q-learning is developed.

A. Modeling of Autonomous Cognitive GPR Sensing

In view of typical missions of the proposed ACGPR system and its limited energy resource, it is considered in this work that the goal of the proposed ACGPR is to detect a subsurface object in some unknown environment with minimum latency and energy consumption.

Let $s_k = (x_k, y_k, E_k) \in \mathcal{S}$ denote the system state in each time epoch k, where x_k and y_k are the horizotal coordinates of the GPR location in a three-dimensional environment, as shown in Fig. 4. E_k is the amount of remaining energy of the battery powering the GPR.By observing the state s_k the GPR chooses an action $a_k = (n_k, s_k, e_k, w_k, p_k, f_k) \in \mathcal{A}$ where n_k, s_k, e_k, w_k denote the actions of moving north, south, east, west, respectively, and p_k and f_k are the adopted transmit power and radio frequency, respectively. Let r_k specify the immediate reward the GPR agent attains after taking action a_k at state s_k and transitioning to state s_{k+1}. Thus, r_k can be defined as a reward function: $r_k : SXA \rightarrow \mathbb{R}$.

The action variables are optimized in each time epoch to maximize the long-term accumulated reward. In addition, the state transition and reward are stochastic and can be modelled as a Markov decision process, where the state transition probabilities and reward depend only on the environment and the obtained policy. The state transition probability $\mathbb{P} = (s_{k+1}, r_k | s_k, a_k)$ is defined as the probability of transition from state s_k to state s_{k+1} with the reward r_k when the action a_k is taken according to the policy π. Therefore, the long-term expected reward is given by

$$V(s, \pi) = \mathbb{E}_\pi \sum_{k=0}^{K} \gamma^k r_k \tag{1}$$

where $\gamma = (0 \leq \gamma \leq 1)$ is the discount factor and $\mathbb{E}$ indicates the statistical conditional expectation with the state transition probability $\mathbb{P}$.

B. The Reward Function

It is assumed that at the start the ACGPR has no knowledge of the environment. Through exploration over time, the GPR agent learns a policy that maximizes the long-term reward.

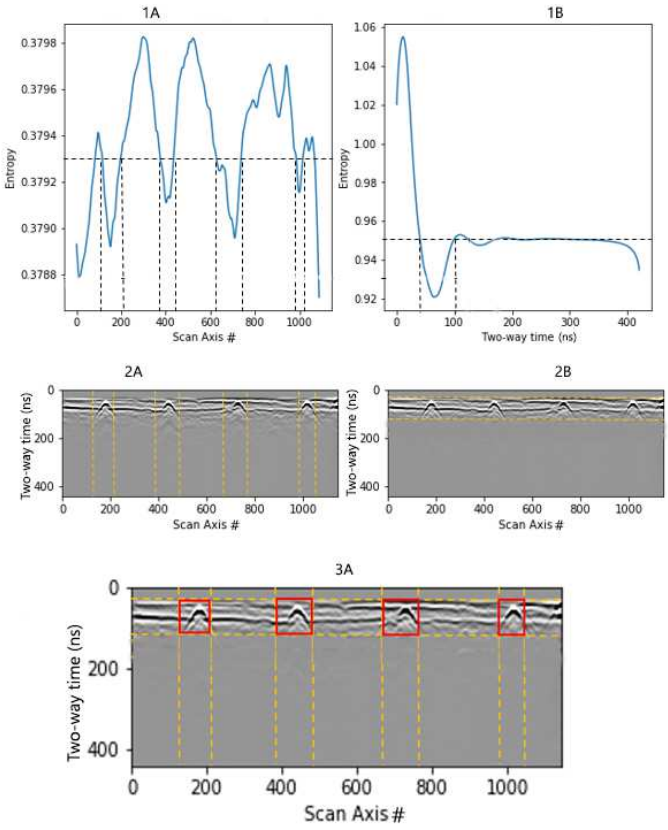

Fig. 5. 1A and 1B show the entropy values with respect to B-Scan's scan axis and two-way time, respectively. OTSU [18] thresholds of 0.3793 and 0.95 for 1A and 1B were used to identify points on the graph that fall below them and forms a valley. 2A and 2B are a one to one mapping with 1A and 1B that shows the exact region where entropy values are low. 3A indicates the identified region-of-interest (ROI) in red boxes derived by superimposing 2A on 2B to detect multiple hyperbola on a B-Scan.

