CISTER
Research Centre in
Real-Time & Embedded
Computing Systems

Journal Paper

Online Velocity Control and Data Capture of Drones for the Internet-of-Things: An Onboard Deep Reinforcement Learning Approach

Kai Li*
Wei Ni
Eduardo Tovar*
Abbas Jamalipour

*CISTER Research Centre

CISTER-TR-201106

2020

Online Velocity Control and Data Capture of Drones for the Internet-of-Things: An Onboard Deep Reinforcement Learning Approach

Kai Li*, Wei Ni, Eduardo Tovar*, Abbas Jamalipour

*CISTER Research Centre

Polytechnic Institute of Porto (ISEP P.Porto)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8321159

E-mail: kai@isep.ipp.pt, Wei.Ni@data61.csiro.au, emt@isep.ipp.pt

https://www.cister-labs.pt

Abstract

Applications of Unmanned Aerial Vehicles (UAVs) for data collection are a promising means to extend Internet-of-Things (IoT) networks into remote and hostile areas, and areas with no access to power supplies. Adequate design of velocity control and communication decisions of UAVs is critical to minimize the data packet losses of ground IoT nodes resulting from overflowing buffers and transmission failure. However, online velocity control and communication decision-making is challenging in UAV-enabled IoT networks, due to the lack of the up-to-date knowledge on the state of the IoT nodes, e.g., battery energy, buffer length and channel conditions, at the UAV. Current methodology using reinforcement learning complements real-time solutions to small-scale decision problems in static IoT networks. However, reinforcement learning is impractical for the online velocity control and communication decision in the UAV-enabled IoT network, due to its rapidly growing complexity (also known as the curse-of-dimensionality). This article discusses the design of onboard deep Q-network to deliver the online velocity control and communication decision of UAVs. The onboard deep Q-network can jointly determine the optimal patrol velocity of the UAV and decide the IoT node to be interrogated for data collection, thereby minimizing asymptotically the data packet loss of the IoT networks.

Online Velocity Control and Data Capture of Drones for the Internet-of-Things: An Onboard Deep Reinforcement Learning Approach

Kai Li, *Senior Member, IEEE,* Wei Ni, *Senior Member, IEEE,* Eduardo Tovar, *Member, IEEE,* and Abbas Jamalipour, *Fellow, IEEE*

***Abstract*—Applications of Unmanned Aerial Vehicles (UAVs) for data collection are a promising means to extend Internet-of-Things (IoT) networks into remote and hostile areas, and areas with no access to power supplies. Adequate design of velocity control and communication decisions of UAVs is critical to minimize the data packet losses of ground IoT nodes resulting from overflowing buffers and transmission failure. However, online velocity control and communication decision-making is challenging in UAV-enabled IoT networks, due to the lack of the up-to-date knowledge on the state of the IoT nodes, e.g., battery energy, buffer length and channel conditions, at the UAV. Current methodology using reinforcement learning complements real-time solutions to small-scale decision problems in static IoT networks. However, reinforcement learning is impractical for the online velocity control and communication decision in the UAV-enabled IoT network, due to its rapidly growing complexity (also known as the curse-of-dimensionality). This article discusses the design of onboard deep Q-network to deliver the online velocity control and communication decision of UAVs. The onboard deep Q-network can jointly determine the optimal patrol velocity of the UAV and decide the IoT node to be interrogated for data collection, thereby minimizing asymptotically the data packet loss of the IoT networks.**

***Index Terms*—Unmanned aerial vehicle, Internet-of-Things, Velocity control, Data capture, Deep reinforcement learning**

I. UAV-enabled IoT Networks

A. Background

Recent advances in scalable Internet-of-Things (IoT) networks and renewable energy have enabled the deployment of a large number of energy harvesting powered IoT nodes over large areas with limited persistent power supply, for sustainable sensing of weather, environmental pollutions, or traffic and road conditions. The IoT nodes can progressively harvest energy from multiple renewable energy sources, e.g., solar panel [1], electret-based wind turbine [2], or wireless power transfer [3]. Sensory data can be collected from the energy harvesting powered IoT nodes. For assisting data collection in IoT networks, Unmanned Aerial Vehicles (UAVs), also known as drones, can act as an aerial platform to communicate with the IoT nodes and capture valuable environmental data.

A UAV can be employed to visit the large number of IoT nodes deployed in the area of interest, and move sufficiently close to the node that is scheduled to transfer data, thanks to UAVs' excellent mobility and maneuverability [4]. The UAV assists the ground nodes in performing task computing after the ground nodes offload their computation tasks to the UAV [5].

