Evaluating Softwarization Gains in Drone Networks

Mohannad Alharthi* § , Abd-Elhamid M. Taha†, Hossam S. Hassanein*
*School of Computing, Queen's University, Kingston, Ontario, Canada. Emails: {harthi, hossam}@cs.queensu.ca
†Electrical Engineering Department, Alfaisal University, Riyadh, Saudi Arabia. Email: ataha@alfaisal.edu

GLOBECOM 2021 - 2021 IEEE Global Communications Conference | 978-1-7281-8104-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/GLOBECOM46510.2021.9685722

***Abstract*—Unmanned Aerial Systems (UASs) or drones are becoming increasingly dependable tools for many civil and industrial applications. Due to the increasing usage and capabilities of drones coupled with advances in innovative technologies and algorithms for managing and conducting tasks, drones are expected to crowd low-altitude airspace in urban areas. This brings many opportunities for service providers to provide drone-related services. Hence, efficient use of drones is required. In this paper, we investigate the benefits of reconfigurable softwarized drones operated by an entity or a service provider to perform tasks for its operations or for interested customers. We model a system of reconfigurable drones that can conduct multiple tasks per flight using Virtual Network Functions (VNFs) running on on-board capable computing systems. We compare our proposed model with alternatives with limited and no softwarization capabilities. Our evaluation demonstrates the performance gains due to reconfigurability in softwarized drone networks. Results show that softwarization allows drones to perform a variety of tasks using a limited number of reconfigurable drones and in a shorter time.**

***Index Terms*—Drones, UAVs, Softwarization, NFV**

I. Introduction

Unmanned Aerial Systems (UASs) or drones are considered essential tools in search and rescue missions, disaster relief efforts, industrial operations, remote sensing, aerial surveillance and security. The rate of growth of such applications is driving innovative technological advances. For instance, the Federal Aviation Administration (FAA) and NASA are developing UAS Traffic Management (UTM) a cloud-based service that provides automated services for drone operators, such as UAS registration, flight planning, airspace authorizations and provision of relevant information to UAS operators. As well, cost-effective and capable drones are becoming increasingly available. Furthermore, research activities are introducing more drone applications driven by machine learning and innovative technologies [1]. These also include using drones for network applications as relays, drone base stations and for offering computing services.

Softwarization of drones enables programmability and reconfigurability. Research efforts focused on integrating Software Defined Networking (SDN) and Network Functions Virtualization (NFV) into networked drones [2]. In addition to the flexible control and efficient utilization of resources, softwarization lends itself to leveraging on-board computing capabilities and enables intelligence and autonomy in conducting tasks. Softwarized drones with on-board computing systems can be reconfigured with different implementations of sophisticated ML algorithms to conduct tasks and process collected data [2]. Relevant results can be transmitted with little delay and without need for further processing. Task software can be implemented as swappable and upgradable Virtual Network Functions (VNFs), which can be reconfigured on-demand at deployment time or inflight, leading to more flexibility and reuse of the physical drone.

Using softwarization, an industrial entity, for instance, can efficiently utilize its fleet and reconfigure drones for its different operations while automating the process of scheduling tasks and deploying configured drones. The operator benefits from such flexibility by serving more tasks with a low turnaround time, and ultimately completing a batch of tasks in a short time. Such flexibility can bring an opportunity for service providers (SPs) to offer their fleet of softwarized drones to conduct tasks for customers such as industries requiring infrastructure inspection tasks, municipalities conducting aerial surveys and mapping missions, law enforcement, weather services, and academic institutions. The reconfigurability of such a system can translate to better efficiency and profitability for the SP. Such a service can also be more flexible from a regulations standpoint as the SP can obtain required authorizations and better integrate with UTM services.

In this paper, we investigate the benefits of reconfigurable softwarized drones used to conduct a variety of tasks. In doing so, we propose and model a system of softwarized drones that can conduct multiple tasks per flight using VNFs running on capable drone-mounted computing systems. We compare the gained efficiency of the system with alternatives with limited and no softwarization capabilities. This evaluation provides novel results quantifying the gains due to softwarization and reconfigurability of networked drones.

