Enhancing Self-sustaining IoT Systems with Autonomous and Smart UAV Data Ferry

Mason Conkel*, Wen Zhang+, Chen Pan*
*Department of Electrical and Computer Engineering, The University of Texas at San Antonio, San Antonio, Texas, USA
+Department of Computer Science and Engineering, Wright State University, Dayton, Ohio, USA
mason.conkel@utsa.edu, wen.zhang@wright.edu, chen.pan@utsa.edu

2024 25th International Symposium on Quality Electronic Design (ISQED) | 979-8-3503-0927-0/24/$31.00 ©2024 IEEE | DOI: 10.1109/ISQED60706.2024.10528757

***Abstract*—The emerging unmanned aerial vehicle (UAV) such as a quadcopter offers a reliable, controllable, and flexible way of ferrying information from energy harvesting powered IoT devices in remote areas to the IoT edge servers. Nonetheless, the employment of UAVs faces a major challenge which is the limited fly range due to the necessity for recharging, especially when the charging stations are situated at considerable distances from the monitoring area, resulting in inefficient energy usage. To mitigate these challenges, we proposed to place multiple charging stations in the field and each is equipped with a powerful energy harvester and acting as a cluster head to collect data from the sensor node under its jurisdiction. In this way, the UAV can remain in the field continuously and get the data while charging. However, the intermittent and unpredictable nature of energy harvesting can render stale or even obsolete information stored at cluster heads. To tackle this issue, in this work, we proposed a Deep Reinforcement Learning (DRL) based path planning for UAVs. The DRL agent will gather the global information from the UAV to update its input environmental states for outputting the location of the next stop to optimize the overall age of information of the whole network. The experiments show that the proposed DDQN can significantly reduce the age of information (AoI) by 3.7% reliably compared with baseline techniques.**

***Index Terms*—UAV, IoT, DRL, Energy Harvesting**

I. Introduction

The emerging energy harvesting technology [1]–[3] that harvests energy, such as solar, wind, and thermal radiations, from the ambient environment has become increasingly appealing due to its low maintenance cost, sustainability, ubiquitous accessibility, and eco-friendliness. Those promising features enable various IoT applications to benefit remote areas such as forests, farmlands, and oceans where electrical infrastructures are generally inaccessible. However, due to limited power budget and line-of-sight (LOS) obstacles, energy harvesting-powered IoT devices in remote areas are inaccessible to the backbone of the IoT network, and hence cannot upload critical data collected to the backbone of the IoT network for further study and decision-making.

To address this issue, emerging unmanned aerial vehicles (UAV) [4]–[8] such as quadcopters offer reliable, controllable, and flexible methods of ferrying information from IoT devices in the remote area to the IoT edge servers, which can connect to the backbone of IoT networks regardless of topology. Nonetheless, the employment of UAVs faces a major challenge which is the limited fly range due to the necessity for recharging, especially when the charging stations are situated at considerable distances from the monitoring area, resulting in inefficient energy usage.

To mitigate these challenges, we can strategically place multiple charging stations in the field and each is equipped with powerful energy harvesters such as large solar panels. This ensures that UAVs can remain in the field continuously to minimize the energy wasted during transit. Moreover, considering the UAV's elevated position to circumvent LOS obstacles, by further integrating long-range communication modules with suitable protocols, the UAV can enhance communication reliability and distance and is capable of communicating with the IoT server directly. However, given the complexity of knowing/foreseeing the availability of IoT devices for communication under energy harvesting scenarios, effective communication between the UAV and all IoT sensor nodes remains extremely challenging. In this case, we will enable the charging station as the data cluster head to collect data from the sensor node under its jurisdiction so that the UAV can get the data while charging.

While the proposed solution looks promising, due to the vast monitoring area size and limited energy harvesting power budget of cluster heads for charging the UAV, there will be a significant amount of delay in the arrival of UAVs at each cluster head. As a result, the information gathered on cluster heads may become stale or even obsolete before the UAV arrives. To tackle this issue, in this work, we proposed a Deep Reinforcement Learning (DRL) based path planning for UAVs, where a unique DRL agent with shared structure but different inferences will be deployed on each charging station to serve as the decision maker of path planning of UAVs for the next destination. The DRL agent will gather the global information from the UAV to update its input environmental states for outputting the location of the next cluster head to optimize the overall age of information of the whole network. To the best of our knowledge, this is the first work implementing DRL for self-charging UAVs to conduct autonomous path planning. In a nutshell, the contribution of this work is as follows.