As a long-range wireless communication technology, low-power wide area network (LPWAN) has been deployed to provide connectivity to IoT networks in rural areas, e.g., in the agricultural and farming sectors. LPWAN provides long-range transmissions of up to 10 km at the expense of low data rates (typically in the order of tens of kilobits per seconds) and subsequently high transmission latencies [6]. On the contrary, the UAV can physically approach each individual IoT node by exploiting the excellent mobility and flexibility of a UAV. A short, line-of-sight (LoS)-dominant communication link between the UAV and an IoT node can enjoy a significant channel gain and support high-speed data transmission [7].

B. Case study: road surveillance

A large number of IoT nodes powered by renewable energy sources can be deployed to monitor road traffic and emergency [8]. Every IoT node is equipped with solar panels, wind power generators, or wireless power receivers to harvest energy to power its operations. In particular, the battery energy of the IoT nodes can be drastically different from each other, depending on the ambient environmental conditions of the individual nodes. Each IoT node generates data packets at an application-specific sampling rate, and buffers the data packets awaiting transmission.

Figure 1 provides an example of the use of a UAV to the surveillance of traffic violations, accidents or other road emergency, where the IoT nodes can be lightweight cameras and portable tachymeters deployed on roadside to take videos or images and collect environmental data. The UAV equipped with a wireless transceiver and an onboard processor is instructed to fly over an unpopulated area with little to no 5G service but a need for bursty transmissions of high-bandwidth data, e.g., high-resolution images or videos. The UAV can select the IoT nodes to collect their data. Moreover, the UAV can decide to either forward any urgent data immediately to a remote command post; or buffer other data until the next time it passes and offload the data to the command post. In most cases, the urgent data which requires a fast response is small in size, while other data could be large and can benefit from the relatively short-range transmissions between the UAV and the command

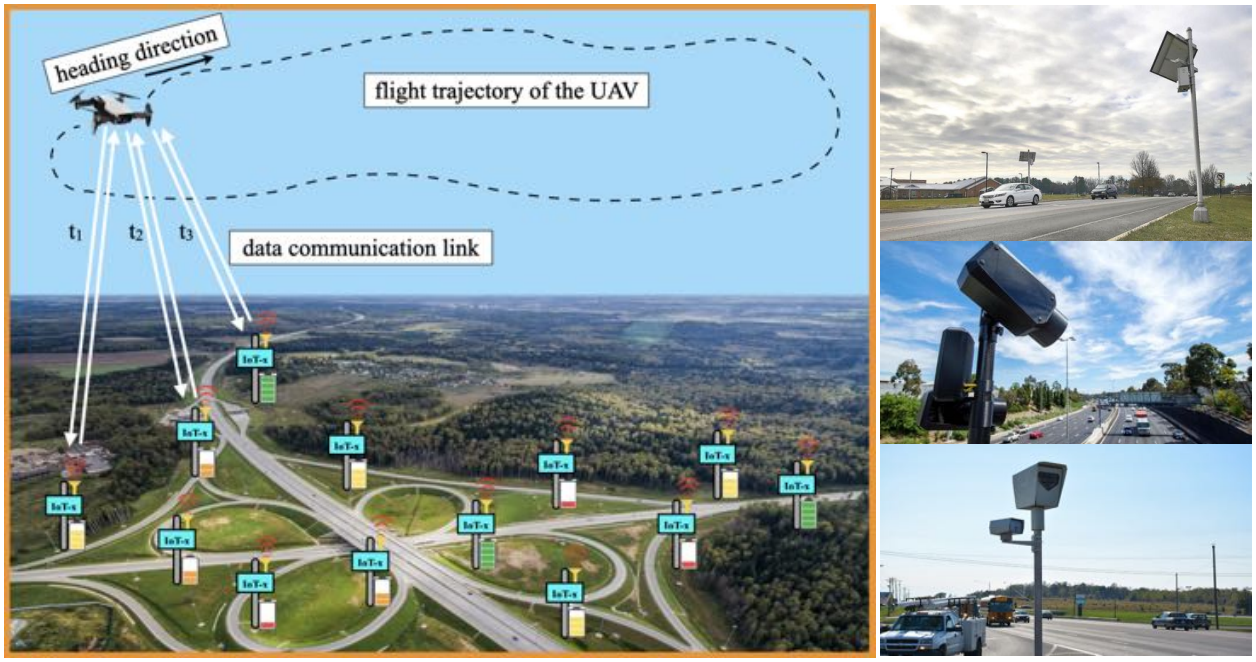


Fig. 1: UAV-enabled IoT networks deployed in unpopulated regions, where a large number of IoT-x nodes serve as data sources with sensing ability, e.g., monitoring the traffic violations, accidents or other road emergency. The UAV is employed to collect the data from the IoT nodes at different time.

post when the UAV is passing the command post and enjoying an excellent channel condition.