This paper is organized as follows. In the next section, we review the relevant literature. In section III, we provide an overview of the system operation, and in section IV, we detail the system model. In section V, we present the evaluation and results.

II. Related Works

Softwarization of drone-assisted wireless networks using SDN/NFV has been an active research area in the last few years [2]. SDN has been utilized to enhance the network performance of such highly dynamic systems. In this work, we focus on on-board computation using NFV, which has been utilized to deploy reconfigurable networks for a variety of applications.

§ Mohannad Alharthi is also affiliated with King Abdulaziz University (KAU), Jeddah, Saudi Arabia

978-1-7281-8104-2/21/$31.00 ©2021 IEEE

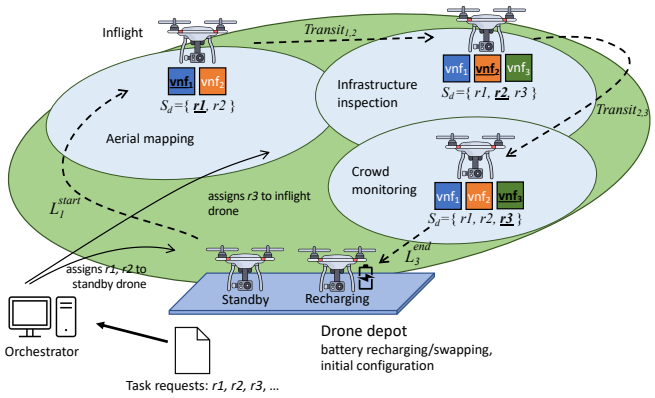


Fig. 1. Overview of the system components and the general operation including drone states, task assignments, and VNF activations. Underlines mark the active task and VNF. Dashed arrows indicate the trajectory of a flight, marked with requests transition distances as described in IV.

In [3], a configurable NFV-system for multi-drone ad-hoc networks is implemented. The system enables deploying several network services (e.g, VoIP) on the virtualized drone computing resources. Authors in [4] describe an SDN/NFV-based architecture for drone networks for rural zone monitoring. The Flying Ad hoc Network (FANET) provides video monitoring as a service where cameras on the ground and on drones capture and stream footage of monitored areas using specialized VNFs. In [5], we propose a scheme for planning the deployment of NFV-based drone networks hosting VNFs that process and delivers mission traffic flows using the virtualized on-board computing resources. Works such as [6] and [7] are recent instances of utilizing drone computing capabilities in the context of Multi-Access Edge Computing (MEC). To the best of our knowledge, none of the existing works evaluated direct gains attributed to softwarization and reconfiguration in networked drones compared to non-softwarized drones.

III. System Overview

Consider an entity or an SP that owns a fleet of drones and offers to perform various tasks and is stationed in an area with demand for such services. Such a system consists of a drone depot or station, a set of reconfigurable drones, and an orchestrator that controls the whole system and mainly is responsible for receiving and deploying tasks. The system is depicted in Figure 1. Parties interested in drones tasks submit task requests to the SP. Requests state the task requirements in terms of the trajectory, duration, and energy required. Requests also supply the software implementation of the task as VNFs or select from an available set of VNFs offered by the SP, which include VNFs that implement algorithms for conducting different tasks such as those shown in Figure 1. Such requests are handled by the orchestrator, which processes a queue of task requests and schedules tasks on the available fleet using an orchestration strategy. Drones, fitted with a common set of sensors and a virtualized computing system, are reconfigurable (or softwarized) and are loaded with VNF images corresponding to assigned tasks. Due to this flexibility, drones can be assigned a series of tasks to perform in a single flight. While a drone cruises along the combined trajectory of assigned tasks, VNF images corresponding to the active task are instantiated and terminated as the drone transitions from one task area to another. Figure 1 demonstrates this process.