1) **Development of a Novel Framework:** We introduce the AoI-Aware UAV Data Aggregation System, a pioneering framework designed to minimize the AoI with a collaborative effort of air-ground IoT devices.
2) **Innovation in Path Planning:** We present a Deep Re-

This work is supported by the National Science Foundation under Grant No.2153524.

inforcement Learning (DRL)-based algorithm and ADF algorithm specifically tailored to optimize real-time path planning for data aggregation.

3) **Comprehensive Experimental Validation:** Solid experimental evaluations are conducted to demonstrate the performance of the proposed solution.

The remainder of this paper is organized as follows. Section II describes work that is closely related to this research. Section III conducts system modeling which describes the relationship between the major considerable factors of the air-ground IoT system. After that, Section IV formulates the DRL model for UAV's path planning to minimize the overall age of information of the IoT devices. Section V conducts experimentation to demonstrate the performance of DRL agents. Finally, Section VI concludes this work.

II. Related Work

In this section, we will summarize the existing works that are related to our research. In a brief overview, we will discuss the research method to reduce the age of information (AoI) for sensing and deep reinforcement learning-based path planning for UAVs conducting data ferry.

A. Age of Information

The assessment of metrics is often highlighted as a primary feature when constructing research. [9] examines the age of information (AoI) in a network by providing a custom metric that is used to train an RL algorithm. Further, they construct a metric for the energy efficiency of the UAV to monitor the performance of the network. These metrics are used to train the UAV and provide information for model evaluation. Next in [10], age is used for path planning and data acquisition. This produced time-based metrics for training their reinforcement learning algorithm. The application of these metrics was applied to configure the algorithm that controlled the UAV and drove performance. Finally, [11] utilizes the AoI and energy efficiency as metrics to determine the performance of their proposed algorithm. This is to ensure the timely acquisition of data while maintaining the power efficiency of the model. Similarly, this paper utilizes AoI and data collection as the primary force behind model reward. These will be explained further in the description of the algorithm.

B. Deep Reinforcement Learning

RL and DRL have seen growth in the last few years as forerunning models in AI research. Specifically, neural networks in DRL have shown potential in a variety of learning applications. In [12], Double Deep Q Networks (DDQN) were implemented for trajectory and data harvesting on a UAV-assisted IoT network. As an extension of Q Learning (QL) DQN and DDQN have been deployed frequently for their ease of use and logical similarity to Q Table learning models. In [13], the research is directed towards developing faster training methods for DRL models by pruning weights. This is to make DRL more efficient in future applications and lessen the time component required for training. They compare

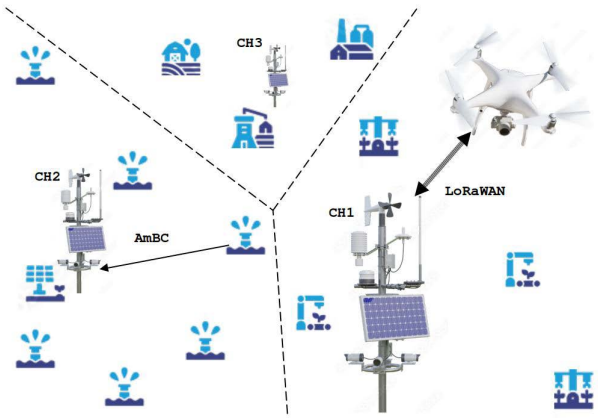

Fig. 1: Self-Sustaining Air-ground IoT Systems

this technique to the performance on standard QL techniques. What's more, [14] utilized a DQN to minimize the AoI of a system and explain their process for constructing it. In the paper, a focus is put on defining the state and action spaces before explaining the model. Finally, [15] applied a similar DQN model to the data collection of a network to optimize energy efficiency. To that extent, they identify the Partially Observable Markov Decision Process (POMDP) and formulation of their DQN.

III. System Model

As depicted in Figure 1, the proposed framework consists of three components, including an EH-powered IoT device, Cluster Head (CH) that also plays as a charging station for UAV, and UAV. A single UAV is deployed to gather data from energy harvesting (EH)–powered IoT devices. We will model the key physical features of each component.