The flight trajectory of the UAV can be carefully planned beforehand, so that all the IoT nodes are accessible to the UAV [9]. For the example of Figure 1, the UAV can fly along the highway. The UAV can carry out a velocity control and data capture (VCDC) strategy online to decide its cruise speed (by selecting the next waypoint), and the next IoT node which will upload its data to the UAV as the UAV is approaching the next waypoint. The UAV sends a short beacon message to notify the selected IoT node of the decisions. The node can also measure the signal-to-noise ratio (SNR) of the channel from the beacon message. The state information of the selected IoT node, i.e., the battery level, data queue length and channel condition, can be put in a control segment of the data packet that is transmitted to the UAV.

C. Motivation and contributions

This article is motivated to address the research challenges and opportunities of online UAV velocity control and IoT transmission schedule. Specifically, we formulate a discrete-time Markov Decision Process (MDP) to minimize the network cost resulting from the buffer overflow and failed transmissions of the IoT nodes. Deep reinforcement learning is applied onboard to learn the optimal velocity control and IoT node selection strategies of the UAV at every MDP state.

As shown in Figure 1, the IoT nodes with finite data buffers undergo random data arrivals. Selecting an IoT node for data collection may result in a buffer overflow at other unselected nodes, since new data arrivals at those nodes may have to be dropped if their buffers are already full and overflow. Particularly, selecting an IoT node with a poor channel condition gives rise to the packet errors of its transmissions or the buffer overflows of other nodes. Moreover, the UAV may not have the complete, instantaneous knowledge of every IoT node, due to limited radio coverage of the IoT nodes. The IoT nodes can only report their states when polled, and send the states together with their respective data. In this sense, it is important to design (or more adequately, learn online) the appropriate velocity control and communication schedule of the UAV, which captures the underlying structure of the data and energy arrivals at the IoT nodes and adapts the speed and the node selection of the UAV to the structure.

II. Challenges and Opportunities

A. What is velocity control and data capture?

Given the movement of the UAV, the time-varying wireless channel between each IoT node and the UAV can suffer from independent signal fading. Having an IoT node transmit during the instants when the channel quality is poor is likely to result in packet reception errors at the UAV. Moreover, the IoT nodes with random data arrivals buffer the data to be collected in the data queue with a finite buffer size. The new data packets have to be dropped when the data queue overflows.

The harvested energy from solar, wind, wireless power transfer, or other renewable energy sources can be affected by cloudy or rainy weather, windless environment, or interference from existing wireless networks. As a result, the battery energy of the IoT nodes can be time-varying and substantially differ from each other. Some IoT nodes can

deplete the battery energy quickly before the data packets in its queue are exported.

In such UAV-enabled IoT networks, it is critical to jointly determine the patrol velocity of the UAV and schedule the data transmission of the IoT nodes, so as to minimize the packet losses resulting from overflowing buffers and channel fading. Inadequate VCDC at the UAV can schedule some of the IoT nodes with short data queue lengths and poor link qualities to transmit. The battery energy of these IoT nodes drains (due to packet (re)transmissions), while some others with good link qualities but not promptly scheduled can have their buffers overflow. Furthermore, in practice, the complete up-to-date knowledge of the battery level and data queue length of all the IoT nodes is not available at the UAV due to non-negligible signaling delay and overhead. This makes the online UAV velocity control and IoT transmission schedule non-trivial.

B. State of the art

The motion of a UAV is designed to assist a point-to-point communication between the UAV and a ground device [10], by taking into account the propulsion energy consumption of the UAV. An algorithm is developed to improve energy efficiency, subject to the constraints on the UAV's trajectory, including its initial/final locations and velocities, and maximum speed. In [11], the UAV trajectory planning with the velocity control is exploited for charging the ground sensors. The problem formulation and solution imply that the hovering location and duration can be designed to enhance the wireless power transfer efficiency. In [12], the UAV is employed to provide emergent data communications to a remote area. A UAV deployment algorithm is studied to control the flight velocity and altitude to reduce deployment time of the UAV with guaranteed network coverage. The existing trajectory planning and communication scheduling approaches in the literature use offline deterministic optimization theories to improve the network coverage or reduce the energy consumption of the UAV. Based on the specific statistical distributions, the transitions of the network states are formulated as a probabilistic random process. In contrast, this article focuses on the online VCDC problem which minimizes the data loss caused by buffer overflows at the IoT nodes and poor channels, where the UAV has no a-priori knowledge of the state dynamics.