Specifically the agent would receive a higher reward when it detects a subsurface object using less energy in a shorter time period. The reward function r_k can be defined as

$$r_k = a\frac{C_{max}}{C_k} + b\frac{p_{max}}{p_k} + c\frac{t_{max}}{t_k} \tag{2}$$

$$C_k = fE_\alpha(t) + gE_\alpha(s) \tag{3}$$

where a, b, c, f, g are the weight coefficients of different quantities; $C_{max}, p_{max}, t_{max}$ are the maximum entropy, maximum transmit power, and the sensing deadline respectively; p_k is the GPR transmit power, t_k is the actual time spent, and C_k denotes the sum of entropy where $E_\alpha(t)$ and $E_\alpha(s)$ are Renyi entropy that characterizes the singularity of the captured GPR data. Higher data singularity, corresponding to lower Renyi entropy, indicates higher chance of detecting the subsurface object. The calculation of $E_\alpha(t)$ and $E_\alpha(s)$ will be explained later as shown by Eq. (6) and Eq. (8), respectively.

In radargram, the region-of-interest (ROI) data have dissimilar features from the background data. By performing statistical analysis to evaluate data singularity, ROI data segments can

be identified. Then by checking corresponding coordinates, the location and burying depth of a subsurface object can be determined. In this study, Renyi entropy analysis [18], [19] is implemented to search for ROI.

For GPR data processing, Renyi entropy characterization is developed to identify the singular region. In particular, a high Renyi entropy value indicates high degree of data similarity while a low entropy value highlights high degree of data singularity. Assume the received GPR reflection signal is $Y(t)$, it can be described as

$$Y(t) = D(t) + S(t) \tag{4}$$

where $D(t)$ represents the reflection signal from the object of interest; $S(t)$ models remaining interference and noise upon preprocessing. In calculation, power normalization is first performed with the summation of the power of the same time index data points on different traces. The normalization equation is expressed as

$$y_i(t) = \frac{||Y_i(t)||^2}{\sum_{i=1}^{M} ||Y_i(t)||^2} \tag{5}$$

where $y_i(t)$ is the normalized signal, i is the trace index, M is the total number of traces included, and t is the time index of pulse data on each reflection trace waveform. Upon power normalization, a generalized Renyiś entropy is calculated to assess data singularity over wave travel time along Y-axis

$$E_\alpha(t) = \frac{1}{1-\alpha} \log_e \sum_{i=1}^{M} y_i(t)^\alpha. \tag{6}$$

where $E_\alpha(t)$ is the entropy quantification, and α denotes the entropy order. Eq. (6) is equivalent to the basic Shannon entropy when α equals 1.

Subsequently, Renyi entropy calculation is applied to scanning traces along X-axis

$$y_j(s) = \frac{||Y_j(s)||^2}{\sum_{j=1}^{T} ||Y_j(s)||^2} \tag{7}$$

where $y_j(s)$ is the normalized signal, j is the time index of pulse and T is the total number of time indexes; s is the trace index of pulse data. Then the Renyi entropy to assess data singularity over scanning position along X-axis is

$$E_\alpha(s) = \frac{1}{1-\alpha} \log_e \sum_{j=1}^{T} y_j(s)^\alpha. \tag{8}$$

Fig. 5 1A and 1B show the entropy values $E_\alpha(s)$ and $E_\alpha(t)$ with respect to B-Scan's scan axis and two-way time, respectively. The four low entropy value regions that fall below the OTSU threshold in Fig. 5 1A along the scan axis are 100 - 201, 382 - 413, 607 - 722, and 987 - 106, each indicating the presence of a subsurface object. Likewise in Fig. 5 1B is the region 49 - 100 along the two-way time axis. These values are marked one-to-one onto the original B-Sans as shown in Fig. 5 2A and 2B, which are superimposed to form Fig. 5 3A displaying the exact ROI marked with red boxes.