A. Ground Communication Model

We placed M EH-powered IoT devices at random locations across the environment, selecting random locations to ensure comprehensive coverage. Utilizing the K-Means algorithm, we deploy N cluster heads, placed at the centroids of each sub-area. The location of node m from M the collection of M total sensors is noted as (x_m, y_m).

These cluster heads are entrusted with the responsibility of aggregating data from the EH-powered IoT devices within their respective clusters. The EH-powered IoT device and cluster head are labeled with m and n, respectively. The time step is identified with t. During each time step t, a random quantity of $D_{m,t}$ data units is stored in the memory of device m, mimicking environmental interference that may corrupt the data.

CH will prepare a channel for the data uploading of its respective cluster members. Ambient Back Scatter Communications (AmBC) is adopted for the communication between the EH-powered IoT device and CH as published in Liu et. al. [16]. This utilized Long Range Wireless Access Network (LoRaWAN) as the carrier signal. The data transmitted for any period is determined by Equation 1. Both communication

protocols operated with the specifications reported in related works and documentation.

$$D_{trans} = R_{max} \times (1 - BER_{type}) \times t_{trans} \tag{1}$$

In which the data transmitted D_{trans} is determined by the reported optimal transmission rate R_{max}, the average bit error rate depending on the type BER_{type} and the time the data was transmitted t_{trans}.

B. UAV Data Aggregation Model

After the ground data aggregation, CH will transmit data to the UAV by utilizing the carrier LoRaWAN signal. As with AmBC, LoRaWAN was selected for its low-power draw and long-range capabilities. The amount of time transferring can be determined using:

$$t_{trans} = \begin{cases} T - t_{HS}, & \text{if } D_{stored} \geq D_{max} \\ \left(T \times \frac{D_n}{D_{max}}\right), & \text{otherwise} \end{cases} \tag{2}$$

where transmission t_{trans} is found with the step size T, the time to establish connection and for protocol backend t_{HS}, the remaining data currently stored by the cluster head D_n and the maximum data that can be transmitted in one step D_{max}. Additionally, the age of information for the data is:

$$age = \begin{cases} T_{packet} - T_{step}, & \text{if age} > T_{threshold} \\ T_{threshold}, & \text{otherwise} \end{cases} \tag{3}$$

where age is determined using the step at which the last packet was transferred from the cluster head to the UAV T_{packet}, the current step T_{step} and the established threshold $T_{threshold}$. By optimizing the traverse path of UAV with DRL, we aim to minimize the peak AoI_{peak} and average AoI_{avg} age of information for the collected data, as shown in Eq. (4).

$$\begin{cases} AoI_{avg} = \dfrac{\sum_i^N age_i}{N} \\ AoI_{peak} = argmax_i(age_i) \end{cases} \tag{4}$$

C. Energy Model

EH-Powered IoT Devices. EH-powered IoT devices m are initialized with solar panel size F_m and battery capacity B_m. The pvlib function "irradiance" returns the spectral irradiance at the specified Geographic coordinates that are identified by the label of devices, date, and time [17]. At every step, the energy harvested by the solar cell is adjusted by converting the spectral irradiance from pvlib into power density and resulting power supplied to the system. Excess power at every step is stored in the battery for instances in which the interference causes the cost of data harvesting and communication to exceed the energy harvested at that step. The harvested energy of m is equal to:

$$P_{harvest} = A_{cell} \times (1 - I_{solar}) \times \int_{low}^{high} (TSI \; d\lambda) \tag{5}$$

in which A_{cell} is the area of the solar cell at device m, I_{solar} is the interference of shade as witness by device m, and the integral on the total solar irradiance TSI is taken from the lowest wavelength to the highest wavelength specified for the solar cell. The TSI is obtained from pvlib's irradiance function. The energy cost of m is given by:

$$P_{cost} = \sum_{j=0}^{\tau} (R[n] \times t) \tag{6}$$

in which for each respective operation, its power draw is determined by the rate at which it draws power $R[n]$ and the time in which it utilizes the process in the time step t. τ labels the total number of time steps for the corresponding operations. The rate at which an operation draws power is taken from the specifications for that element, such as LoRaWAN communication specifications for transmitting, receiving and idle power draw.