Drones are stationed at the depot, where they initially get configured. The depot is equipped with facilities to connect to drone computing systems to configure them as instructed by the orchestrator. The depot also handles the charging, or battery swapping when drones land after performing tasks. The reconfigurability of drones permits the assignment of additional tasks while inflight, presuming a high data rate connectivity between the drone and the orchestrator.

The reconfigurability of drones is enabled by an on-board system with virtualization capability to host virtual machines (as containers or VNFs). The system should be designed so VNFs are given controlled access to drone sensors and the ability to specify a flight path for the duration of the task. The underlying system controls the time of activation of VNFs and their deactivation when a task exceeds its allotted time or energy as configured by the orchestrator.

The orchestrator oversees the operation of the system. It employs orchestration strategies to prioritize task requests in a queue and make assignment decisions to reconfigure drones with assigned tasks. Furthermore, it monitors the overall state of the system, including the deployment status, current locations, trajectories, and energy levels of drones. Once a number of tasks are assigned to a drone, the orchestrator generates the flight plan for the drone. However, tasks may choose to have their own flight paths for the duration when in control of the drone. The orchestrator may alter trajectories in response to errors, e.g., an unexpected battery drain or task software malfunction. The orchestrator is assumed to communicate such aspects using a well-defined control interface and channel, which is outside of the scope of this work.

IV. System Model

We denote by D the set of available drones. At any time instant, a drone can be in one of three states: standby, inflight (performing tasks), and recharging. After landing, a drone stays at the recharging state for a duration T_{charge} to charge or swap batteries, after which the drone goes into standby state. C_d denotes the fully charged energy capacity of d.

A task request r arriving to the orchestrator is represented by a tuple $\langle r, Dur_r, E_r, K_r, L_r^{start}, L_r^{end}, T_r^{arrival} \rangle$. Here, Dur_r is the time duration of the task, E_r is the energy required to fly while conducting the task, and K_r is the type of the task, which maps to the task VNF image. Furthermore, L_r^{start} and L_r^{end} represent the distances from the depot to the task area and from the end point of the task back to the depot, respectively, while $T_r^{arrival}$ is the arrival time of the request. R denotes the set of all requests. We denote by $Transit_{r,\bar{r}}$ a matrix holding the distances between locations of any pair of tasks r and $\bar{r}$. $Tr(.)$ denotes a function that calculates the time required for a drone to travel a given distance in

a predetermined speed. Figure 1 demonstrates the distances associated with task requests.

The orchestrator maintains a queue Q of all arrived task requests. The orchestrator engages its assignment procedure at the arrival of task requests or at the availability of standby drones in order to process Q and assign tasks to drones. At any time instant, a softwarized drone d, can be assigned a series of tasks denoted by S_d, with different tasks $r \in R$ as its constituents. We denote by S_{di} and S_{de} the ith and last tasks assigned to S_d, respectively, whereas the active task is denoted by S_{da}.

A. Orchestration Strategy

The orchestration strategy is responsible for prioritizing requests in Q and matching tasks to drones using certain criteria. We serve requests with the smallest Dur_r first as it reduces the average starting time of tasks. Then an orchestration procedure assigns tasks to drones by selecting drones that can start assigned tasks in the shortest time given drones current assignments. The assignment is completed by allocating drones available energy capacity to the assigned task. To make the allocation, the energy required to perform the task on the selected drone must include, in addition to E_r, the energy required to fly the drone to the task area or to transition from the preceding task of the drone, as well as the energy required to return to the station.