In our recent works [9] and [13], we start to explore learning-based joint optimization of the UAV speed and IoT transmission. In [9], we study an energy-efficient data relaying scheme to balance the battery lifetime of the UAVs, where the UAV moves at a predetermined velocity. In [13], a double Q-learning-based scheduling algorithm is developed to select the IoT node for data collection and wireless power transfer along the flight trajectory of the UAV. This article generalizes the ideas of [9] and [13] to jointly optimize the UAV speed and communication schedule in the absence of the complete instantaneous knowledge of the IoT nodes at the UAV. Deep reinforcement learning is applied onboard to learn the underlying characteristics of data and energy arrivals at the IoT nodes, and deliver the optimal velocity control and communication schedule in real-time.

C. Markov decision process for velocity control and data capture

The trajectory and altitude (i.e., the waypoints) of the UAV are predesigned to allow the UAV to patrol over the area where the IoT nodes are distributed. The optimization of the UAV velocity control and IoT transmission schedule needs to be conducted in real-time over the entire VCDC process. The correlation between the scheduling decisions of different time slots needs to be captured, and to validate the long-term optimality of VCDC. For this reason, a discrete-time MDP is typically the suitable model to discretize the VCDC problem with the network states consisting of the battery energy and the data queue length of the IoT nodes, channel conditions between the UAV and the IoT nodes, and waypoints of the UAV. The UAV can take the actions to control the instantaneous patrol velocity, and select the IoT nodes. The future battery energy and data queue length of every IoT node can be affected by the VCDC decisions of the UAV, which leads to a non-negligible impact on the future actions of the UAV. The actions of VCDC can be modeled as a discrete-time random process since the data arrival and queueing process of sensory data at every IoT node are random and independent. Particularly, the actions of VCDC are also partially controllable at the UAV.

Note that the VCDC problem is different from travelling salesman problem (TSP) in the sense that all the possible waypoints are daisy chained in a loop and the VCDC strategy decides the next waypoint of the UAV for the next time slot on-the-fly. In other words, the VCDC strategy decides the patrol speed of the UAV in real-time. This is done by the UAV flying along the loop repeatedly and learning online the underlying patterns of the data and energy arrivals at all the IoT nodes.

The VCDC actions of the UAV can be optimized in a long-term stochastic control process, where the optimality is achieved in regards of a specific metric, e.g., packet loss stemming from both overflowing buffers and unsuccessful data transmissions of the IoT nodes. The optimal policy of the MDP can be obtained by taking classical approaches, e.g., value iteration or policy iteration [3]. In particular, the value iteration iteratively optimizes an estimate of the action-value function, while the policy iteration updates the policy at each step and obtains the action-value function with this new policy. It is assumed that the transition of the network states and the packet loss at the states are known to the UAV in prior. The value/policy iteration method repeatedly updates the estimate of the optimal action-value function according to the Bellman optimality equation. When the Bellman optimality equation converges, the cost function is minimized. The MDP model is stabilized. However, the action-value function of the MDP can only be

evaluated offline since the UAV has the prior knowledge on the transition of the network states and the packet loss at all the states, which is not applicable to the online VCDC problem.

Q-learning, one of the most popular reinforcement learning techniques, can minimize the expected long-term accumulated discounted costs (i.e., the expected packet loss of the IoT nodes) in small-scale static IoT networks [14]. Q-value of the action-value function can be learned as the expected accumulated discounted cost when the UAV takes an action following one of the VCDC strategies thereafter. The future actions of the UAV can be determined by using the gained experience and the current action-value function. By learning the action-value function, the optimal scheduling strategy can be obtained without the transition and/or cost functions. However, Q-learning is known to suffer from the well-known curse-of-dimensionality, which is impractical for the online VCDC problem due to a large number of network states and actions in UAV-enabled IoT networks.

In contrast, this paper aims to enable autonomous UAVs to control online its own velocity and its selection of IoT nodes to transmit data to the UAVs, so as to minimize the loss of valuable IoT data. Each network state of the MDP consists of waypoints of the UAV on the trajectory, the battery levels and data queue lengths of the IoT nodes. The VCDC actions of the MDP are the instantaneous patrol speed of the UAV, and the selection of IoT nodes for data transmission. The state space and the action space of VCDC can be exceedingly large and grow increasingly fast with the number of MDP states and actions, hence the dynamic programming based approach in [3] and the reinforcement learning algorithm in [14] are not applicable to the VCDC problem in UAV-enabled IoT networks.

III. Onboard deep reinforcement learning for online VCDC

A. Architecture of onboard deep reinforcement learning

To circumvent the curse-of-dimensionality problem of reinforcement learning, deep reinforcement learning can be developed according to two typical neural network models, i.e., convolutional neural network (CNN) or deep Q-network. A CNN architecture can contain a bunch of convolution, pooling, and fully connected layers. To construct a CNN, numerous parameters have to be determined in priori, such as the number of layers, the order of layers, and the type of each layer. The setting of the parameters can make the CNN architecture large and, hence, result in high complexity of the design and implementation.