UAV. Define B_{uav} as the maximum onboard energy. UAV will recharge by docking on the CHs. For cluster head n with the current battery level B_n. The harvested energy of UAV is:

$$P_{charge} = t_{charge} \times R_{charge}, \quad P_{charge} \leq B_{uav} \tag{7}$$

where t_{charge} is the time spend charging and R_{charge} is the charging rate as specified by the UAV specification for which we chose to model the UAV on. The energy consumption of UAV consists of propulsion and communication. Therefore, the on-board energy of UAV is:

$$B_{current} = B_{stored} + P_{charge} - (P_{move} + P_{LoRaWAN}) \tag{8}$$

in which the $B_{current}$ is the current battery at the end of the step, B_{stored} is the battery at the start of the step and P_{move} and $P_{LoRaWAN}$ are the costs of movement and LoRaWAN communication, respectively.

IV. AoI Minimization for Self-sustaining Air-ground IoT System

To minimize the age of information (AoI) for globally collected data, we developed Deep Reinforcement Learning (DRL)-based path planning and ADF algorithm for UAV navigation. The DRL algorithm strategically determines the next charging station to target, optimizing the UAV's navigation process. However, given the intricate nature of communication networks, we have augmented the DRL with the ADF algorithm. This supplementary algorithm adeptly addresses emergencies and constraints that may arise, ensuring the UAV's navigation strategies remain robust and adaptable. We first delve into the exceptional cases handled by the ADF algorithm before delving into the detailed design of the DRL-based algorithm.

A. ADF Algorithm

Case 1: UAV Energy Constraints. Taking into account the constraints imposed by the UAV's limited on-board energy capacity, the UAV is programmed to initiate a return flight whenever its on-board energy reserves P_{uav} fall below the threshold value P_{uav_thresh}. In such instances, the ADF algorithm directs the UAV to dock with the current target CH.

Algorithm IV.1: ADF:AoI-aware Data Ferry

Input: $\{1, 2, \ldots, N\}, (x_n, y_n) \forall n \in \{1, 2, \ldots, N\}, P_{UAV thresh}, D_{thresh}$

if *No* CH_{Target} **then**
  $CH_{target} = argmin \quad Distance_{\{uav,n\}}$
else if $P_{UAV} \leq P_{UAV\_thresh}$ **then**
  UAV returns back to CH_{target}
else
  if $D_n \geq D_{thresh}$ **then**
    $CH_{target} = n$
  else if $\forall CH\ \exists\{CH | CH_{aoi} = 0\}$ **then**
    $CH_{target} = \{CH \,|\, CH_{aoi} = 0\}$
  else
    $CH_{Target} = rand\{Q^A(s) \parallel Rand(A)\}$ ## $\mathcal{N}$
return CH_{Target}

Case 2: Data Overload. Each CH possesses a finite storage capacity for data. If any CH exceeds its storage threshold, denoted as D_{thresh}, the ADF algorithm orchestrates the UAV to navigate directly to that CH for data transfer.

Case 3: Null Action. During the gap time required for data preparation and sensing by EH-powered IoT devices, the UAV's services are not required. In such instances, the ADF algorithm proactively assigns a CH as the target destination for the UAV's break period.

Case 4: Unbalance Service. To mitigate the risk of the DRL agent consistently selecting outdated CHs, the ADF algorithm intervenes by directing the UAV towards unserviced CHs.

The details are described in Algorithm IV.1.

B. DDQN-based AoI Minimization

State. The environment state is encapsulated within a $(N + 1) \times 3$ matrix, where each row represents the state of a CH, denoted by $s_n = \{n, D_n, \text{age}_n\}, \forall n \in 1, 2, \ldots, N$. Here, Dn signifies the total data contributed to D_{uav} by the CH, while age_n denotes the Age of Information (AoI) computed using Eq. (3) for that CH. An additional row, appended to the matrix, characterizes the real-time status of the UAV, represented by $s_0 = \{0, D_{\text{uav}}, P_{\text{UAV}}\}$. Here, D_{uav} corresponds to the total data collected during the simulation, while P_{UAV} reflects the on-board energy reserves of the UAV. The state varies at each time step t. Therefore, we omit the label on t.

Action. In contrast to the popular model that outputs coordinate locations or a trajectory for the UAV to follow, this model outputs a specific CH $a = \mathcal{N} \in \{1, 2, \ldots, N\}$ for the UAV to target next. This is to ensure scalability and reduce complexity when the model makes a decision on the next target while limiting the amount of environmental input required for the model to make such a decision.