The above strategy is encapsulated in the procedure listed in algorithm 1. For each r in Q, the procedure iterates over available standby and inflight drones and calculates e_d, the energy required to perform r using d given its preceding tasks. If d has sufficient unassigned energy greater than e_d, then we calculate $t_d^{tostart}$, the duration until starting the task on d. Then, d is added to the candidate assignment set A_{cand}, where each candidate is expressed as a tuple $\langle d, e_d, t_d^{tostart} \rangle$. Once a number of candidates are collected, the candidate with the lowest $t_d^{tostart}$ is selected and assigned r. If no candidate is found, then r is rejected. The procedure continues to the next r in Q and repeats the steps above. Then, all rejected requests are put back in Q for subsequent calls of the procedure. The formulae for calculating e_d and $t_d^{tostart}$ are omitted for brevity.

B. Flight Duration and Energy Consumption

Once assignment decisions are made and drones configured, flight plans are created for standby drones and updated for inflight drones. Once a drone d is assigned a set of tasks S_d, the flying duration can be calculated as:

$$FlightDur = Tr(L_{S_{d1}}^{start} + L_{S_{de}}^{end}) + \sum_{i \in S_d} Tr(Transit_{i-1,i}) + Dur_i \quad (1)$$

Algorithm 1 Orchestration Procedure

Input: Q, D
Output: Assignments of tasks to drones

$R_{rejected} \leftarrow \phi$
Sort Q by $Dur_r \quad \forall r \in Q$ (Shortest Dur_r first)
while Q is not empty **do**
  r = dequeue(Q)
  $A_{cand} \leftarrow \phi$
  for all $d \in D$ **do**
    Calculate e_d and $t_d^{tostart}$
    if GetUnallocatedEnergy(d) $> e_d$ **then**
      $A_{cand} = A_{cand} \cup \{\langle d, e_d, t_d^{tostart} \rangle\}$
    end if
  end for
  if $A_{cand} \neq \phi$ **then**
    $\langle d, e_d, t_d^{tostart} \rangle \leftarrow$ Select from A_{cand} the tuple with minimum $t_d^{tostart}$
    Append r to S_d and update allocated energy
  else
    $R_{rejected} = R_{rejected} \cup \{r\}$
  end if
end while
Add all $r \in R_{rejected}$ back to Q

Note that $Transit_{0,1} = 0$. If the drone is already inflight, the remaining duration from the current time instant is:

$$RemDur = \text{Remaining Duration of } S_{da} + \left[\sum_{i \in S_d} x_i \times \left(Tr(Transit_{i-1,i}) + Dur_i\right)\right] + Tr(L_{S_{de}}^{end}) \quad (2)$$

where $x_i \in \{0, 1\}$ indicates if $i \in S_d$ is pending execution when equal to 1.

The energy consumption in Joules for a drone in forward flight for any duration T in seconds and speed v in m/s is:

$$E_{flight} = \textit{Energy}_f(v) \times T \quad (3)$$

where $\textit{Energy}_f$ is a function that computes the instantaneous power required for flight in watts given v. We adopted an energy model for rotary-wing drones that models the power consumption of drones in hovering ($v = 0$) and in forward flight. The model involves a host of parameters that include the drone mass, rotors and blade configuration and measurements, as well as air dynamics. The model is described in detail in [8]. We omit the energy required for ascending and descending as they are negligible and to simplify the evaluation.

V. Performance Evaluation

We evaluate the benefits of softwarized drones (referred to as dynamic NFV) compared to drones with limited and no softwarization. Drones with limited softwarization (hereafter called fixed NFV), are only reconfigurable when in standby and allows for assigning a single task per flight. On the other

hand, a non-softwarized drone is only capable of performing a single type of task (e.g, only aerial mapping), and is also assigned a single task per flight. The assignment strategies for these two models are simple assignments to the first available drone. For non-softwarized drones, a request task type K_r must match the assigned drone task type.

A discrete-event simulation environment is built using Python. The environment models the different drone states, energy consumption, task request generation and assignment according to our system model. It also records task starting and completion times on assigned drones according to tasks durations and transition times.