Deep reinforcement learning can be used to develop a new onboard deep Q-networks (ObDRL) for the online control and plan for VCDC. The ObDRL scheme can be designed to minimize the data packet loss of the entire system by training the onboard deep Q-network at the UAV. The onboard deep Q-network can jointly optimize the instantaneous patrol velocity of the UAV and the selection of the IoT nodes, with extended state and action spaces of the above-mentioned MDP.

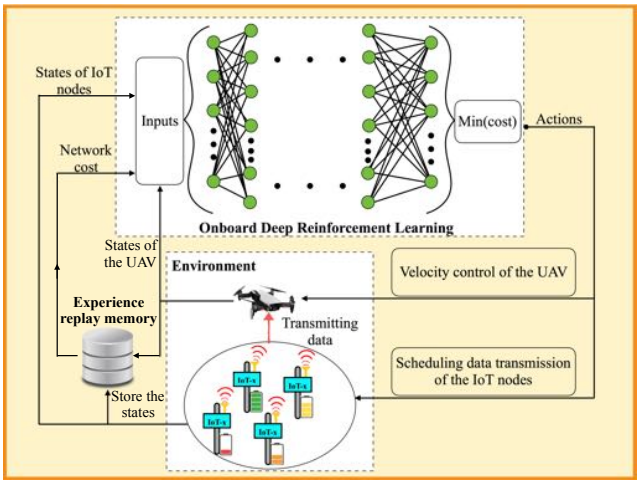


Fig. 2: Architecture of the onboard deep Q-networks for jointly optimizing the instantaneous patrol velocity of the UAV and the selection of the IoT node for the data transmission.

Figure 2 shows the architecture of the onboard deep Q-network. The network cost is the packet loss resulting from the buffer overflow and failed transmissions of the IoT nodes. The UAV observes the current network state, i.e., the battery energy and data queue length of the selected IoT nodes, the channel quality, and the waypoint of the UAV, from the real-world environment. Deep reinforcement learning can train the deep Q-network by taking the current network state and immediate network cost as the input and evaluating the corresponding Q-value. The onboard deep Q-network can approximate the Q-value to minimize the network cost by learning the optimal VCDC strategies of the UAV at each network state. Moreover, the onboard deep Q-network can measure the quality of the VCDC strategy in a given network state of the MDP model. When the optimal velocity is allocated to the UAV and the optimal IoT node is scheduled to transmit data in the environment, new network states and cost can be obtained and used as the next inputs to further train the onboard deep Q-network.

B. Onboard deep Q-networks for online VCDC

Deep reinforcement learning can be conducted over multiple episodes to approximate the minimum of the Q-values in the deep Q-network, by adapting a set of learning weights. Each episode is a number of consecutive time epochs, where the deep Q-network is trained to find the optimal actions for the VCDC (i.e., the velocity of the UAV and transmission schedule of the IoT nodes). As shown in Figure 2, in every episode, the deep Q-network updates the learning weight at the current network state to minimize the mean-squared Bellman error, by optimizing an immediate network cost. By learning and minimizing the network cost iteratively, ObDRL can achieve the optimality of velocity control and transmission scheduling asymptotically, with the growing state and action sizes/spaces.

Experience replay can be carried out with deep reinforcement learning to randomize over the network states and

the actions of the UAV at each episode in the deep Q-network [15]. The network states and actions for online VCDC at every episode are stored in a dataset on the UAV, pooled over many episodes into the memory allocated for the experience replay. The experience in the deep Q-network contains a number of samples (or minibatches) and can be accordingly updated during the learning. The experience replay can remove oscillations or divergence in the observation of the network states from the environment and smooth over changes in the data distribution, thereby reducing the variance of learning updates in the deep reinforcement learning.

An ϵ-greedy policy can be utilized to balance the network cost minimization with respect to the UAV velocity control and IoT transmission scheduling decisions already known with trying some new actions of the UAV to obtain the knowledge previously unknown. Specifically, the UAV can randomly determine its instantaneous velocity and the IoT node for data transmission according to a probability ϵ. This explores the unknown knowledge of VCDC. Meanwhile, the network cost can still be minimized based on the knowledge of VCDC already learned with $1 - \epsilon$.