Reward. Five factors contributed to the reward function: AoI_{avg}, AoI_{peak}, distribution of data, D_{uav}, and termination by crash. Using these metrics and accumulated data, the reward functions were designed to represent the environment response to the model. Further, the reward for the average AoI can be determined by the average and peak metrics as formulated in Eq. (9).

$$r_{AoI} = 1 - 2 \times \frac{AoI_{peak} - AoI_{avg}}{T_{threshold}} \tag{9}$$

After defining the reward of average AoI, we will further formulate the reward for peak AoI in Eq. (10) which uses the peak and threshold. Here, $T_{threshold}$ was to encourage the hourly service of all CHs. Taking the difference between the peak and the average for the reward is to ensure that the peak is drawn closer to the average AoI. The reward from this equation is used further to drive the peak AoI away from the threshold value.

$$r_{peak} = 1 - 2 \times \frac{AoI_{peak}}{T_{threshold}} \tag{10}$$

Next, the distribution of data was determined using the percentage of data collected at each cluster head over all time. This is done by finding the range of the percent contribution of each cluster head to the total accumulated data and subtracting from one as shown in Eq. (11).

$$r_{distri} = 1 - 2 \times \frac{D_{Dist}}{D_{uav}} \tag{11}$$

where D_{uav} is the total data accumulated by the UAV and the metric for distribution D_{dist} which can be formulated as in Eq. (12) by using the total contributed data from each CH. Here D_{max} and D_{min} are the maximum and minimum amounts collected, respectively.

$$D_{Dist} = D_{max} - D_{min} \tag{12}$$

The reward is calculated as a weighted sum of the rewards defined above, as illustrated in Eq. (13), where δ, β, and ζ represent the respective weights assigned to each term.

$$reward = \begin{cases} -100, \text{ if UAV crashes} \\ \delta r_{AoI} + \beta r_{peak} + \zeta r_{distri}, \text{ else} \end{cases} \tag{13}$$

Training. We adopt DDQN as underlying for DRL agents training. Algorithm IV.2 described the training process. The algorithm utilized during training was constructed using conditional statements and a DRL model.

Training occurred over 1e5 steps and was kept constant for all models. The first 5e4 steps were allocated for pre-training. The next 5e4 steps were documented and evaluated within Section V. During this time, epsilon variables were gradually adjusted 1 to 0.0001. During displayed periods of evaluation, the epsilon variable, which is used to determine the randomness of the action performed, was set to 1 for proper evaluation of runtime performance.

C. Rationale

ADF utilizes three major stages of decision making based on a priorities of the model. Priority hierarchy begins with the UAV, then depends on the currently targeted CH, and ends with the DDQN algorithm.

UAV. To keep the onboard computational overhead at a minimum, the UAV ranks as high priority for initializing a target

Algorithm IV.2: Training:ADF Training Process

Input: Simulated Network Environment env
$Q^A \leftarrow$ **initialize** DQN
$Q^B \leftarrow$ **initialize** DQN
$B_{mem} \leftarrow$ **empty** Deque
$s \leftarrow env(\textbf{reset})$
while *step* $\leq$ *max_steps* **do**
  $a = Q^A(s)$
  $s', r, d \leftarrow env(a|s)$
  $B_{mem} \leftarrow \{s, a, s', r, d\}$
  if *len*(B_{mem}) $\geq$ *batch_size* **then**
    Update Q^A
  if *step % hard_update_timer = 0* **then**
    $Q^B \leftarrow Q^A$
  $s \leftarrow s'$

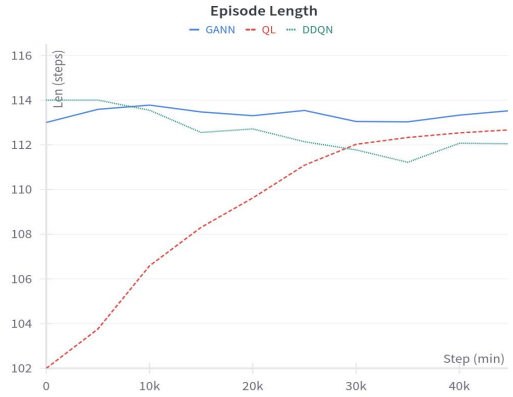

Fig. 2: The average episode length at each evaluation period for each of the three algorithms used.

when none is provided and preventing crashes when battery reaches a threshold. This is seen in Algorithm IV.1 during the initial if-else cases. This priority is highest as it allows for the continued operation of the UAV during extreme circumstances.