We investigate the benefits of softwarizing drones performing tasks as services. For scenarios where the batch of tasks is known beforehand, we report the total time to complete all tasks, which is the landing time of the last drone after performing all tasks. For use cases where task requests are not known a priori and thus arrive at random times, the total completion time is affected by inter-arrival times. Instead, we report the average task completion time, which is the delay from the arrival of a task to the end of its execution, averaged over the total number of tasks. The average task completion time can be calculated as:

$$\frac{1}{|R|} \times \sum_{r \in R} T_r^{completion} - T_r^{arrival} \tag{4}$$

where $T_r^{completion}$ is the recorded time of deactivating the task VNF in the drone after completing the task. We also calculate the total energy consumption for executing all tasks

A. Simulation Setup

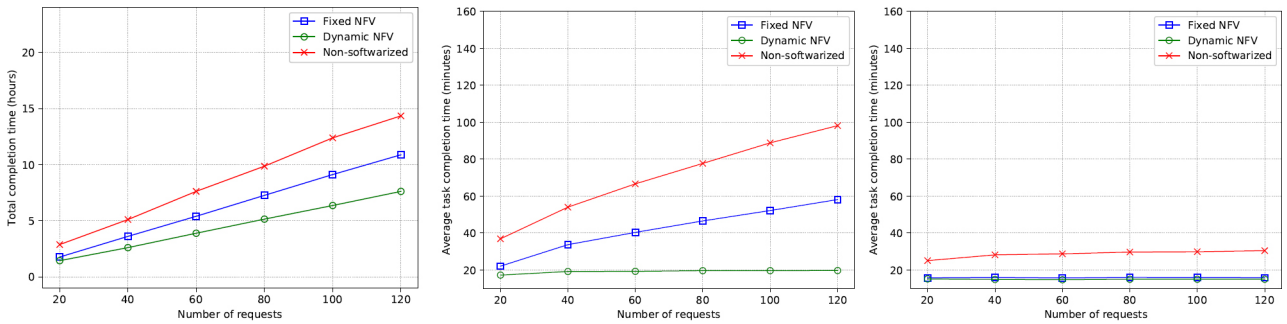
The simulation is setup as follows. Task durations are uniformly distributed in the range $[5, 20]$ minutes, while energy requirements are equal to the energy required for forward flight for the respective task durations. Request task types are also uniformly distributed over five task types. Assuming tasks take place in a 2×2 km^2 area and the depot located at $(0, 0)$, all distance in L and $Transit$ are drawn from two normal distributions, the first with $\mu = 1.6$ and $\sigma = 0.5$, and the other with $\mu = 1.2$ and $\sigma = 0.5$. For the scenario with random requests, inter-arrival times are exponentially distributed with with mean 5 and 10 minutes, denoted as $R_{arrival}$. In simulations with non-softwarized drones, drones are divided equally to task types. For example, for five drones, there is one non-softwarized drone available for each task type. Drone flying speeds are fixed at 10 m/s, while battery capacity is 702.58 kJ as the total of two batteries, resulting in about 45 minutes of maximum flight time after leaving out 10% of the capacity for safety. The parameters of the energy model used are as stated in [8], except drone mass is set to 3.5 kg. The charge time T_{charge} is set to 10 minutes. Simulations are terminated when all tasks are completed. All reported results are averages of 10 experiments.

B. Results

Figures 2a and 3a show the total completion time and energy consumption for the predefined requests scenario. The results are for five softwarized drones compared to five fixed NFV and and five non-softwarized drones. Tasks are completed faster using dynamic NFV compared to fixed NFV and non-softwarized drones. This is due the ability of reconfiguring and assigning multiple tasks to any available drone and due to the time saved traveling between tasks instead of returning to the depot after every task. Without softwarization, tasks take the longest time to complete due to having to wait for the availability of drones that match requested task types. However, with fixed NFV, the system waits for the availability of any drone to be reconfigured for the requested task type. As expected, the total completion time increases linearly with the number of requests, as does the difference in performance. The reduction in energy consumption, as shown in Figure 3a, is a direct result of the reduced travel times between tasks due to reconfigurability. Fixed NFV and non-softwarized drones are equal in energy consumption due to the identical operation in terms of performing a single task per flight.