Algorithm 1: Autonomous Cognitive GPR sensing optimization algorithm based on ϵ-greedy Q-Learning

Initialize learning rate $\alpha \in (0 < \alpha \leq 1)$, epsilon $\epsilon = 0.9$, discount factor $\gamma \in (0 \leq \gamma \leq 1)$, EPS_DECAY = 0.9998, power max, environment
Initialize $Q(s, a)$, for all $s \in \mathcal{S}, a \in \mathcal{A}$
Set $x = 1$, Maximum epoch X
for $x \leq X$ **do**
    obs = env.reset();
    $e \leftarrow$ random number from (0,1);
    if $e < \epsilon$ **then**
        Choose action a_k randomly;
    else
        Choose action a_k according to
        $arg\max_{a_k \in \mathcal{A}} Q(s_{k+1}, a)$
    end
    Generate B-Scan and calc. Renyi entropy;
    Calculate C_k by Eq. (3);
    if $C_{k+1} \leq C_k$ **then**
        $r_k = a\frac{C_{max}}{C_k} + b\frac{p_{max}}{p_k} + c\frac{t_{max}}{t_k}$;
        $C_k = C_{k+1}$;
        done = True
    else
        done = False
    end
    new_obs = env.(obs);
    $\max_a Q(s_{k+1}, a)$ = np.max(q_table[new_obs]);
    if $done = True$ **then**
        $Q^{new} = r_k$;
    else
        $Q^{new} = (1-\alpha) * Q(s_k, a_k) + \alpha * (r_k + \gamma . \max_a Q(s_{k+1}, a))$
    end
    Update $Q(s_k, a_k)$ with Q^{new};
    Set $s_{k+1} = (x_{k+1}, y_{k+1})$;
    ϵ *= EPS_DECAY;
end

The entropy analysis is an intensive computation process that highly demands for computing power and CPU time. To leverage the strength of edge computing, in the proposed edge computing enabled ACGPR system, the entropy analysis is most likely implemented on the edge server instead of on the front-end microcomputer.

C. The Proposed Q-Learning Approach

In reality, the state transition probability $\mathbb{P}$ is difficult to obtain due to the uncertainty of the sensing environment. In this work, a model-free reinforcement learning approach, Q-learning, is investigated to solve the decision-making problem for optimal ACGPR sensing. The optimal policy π^* will be

derived to maximize the long-term reward $V(s,\pi)$. For any given state s, the optimal policy π^* can be obtained by

$$\pi^* = arg\max_{\pi} V(s,\pi), \forall s \in S \quad (9)$$

We denote the Q-value, $Q(s,a)$, as the expected accumulated reward when taking an action $a_k \in A$ following a policy π for a given state-action pair (s,a). Thus, the action-value function $Q(s,a)$ can be defined as

$$Q(s,a) = \mathbb{E}_\pi[r_k + \gamma Q_\pi(s_{k+1}, a_{k+1}) | s_k = s, a_k = a] \quad (10)$$

In our proposed Algorithm 1, $Q(s,a)$ is the value calculated from reward function Eq. (2) for any given state s and action a, and is stored in a Q-table which is built up to save all the possible accumulative rewards. The Q-value is updated during each time epoch if the new Q-value is greater than the current Q-value. The $Q(s,a)$ is updated incrementally based on the current reward function r and the discounted Q-value $Q(s_{k+1}, a), \forall a \in \mathcal{A}$ in the next time epoch. This is achieved by the one-step Q-update equation:

$$Q(s_k, a_k) \leftarrow (1-\alpha).Q(s_k, a_k) + \alpha(r_k + \gamma.\max_a Q(s_{k+1}, a)) \quad (11)$$

where r_k is the immediate reward for the current state, α is the learning rate $(0 < \alpha \leq 1)$ and γ discount factor $(0 \leq \gamma \leq 1)$ of the learning algorithm. In each time epoch, the Q-value is calculated in the next step by taking into account all the possible actions that could be taken, and then the maximum Q-value is chosen and the corresponding action is recorded.