IV. Performance Enhancement

A. Implementation of ObDRL

The data packet generation rate of the IoT node is 100 data packets, where each data packet has 128 bytes. The required bit error rate (BER) of the channel can be 0.05%, however, the BER can be configured depending on the traffic type and quality-of-service (QoS) requirement of the applications, as well as the transmission capability of the UAV. The battery readings are continuous variables with variance difficult to be traced in real-time. Therefore, to improve the tractability of the performance and for illustration convenience, the battery capacity of the IoT node is discretized to 50 levels, where the battery readings can be lower rounded to the closest discrete level. The replay memory size can be 5000. The transmit power of the UAV can be set to 100 milliwatts. The simulation parameters are also listed in Table I.

TABLE I: TensorFlow configurations

Parameters	**Values**
Packet generation rate	100
Packet size	128 bytes
Required BER	0.05%
Battery levels	50
Replay memory size	5000
Transmit power of the UAV	100 mW

ObDRL is implemented in Python 3.5 by using Keras deep learning library with Google TensorFlow as the backend engine. Three fully-connected hidden layers are created by using *tensorflow.layers.dense(inputs, dimensionality of the output space, activation function)*. Then, an optimizer function *tensorflow.train.AdamOptimizer().minimize(loss function)* is called to minimize the loss function. The optimizer is imported from the Keras library. For online training the ObDRL, the memory stores the learning outcomes, a.k.a. experience at every step, using the quadruplet <state, action, cost, next state>. The memory is updated by calling the function *memory.add sample(state, action, cost, next state)*, and the experiences are retrieved by calling the function *memory.sample(batch size)*.

B. Numerical analysis

For performance comparison, we also simulate three existing techniques as the performance benchmarks, namely, Constant Velocity Data Queue (CVDQ), Constant Velocity Highest Channel Quality (CVHC), and Constant Velocity Random Scheduling (CVRS). CVDQ is a greedy policy based on the data queue lengths of the IoT nodes, where the UAV maintains the constant velocity. Assuming hypothetically that the data queue lengths of the IoT nodes within the radio range of the UAV are known to the UAV (which is not the case in the proposed ObDRL approach). The IoT node with the longest queue is selected to transmit data. CVHC is a greedy policy based on the channel qualities of the IoT nodes, where the node with the highest SNR is selected to transmit data. CVRS schedules randomly an IoT node at a time to transmit data, and the VCDC decision is independent of the battery level, queue length and channel condition of the IoT node, and the UAV's velocity.

Figure 3 presents the network cost of ObDRL with regards to the episodes (i.e., learning time) given the discount factor of 0.99. The number of IoT nodes and the maximum length of their data queues are set to 300 nodes and 20 packets, respectively. It can be observed in the figure that the network cost of ObDRL that takes advantage of deep reinforcement learning substantially drops from episode 1 to episode 350. In particular, the network cost with ObDRL quickly falls within the initial 70 episodes. The performance of ObDRL converges within about 360 episodes, and remains relatively stable afterwards. This is because the deep Q-network gets trained at the beginning of the learning process. After a number of episodes, ObDRL can store a set of the MDP states and the corresponding actions as the learning experience in the replay memory. Based on the experience, the learning weights in ObDRL can be optimally updated to approximate the action outputs of the UAV for VCDC more accurately.

In Figure 3, we also observe that ObDRL outperforms CVDQ, CVHC, and CVRS by 75.5%, 77.3%, and 84.9%, respectively. This is because ObDRL can be adequately trained to optimize the VCDC decisions of the UAV with an increasing number of episodes. At every learning iteration, the deep Q-network can reduce the mean-squared Bellman error, by minimizing the loss function.

Figure 4 shows the network cost of ObDRL with an increasing number of IoT nodes. As observed, ObDRL can reduce the network cost to a great extent when the maximum queue length of the IoT nodes increases from 10 to 50. Moreover, ObDRL achieves a lower packet loss than CVDQ, CVHC, and CVRS, while the performance

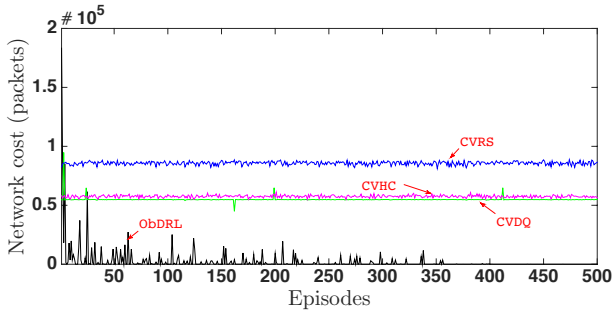


Fig. 3: Network cost (i.e., data packet loss) with the increase of the episodes.