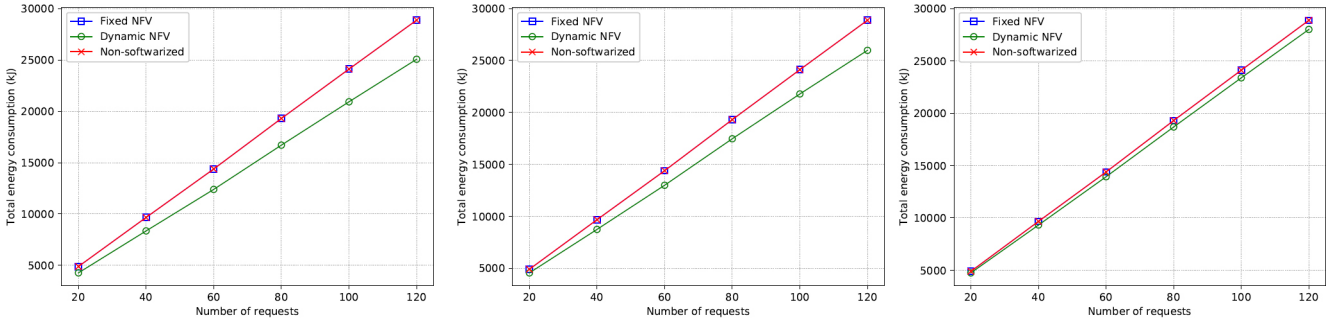
Next, we repeat the experiment above with task requests spread out with a 5 and 10 minutes mean inter-arrival time $R_{arrival}$. In Figures 2b and 2c, we report the average task completion time. In both cases, with dynamic NFV, the orchestrator capitalizes on the dynamic reconfiguration ability to assign tasks to drones inflight when possible resulting in the shortest per task completion time, which includes waiting time since request arrival. With $R_{arrival}$ = 5, the dynamic system deploys and completes tasks faster than the fixed NFV and non-softwarized systems. Using fixed NFV, tasks wait for drones to return and recharge before getting assigned, whereas with the non-softwarized system, tasks wait further for drones of the required task type, leading to higher completion times. The advantage is also evident with $R_{arrival} = 10$, which leads to equal performance for both dynamic and fixed NFV due to the lower utilization of the fleet. Both softwarized systems show an advantage over the non-softwarized system, albeit with a narrower difference. The corresponding energy consumption is shown in Figures 3b and 3c. The energy consumption difference decreases as $R_{arrival}$ increases. This is due to the fact that task requests are more spread out in time and drones become less utilized. This increases the chances of having available drones on standby to serve requests regardless of reconfigurability. As a result, many drone flights in dynamic NFV will constitute a single task per trip leading to similar energy consumption across all variants.

Figures 4a and 4b show the total completion time and the average task completion time for 120 predefined and random task requests (with $R_{arrival} = 5$) plotted against the number of drones. Tasks are completed faster using more drones. Naturally, an abundance of drones reduces the performance gains of softwarization. However, the notable result here is that softwarization enhances the performance of a limited number of drones to an extent equal to or better than a larger non-softwarized fleet. For example, 5 and 10 softwarized drones perform similar to 10 and 20 non-softwarized drones, respectively.



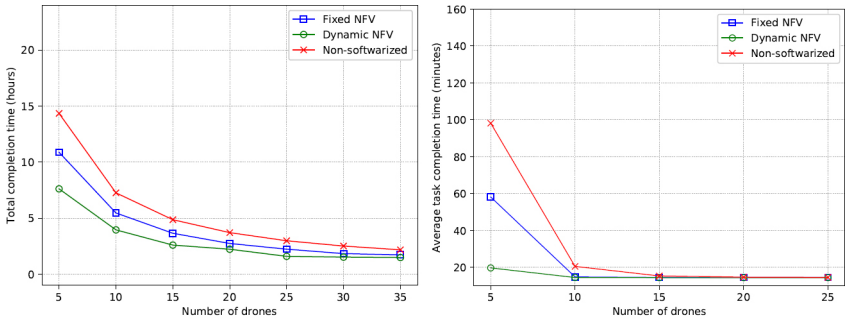
(a) Predefined tasks (b) Random tasks $R_{arrival} = 5$ (c) Random tasks $R_{arrival} = 10$