Clusterhead. To prevent the build up of information and leave room for future modifications, the CH currently being targeted is given the second highest priority in the ADF algorithm. Given that the DDQN model operates on the CH level, the implementation of CH-specific constraints offers a chance to implement mew features. In ADF, these restrictions first activate to ensure that data stored on the CH is below a maximum threshold. This allows for optimal gathering from assigned sensors by ensuring a maximum quantity of data is transmitted to the UAV.

Next, the CH analyzes the state and ascertains whether any CH requires service. This process is essential to initialize AoI values before allowing the state to pass to ADF's DDQN. Without initialization of all AoI values, the state may be too underdeveloped for the DDQN algorithm.

Model. Given all restrictions are avoided, the CH being targeted will operate the onboard DDQN model using the provided state. The training process of the DDQQN model is seen in IV.2. The model is designed to output the next CH to be targeted by the UAV. For the ADF algorithm, this has a lower priority but was utilized more often than the constraints during training.

V. Experiments

In this section, we will first set up the experiment including the data set and baselines. Then we will conduct an experimental evaluation of our proposed method.

A. Experimental Setup

In this paper, we utilize the pvlib library developed by Sandia National Laboratory's PVPMC [17]. This library was developed to simulate the solar irradiance given at any geographic coordinates. The function of this library is to provide solar metrics for the realistic simulation of solar harvesting. The library extends to simulate the efficiency and positioning of solar farms. We will also set up two baselines to compare with the proposed DQNN. First, Q-Learning QL was implemented for comparison versus its successor, the DQNN. Next, a genetic algorithm was implemented using the PyGAD library [18] to adjust the weights of the deep neural network.

Our environment was initialized with an environment of 1e8 km^2 size in which 50 WS were randomly dispersed, 10 CHs were placed and 1 UAV patrolled. WS were initialized with 2500 mm^2 solar cells and 1-Ah, 10-hour-rate batteries. CHs, by contrast, were initialized with 5000 mm^2 solar panels and 2-Ah, 10-hour-rate batteries. AmBC specifics were determined to be a max distance of 2 km incurring a 70 μA current draw when transmitting data. In contrast, LoRaWAN was set at a maximum distance of 5000 m and current draws of 1.6 μA idle, 14.22 mA receiving, and 38.9 mA transmitting. Finally, the UAV had the specifications of 15 m/s speed, 6800 mAh battery capacity, 2.5-hour-rate charging, and 0.5-hour-rate flight. The thresholds were chosen before running instead of utilizing tunable parameter options. P_{uav_thresh} was set to 40% maximum battery capacity, D_{thresh} was set to 1 kB, the maximum number of steps was set to 720 per episode and 1e5 for all training steps, a batch size of 64 steps, a deque size of 2500, a D_{scale} of 1e4, $T_{threshold}$ = 60 minutes, the hard update timer is set to 720 and the gamma is defined at 0.95.

B. Experimental Evaluation

1e5 steps was the time limit set for training divided between 5e4 pre-operation steps and 5e4 evaluation steps. The experiments produced an average episode length of 112 to 114 minutes as seen in figure 2. With a maximum episode length of 720 minutes, this indicates the constraint that power has on the system and the limitation it imposes upon the simulation. The return also shows interesting trends as all models gravitated towards rewards between -7e3 and -10e3 as depicted in figure 3. Which indicates a potential focus for future work.

Utilizing the data metrics explained prior, the average data collected per step reward can be seen in Figure 4. For this metric, the algorithm shows 33% better results with the implemented DDQN than with its closest competitor. This shows the model prioritizing the total collection of data for maximizing

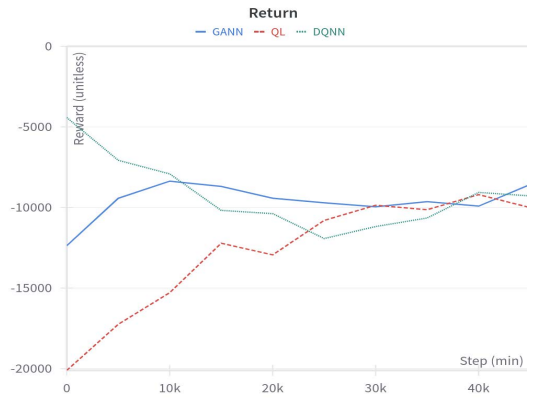

Fig. 3: The average episodic return per each evaluation period.