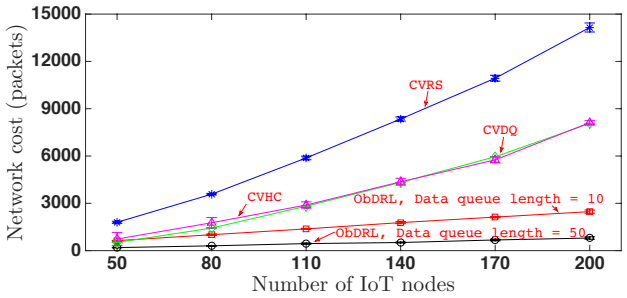


Fig. 4: Network cost with the onboard deep Q-network in terms of number of IoT nodes.

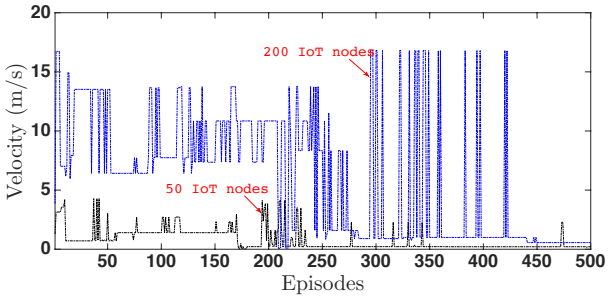


Fig. 5: Patrol velocity of the UAV with ObDRL.

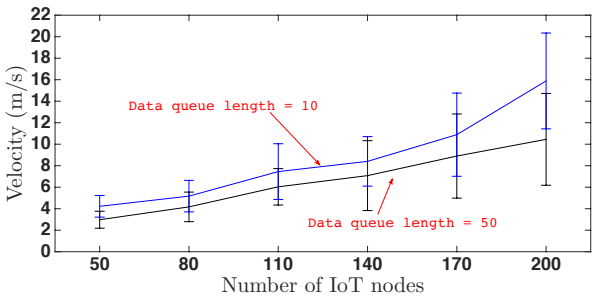


Fig. 6: Patrol velocity of the UAV with regards to number of IoT nodes.

gains keep growing with the network size. The reason is that ObDRL learns the pattern of the IoT nodes' energy consumption and data queue states to make the VCDC decisions which minimize the data packet loss of the entire network.

Figure 5 shows the patrol velocity control on the UAV by applying ObDRL, with the increasing number of learning episodes. It can be observed that the velocity of the UAV varies drastically at the beginning of the training process of ObDRL. With an increasing number of episodes, ObDRL learns the network dynamics and the variation of the velocity is significantly reduced. The results in Figure 5 also reveal a significant insight that the velocity of the UAV is adjusted frequently when the number of IoT nodes is large, given the same data queue length of the IoT nodes. This is due to the fact that the states of the IoT nodes exhibit more variations and changes in a large-scale ground IoT network than a small-scale one. The UAV has to accelerate and decelerate more frequently to minimize the overall packet loss of all the IoT nodes in the network.

Figure 6 shows the patrol velocity of the UAV with regards to the number of IoT nodes. As observed, the patrol velocity raises with an increase of network size. This is because the UAV has to accelerate the flight for collecting data from more IoT nodes in order to reduce buffer overflow. Furthermore, increasing the maximum data queue length of the IoT node can reduce the instantaneous velocity of the UAV. A larger data queue of the IoT node can hold more data packets. This allows an extended data transmission time between the IoT node and the UAV, which can slow down the UAV's flight.

V. Conclusions

In this article, design of online velocity control and data capture decision in UAV-enabled IoT networks was presented. An architecture of onboard deep Q-network was built, which can minimize the overall data packet losses of the IoT nodes resulting from overflowing buffers and transmission failure. Instantaneous patrol velocity of the UAV and scheduling the data transmission of the IoT nodes were also optimally determined without up-to-date knowledge of the network states. It is found that for training the onboard deep Q-network, experience replay can be conducted with deep reinforcement learning to store the network states and actions at every learning episode. The onboard deep Q-network can be typically implemented by using Keras deep learning library with Google TensorFlow. Finally, scalability of the IoT network and data queue lengths of the nodes can have a strong impact on the velocity control and data capture of the UAV.

Acknowledgements

This work was partially supported by National Funds through FCT/MCTES (Portuguese Foundation for Science and Technology), within the CISTER Research Unit (UIDB/04234/2020); also by the Operational Competitiveness Programme and Internationalization (COMPETE 2020) under the PT2020 Partnership Agreement, through the European Regional Development Fund (ERDF), and by national funds through the FCT, within project(s) POCI-01-0145-FEDER-029074 (ARNET).

REFERENCES

[1] R. Haight, W. Haensch, and D. Friedman, "Solar-powering the Internet of Things," *Science*, vol. 353, no. 6295, pp. 124–125, 2016.

[2] Y. Wu, W. Liu, and Y. Zhu, "Design of a wind energy harvesting wireless sensor node," in *International Conference on Information Science and Technology (ICIST)*. IEEE, 2013, pp. 1494–1497.