Fig. 2. Total and average task completion for predefined and random tasks scenarios with $|D| = 5$



(a) Predefined tasks (b) Random tasks $R_{arrival} = 5$ (c) Random tasks $R_{arrival} = 10$

Fig. 3. Total energy consumption for predefined and random tasks scenarios with $|D| = 5$



(a) Predefined tasks (b) Random tasks, $R_{arrival} = 5$

Fig. 4. Performance of the predefied and random tasks scenario with 120 requests

VI. Conclusion

In this paper, we discussed the motivations for drone softwarization to enable reconfigurable drones for providing a variety of sensing tasks. We envisioned a system where a fleet of drones performs civil and industrial tasks. A model for such a system was presented, where the proposed NFV-enabled drones can perform a multiplicity of dynamically configured tasks. We compared the proposed system performance to counterparts with limited and no softwarization. Our results show that a service provider can efficiently perform such a service and complete a predefined campaign of tasks in a shorter period of time compared to the alternatives. As well, a short turnaround time was achieved for random tasks.

The results presented herein represent a subset of results of an extensive investigation. Additional results include a scenario where higher priority task requests need to be deployed to respond to urgent events. For such a scenario, we employ an alternative orchestration strategy and then evaluate the ability of the softwarized system to respond to such events compared to a fixed system. Future work may consider sensing tasks that perform simultaneously when applicable, or more specialized use cases that seek objectives related to certain types of tasks. Such objectives may include maintaining coverage of a service or maintaining up-to-date sensing information by scheduling flights efficiently.

Acknowledgment

This research is supported by a grant from the Natural Sciences and Engineering Research Council of Canada (NSERC)

under grant number STPGP 521432-2018

The second author would like to acknowledge the generous support of Alfaisal University's Office of Research and Innovation.

REFERENCES

[1] H. Shakhatreh, A. H. Sawalmeh, A. Al-Fuqaha, Z. Dou, E. Almaita, I. Khalil, N. S. Othman, A. Khreishah, and M. Guizani, "Unmanned aerial vehicles (UAVs): A survey on civil applications and key research challenges," *IEEE Access*, vol. 7, pp. 48 572–48 634, 2019.

[2] O. Sami Oubbati, M. Atiquzzaman, T. Ahamed Ahanger, and A. Ibrahim, "Softwarization of uav networks: A survey of applications and future trends," *IEEE Access*, vol. 8, pp. 98 073–98 125, 2020.

[3] B. Nogales, V. Sanchez-Aguero, I. Vidal, and F. Valera, "Adaptable and automated small uav deployments via virtualization," *Sensors*, vol. 18, no. 12, p. 4116, 2018.

[4] C. Rametta and G. Schembra, "Designing a softwarized network deployed on a fleet of drones for rural zone monitoring," *Future Internet*, vol. 9, no. 1, p. 8, 2017.

[5] M. Alharthi, A.-E. M. Taha, and H. S. Hassanein, "Utilizing network function virtualization for drone-based networks," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–5.

[6] G. Faraci, C. Grasso, and G. Schembra, "Design of a 5g network slice extension with mec uavs managed with reinforcement learning," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 10, pp. 2356–2371, 2020.

[7] F. Busacca, L. Galluccio, and S. Palazzo, "Drone-assisted edge computing: a game-theoretical approach," in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2020, pp. 671–676.

[8] H. Sallouha, M. M. Azari, and S. Pollin, "Energy-constrained uav trajectory design for ground node localization," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–7.