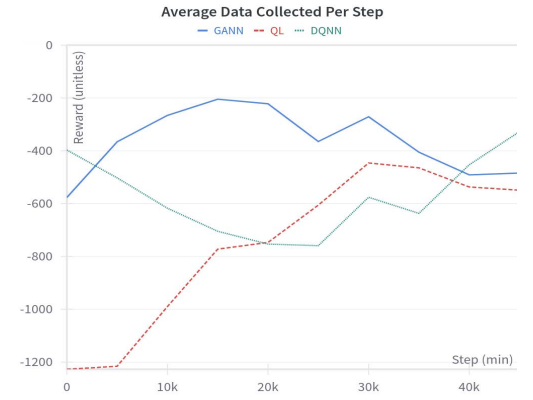

Fig. 4: The average episode length at each evaluation for each of the three algorithms used.

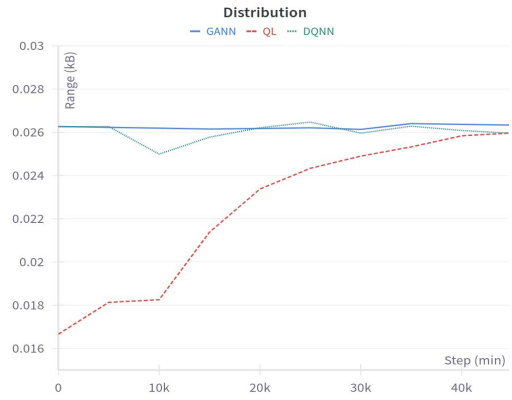

Fig. 5: The average difference between most data contributed by a CH and least during evaluation.

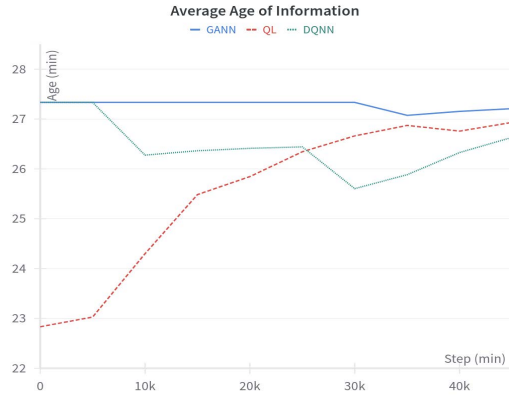

Fig. 6: The average AoI for each evaluation.

reward. The distribution in Figure 5 was determined by the difference between the maximum data collected from a CH and the minimum. All models performed well at minimizing the bias between CHs with distributions ending between 25 and 26 Bytes of data difference.

Finally, the AoI is recorded in the last two Figures 6 and 7. Figure 6 shows the average AoI across CHs and reveals the effectiveness of the algorithm utilizing DDQN to minimize this age by 1.8% beyond that of the other models. The ability of the algorithm to handle AoI is further revealed in 7 by showing the peak age of information almost 1.3% lower for the algorithm implemented with a DRL model versus the other models. Typically, DRL performs better with more training. By increasing steps from 1e5 to 1e6, the algorithm could be predicted to perform more optimally.

VI. Conclusion

In this paper, we discussed a complex simulation involving power as the primary constraint on the simulation. Additionally, metrics were defined that measure the data accumulation, distribution, and age. Additionally, we define a model that addresses these metrics and compare them to a baseline as proof of its reliability. Future research will further explore the application of DRL to this simulation and real-time UAV-assisted IoT networks. Other directions into the production of increasingly realistic simulations for the assistance of developing models for real-time applications.

References

[1] Chen Pan, Mimi Xie, and Jingtong Hu. Maximize energy utilization for ultra-low energy harvesting powered embedded systems. In *2017 IEEE 23rd International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 1–6. IEEE, 2017.

[2] Chen Pan, Mimi Xie, Song Han, Zhi-Hong Mao, and Jingtong Hu. Modeling and optimization for self-powered non-volatile iot edge devices with ultra-low harvesting power. *ACM Trans. Cyber-Phys. Syst.*, 3(3), August 2019.

[3] Zhenyu Zhou, Chuntian Zhang, Jingwen Wang, Bo Gu, Shahid Mumtaz, Jonathan Rodriguez, and Xiongwen Zhao. Energy-efficient resource allocation for energy harvesting-based cognitive machine-to-machine communications. *IEEE Transactions on Cognitive Communications and Networking*, 5(3):595–607, 2019.