To navigate through the unknown states instead of trusting the learned values of $Q(s,a)$ completely, the ϵ-greedy approach is used in the Q-learning algorithm with epsilon decay factor to strike a balance between the exploration and exploitation dilemma. Specifically, a random number $e = (0 < e \leq 1)$ is picked and if it is less than ϵ the agent selects a random action, otherwise chooses an action that maximizes $Q(s_{k+1}, a)$ as shown on lines (6-11) of Algorithm 1.

TABLE I
RELATIVE DISTANCE BETWEEN THE ACGPR AND THE SUBSURFACE OBJECT, AND THE RESPECTIVE ENTROPY VALUES WITH RESPECT TO SCAN AXIS AND THE TWO WAY TIME DERIVED FROM THE B-SCAN X AXIS AND Y AXIS RESPECTIVELY, C_k DERIVED FROM EQU. (3)

Distance (m)	**Traces**	**Entropy w.r.t scan axis** ($E_\alpha(s)$)	**Entropy w.r.t two way time** ($E_\alpha(t)$)	C_k
1	21	7.65	1.58	1.538
2	23	8.48	1.57	1.675
3	25	5.79	1.56	1.225
4	27	6.11	1.55	1.277
5	30	4.66	1.54	1.034
6	33	4.78	1.53	1.057
7	38	3.47	1.525	0.833
8	42	3.36	0.152	0.585
9	50	2.76	1.51	0.712
10	60	2.62	1.506	0.688
11	75	2.50	1.5	0.667
12	100	2.28	1.49	0.628
13	150	2.09	1.48	0.595
14	300	1.09	1.43	0.42

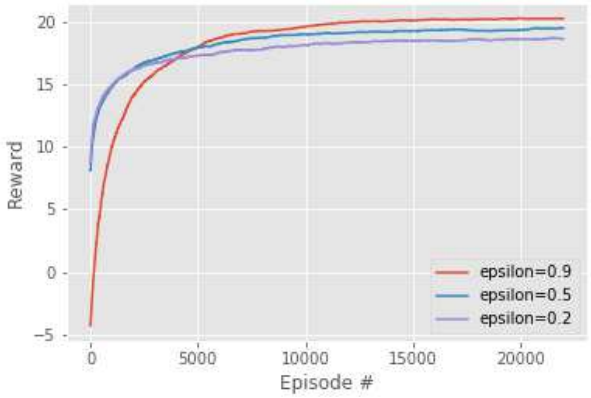

Fig. 6. Learning curves of different epsilon values $\epsilon = \{0.2, 0.5, 0.9\}$ applied to the autonomous cognitive GPR with 25k episodes.

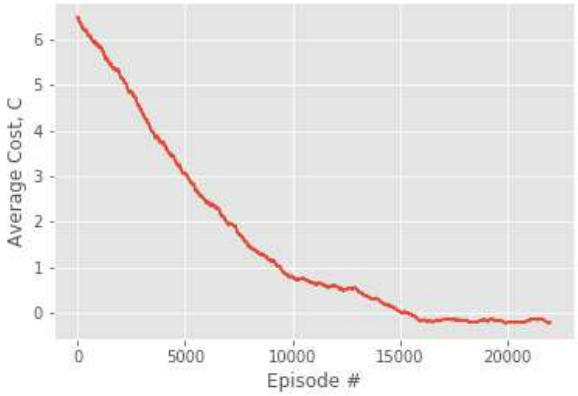

Fig. 7. Convergence performance of the proposed ϵ-greedy Q-learning algorithm in terms of the average cost.

IV. PERFORMANCE EVALUATION

In this work simulation is curried out to evaluate the proposed ACGPR. GprMax [20], designed for modeling GPR, is used to simulate a GPR agent that is configured to operate in two radio frequency bands, 400 MHz and 1.6 GHz. Selecting different radio frequencies and moving in different directions are considered as the possible actions of the GPR agent. Based on the relative distance between the GPR and the subsurface object, corresponding B-Scan can be generated as the collected observation data from the environment. Renyi entropy is calculated with respect to scan axis and two-way time as shown in Table I.