[3] K. Li, W. Ni, L. Duan, M. Abolhasan, and J. Niu, "Wireless power transfer and data collection in wireless sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 3, pp. 2686–2697, 2018.

[4] J. Wang, C. Jiang, Z. Han, Y. Ren, R. G. Maunder, and L. Hanzo, "Taking drones to the next level: Cooperative distributed unmanned-aerial-vehicular networks for small and mini drones," *IEEE Vehicular Technology Magazine*, vol. 12, no. 3, pp. 73–82, 2017.

[5] F. Zhou, R. Q. Hu, Z. Li, and Y. Wang, "Mobile edge computing in unmanned aerial vehicle networks," *IEEE Wireless Communications*, vol. 27, no. 1, pp. 140–146, 2020.

[6] U. Raza, P. Kulkarni, and M. Sooriyabandara, "Low power wide area networks: An overview," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 855–873, 2017.

[7] N. H. Motlagh, T. Taleb, and O. Arouk, "Low-altitude unmanned aerial vehicles-based internet of things services: Comprehensive survey and future perspectives," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 899–922, 2016.

[8] R. Du, P. Santi, M. Xiao, A. V. Vasilakos, and C. Fischione, "The sensable city: A survey on the deployment and management for smart city monitoring," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1533–1560, 2018.

[9] K. Li, W. Ni, X. Wang, R. P. Liu, S. S. Kanhere, and S. Jha, "Energy-efficient cooperative relaying for unmanned aerial vehicles," *IEEE Transactions on Mobile Computing*, no. 6, pp. 1377–1386, 2016.

[10] Y. Zeng and R. Zhang, "Energy-efficient UAV communication with trajectory optimization," *IEEE Transactions on Wireless Communications*, vol. 16, no. 6, pp. 3747–3760, 2017.

[11] J. Xu, Y. Zeng, and R. Zhang, "UAV-enabled wireless power transfer: Trajectory design and energy optimization," *IEEE Transactions on Wireless Communications*, 2018.

[12] X. Zhang and L. Duan, "Fast deployment of UAV networks for optimal wireless coverage," *IEEE Transactions on Mobile Computing*, vol. 18, no. 3, pp. 588–601, 2018.

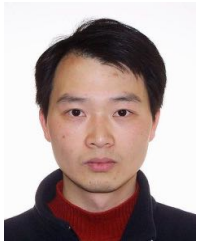
[13] K. Li, W. Ni, B. Wei, and E. Tovar, "Onboard double Q-learning for airborne data capture in wireless powered IoT networks," *IEEE Networking Letters*, vol. 2, no. 2, pp. 71–75, 2020.

[14] K. Li, W. Ni, M. Abolhasan, and E. Tovar, "Reinforcement learning for scheduling wireless powered sensor communications," *IEEE Transactions on Green Communications and Networking*, vol. 3, no. 2, pp. 264–274, 2018.

[15] K. Li, W. Ni, E. Tovar, and A. Jamalipour, "On-board deep Q-network for UAV-assisted online power transfer and data collection," *IEEE Transactions on Vehicular Technology*, 2019.

VI. ABOUT THE AUTHORS

Kai Li (S'09-M'14-SM'20) is a Senior Research Scientist and Project Leader at Real-Time and Embedded Computing Systems Research Centre (CISTER), Portugal. He focuses on wireless communications and security in Platoon-based Vehicular Cyber-Physical Systems. Contact him at kai@isep.ipp.pt.



Wei Ni (M'09-SM'15) is a Group Leader and Principal Research Scientist at Data61 Business Unit, CSIRO, Australia. He is also an Honorary Professor at Macquarie University (MQ) and the University of Technology Sydney (UTS). He also serves as the Chair of IEEE VTS NSW Chapter since 2020. Contact him at wei.ni@data61.csiro.au.

Eduardo Tovar is a Professor of Industrial Computer Engineering in the Computer Engineering Department, Polytechnic Institute of Porto (ISEP-IPP), and the director of CISTER, Portugal. Contact him at emt@isep.ipp.pt.

Abbas Jamalipour (S'86-M'91-SM'00-F'07) holds a PhD in Electrical Engineering from Nagoya University and currently he is the Professor of Ubiquitous Mobile Networking at the University of Sydney and President of the IEEE Vehicular Technology Society. He is a Fellow of the Institute of Electrical, Information, and Communication Engineers (IEICE) and the Institution of Engineers Australia, and an IEEE Distinguished Speaker. He has authored nine technical books, eleven book chapters, over 550 technical papers, and five patents in wireless communications. He has received a number of prestigious technical awards as well as 15 Best Paper Awards.