[4] David W Matolak and Ruoyu Sun. Air–ground channel characterization for unmanned aircraft systems—part i: Methods, measurements, and models for over-water settings. *IEEE Transactions on Vehicular Technology*, 66(1):26–44, 2016.

[5] Yong Zeng and Rui Zhang. Energy-efficient uav communication with trajectory optimization. *IEEE Transactions on Wireless Communications*, 16(6):3747–3760, 2017.

[6] Cheng Zhan, Yong Zeng, and Rui Zhang. Energy-efficient data collection in uav enabled wireless sensor network. *IEEE Wireless Communications Letters*, 7(3):328–331, 2017.

[7] Jie Gong, Tsung-Hui Chang, Chao Shen, and Xiang Chen. Aviation time minimization of uav for data collection from energy constrained sensor networks. In *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6. IEEE, 2018.

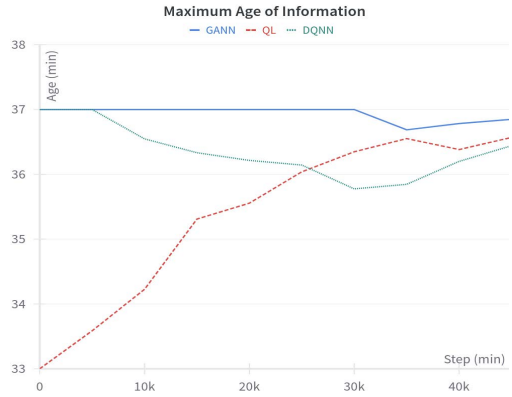

Fig. 7: The peak AoI for each evaluation.

[8] Wen Zhang, Wenlu Wang, Mehdi Sookhak, and Chen Pan. Joint-optimization of node placement and uav's trajectory for self-sustaining air-ground iot system. In *2022 23rd International Symposium on Quality Electronic Design (ISQED)*, pages 1–6, 2022.

[9] Sarder Fakhrul Abedin, Md Shirajum Munir, Nguyen H Tran, Zhu Han, and Choong Seon Hong. Data freshness and energy-efficient uav navigation optimization: A deep reinforcement learning approach. *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[10] Zekun Jia, Xiaoqi Qin, Zijing Wang, and Baoling Liu. Age-based path planning and data acquisition in uav-assisted iot networks. In *2019 IEEE International Conference on Communications Workshops (ICC Workshops)*, pages 1–6. IEEE, 2019.

[11] Kun Chen and Longbo Huang. Age-of-information in the presence of error. In *2016 IEEE International Symposium on Information Theory (ISIT)*, pages 2579–2583, 2016.

[12] M. Theile, H. Bayerlein, R. Nai, D. Gesbert, and M. Caccamo. Uav path planning using global and local map information with deep reinforcement learning. In *2021 20th International Conference on Advanced Robotics (ICAR)*, pages 539–546, 2021.

[13] Yilan Li, Haowen Fang, Mingyang Li, Yue Ma, and Qinru Qiu. Neural network pruning and fast training for drl-based uav trajectory planning. In *2022 27th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 574–579, 2022.

[14] Mohamed A Abd-Elmagid, Aidin Ferdowsi, Harpreet S Dhillon, and Walid Saad. Deep reinforcement learning for minimizing age-of-information in uav-assisted networks. In *2019 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2019.

[15] S. Fu, Y. Tang, Y. Wu, N. Zhang, H. Gu, C. Chen, and M. Liu. Energy-efficient uav-enabled data collection via wireless charging: A reinforcement learning approach. *IEEE Internet of Things Journal*, 8(12):10209–10219, 2021.

[16] Vincent Liu, Aaron Parks, Vamsi Talla, Shyamnath Gollakota, David Wetherall, and Joshua R. Smith. Ambient backscatter: wireless communication out of thin air. *SIGCOMM Comput. Commun. Rev.*, 43(4):39–50, aug 2013.

[17] William F. Holmgren, Clifford W. Hansen, and Mark A. Mikofski. pvlib python: a python package for modeling solar energy systems. *Journal of Open Source Software*, 3(29):884, 2018.

[18] Ahmed Fawzy Gad. Pygad: An intuitive genetic algorithm python library, 2021.