To evaluate the proposed Q-learning algorithm, its performance with epsilon $\epsilon = \{0.2, 0.5, 0.9\}$ and learning rate $\alpha = \{0.01, 0.2, 0.7\}$ is measured. High long-term reward r of 20.04 is achieved by the ACGPR when ϵ = 0.9, learning rate α = 0.1, discount factor γ = 0.95, and the epsilon decay = 0.9998.

Fig. 6 shows the learning curves of different epsilon values $\epsilon = \{0.2, 0.5, 0.9\}$ applied to the ACGPR with 25k episodes.

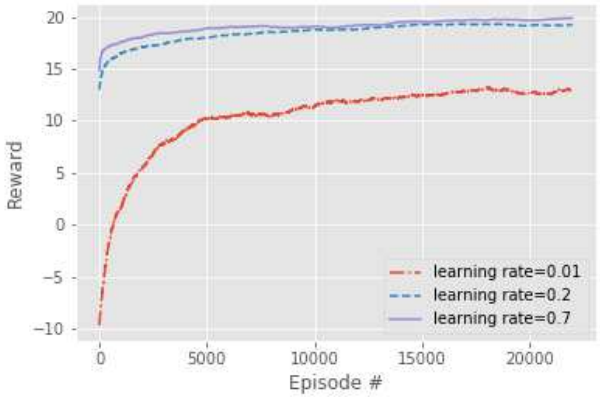

Fig. 8. Learning curves of different learning rate values $\alpha = \{0.01, 0.2, 0.7\}$.

It illustrates a high epsilon value close to 1 enables the agent to learn to achieve a higher long-term reward of approximately 20.04 while a lower epsilon value close to zero (i.e. 0.2 and 0.5) enables the agent to achieve a descent long-term reward of 17.1 and 18.9, respectively.

Fig. 7 illustrates the convergence performance of the proposed Q-learning algorithm, where Y-axis presents the average cost C, in each episode. It is observed that the average cost is converged to stable values below zero from episode 15K onward, where the ACGPR was able to locate the subsurface object accurately with the shortest time period.

Different configurations of learning rate α were carried out, as shown in Fig. 8. It can be observed that high Learning rates outperform lower learning rates.

V. Conclusions

In this paper, we proposed an autonomous cognitive GPR (ACGPR) system enabled by edge computing and reinforcement learning. The system architecture of the ACGPR was presented. A typical perception-action cycle of the ACGPR was explained. To adaptively adjust the movement of the GPR scanner and its operational parameters with the constraints of operational latency and power resource limitation in an unknown sensing environment, an ϵ-greedy Q-learning algorithm was developed to derive the optimum policy which tells the action of the ACGPR system at a given state, hence achieving a long-term reward for accurately identifying a subsurface object. Simulation was conducted to demonstrate the efficacy of the proposed system.

In real-world scenarios with high-dimensional observation spaces Q-learning algorithm is intractable because a huge size of memory is needed to store all the Q-values. Studies show that Deep Q-Network (DQN) algorithm may address this issue. However DQN can only handle the cases with discrete and low-dimensional action space [21]. Since ACGPR sensing is a physical control task with continuous and high-dimensional action spaces, as future work we will consider the Deep Deterministic Policy Gradient (DDPG) method which concurrently learns a Q-function and a policy [21].

References

[1] H. M. Jol, *Ground Penetrating Radar Theory and Applications*. Elsevier, 2009.

[2] "WaveSense," https://wavesense.io, 2019, [Online; accessed 25-June-2019].

[3] A. Turk, A. Hocaoglu, and A. Vertiy, *Subsurface Sensing*. Wiley, 2011.

[4] M. Cornick, J. Koechling, B. Stanley, and B. Zhang, "Localizing Ground Penetrating RADAR: A Step Toward Robust Autonomous Ground Vehicle Localization," *Journal of Field Robotics*, vol. 33, no. 1, pp. 82–102, 2016.

[5] R. M. Williams, L. E. Ray, and J. Lever, in *2012 IEEE International Geoscience and Remote Sensing Symposium*.

[6] A. Foessel-Bunting, D. Apostolopoulos, and W. L. Whittaker, "Radar sensor for an autonomous Antarctic explorer," in *Mobile Robots XIII and Intelligent Transportation Systems*, H. M. Choset, D. W. Gage, P. Kachroo, M. A. Kourjanski, M. J. de Vries, P. Kachroo, M. A. Kourjanski, and M. J. de Vries, Eds., vol. 3525, International Society for Optics and Photonics. SPIE, 1999, pp. 117 – 124. [Online]. Available: https://doi.org/10.1117/12.335690

[7] S. Haykin, "Cognitive Radar: A Way of the Future," *IEEE Signal Processing Magazine*, pp. 30–40, Jan. 2006.

[8] M. Chiang and T. Zhang, "Fog and IoT: An Overview of Research Opportunities," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854–864, Dec. 2016.

[9] S. Mishra, D. Putha, J. Rodrigues, B. Sahoo, and E. Dutkiewicz, "Sustainable Service Allocation Using Metaheuristic Technique in Fog Server for Industrial Applications," *IEEE Transactions on Industrial Informatics*, vol. PP, no. 99, Jan. 2018.

[10] D. Miao, L. Liu, R. Xu, J. Panneerselvam, Y. Wu, and W. Xu, "An Efficient Indexing Model for the Fog Layer of Industrial Internet of Things," *IEEE Transactions on Industrial Informatics*, vol. PP, no. 99, Jan. 2018.

[11] J. Tang, S. Liu, B. Yu, and W. Shi, "PI-Edge: A Low-Power Edge Computing System for Real-Time Autonomous Driving Services," 2018.

[12] M. Maheswaran, T. Yang, and S. Memon, "A Fog Computing Framework for Autonomous Driving Assist: Architecture, Experiments, and Challenges," 2019.

[13] A. Arato, F. Garofalo, G. Sammartano, and A. Spano, "Gathering GPR Inspections and UAV Survey in Cultural Heritage Documentation Context," in *2nd International Conference on Geographical Information Systems Theory, Applications and Management*, 01 2016, pp. 85–91.

[14] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An introduction*. Cambridge: MIT press, 2018.

[15] T. Sugimoto and M. Gouko, "Acquisition of Hovering by Actual UAV Using Reinforcement Learning," in *Proc. The 3rd International Conference on Information Science and Control Engineering (ICISCE)*, July 2016.

[16] W. Xia, H. Li, and B. Li, "A Control Strategy of Autonomous Vehicles Based on Deep Reinforcement Learning," in *Proc. The 9th International Symposium on Computational Intelligence and Design (ISCID)*, 2016.

[17] D. Wu, M. M. Omwenga, Y. Liang, L. Yang, D. Huston, and T. Xia, "A Fog Computing Framework for Cognitive Portable Ground Penetrating Radars," in *Proc. IEEE ICC*, May 2019.

[18] Y. Zhang, P. Candra, G. Wang, and T. Xia, "2-D Entropy and Short-Time Fourier Transform to Leverage GPR Data Analysis Efficiency," *IEEE Transactions on Instrumentation and Measurement*, vol. 64, no. 1, pp. 103–111, 2015.

[19] P. Jizba and T. Arimitsu, "On Observability of Renyi's Entropy," *Physical Review*, vol. 69, no. 2, 2004.

[20] C. Warren, A. Giannopoulos, and I. Giannakis, "gprMax: Open Source Software to Simulate Electromagnetic Wave Propagation for Ground Penetrating Radar," *Computer Physics Communications*, vol. 209, pp. 163 – 170, 2016.

[21] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep Reinforcement Learning for Robotic Manipulation with Asynchronous Off-policy Updates," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 3389–3396.