CBDN: Cloud-Based Drone Navigation for Efficient Battery Charging in Drone Networks

Jinyong Kim, Student Member, Seokhwa Kim, Student Member, IEEE, Jaehoon Jeong, Member, IEEE, Hyoungshick Kim, Member, IEEE, Jung-Soo Park, Member, IEEE, and Taeho Kim, Member, IEEE

Abstract—For long-distance flying, drones often need to charge their battery at quick battery-charging machines (QCMs) because of their limited battery capacity. If a drone individually chooses a QCM without any coordination, a drone network may experience QCM congestion when multiple drones select the same QCM. This QCM congestion may lead to an increasing drone traffic delay. In order to solve this problem, we propose a cloud-based drone navigation (CBDN) system for efficient drone battery charging in drone networks. In order to achieve this goal, the CBDN gathers drone traffic information and determines efficient drone routes so as to minimize the overall QCM congestion level for drone battery charging using cloud-based management. Our key idea is to find globally coordinated drone routes so as to minimize the total traffic delay in a drone network by reducing the overall QCM congestion level. In order to demonstrate the effectiveness of the proposed system, we evaluated the performance of CBDN by simulating a drone network under various network conditions. The simulation results show that CBDN is more efficient than the existing shortest-path-based drone route planning algorithms in terms of end-to-end traffic delay and QCM average utilization.

Index Terms—Drone, battery charging, navigation, route, schedule, reservation.

I. INTRODUCTION

W
ting the growing popularity of Unmanned Aerial Vehicles (UAV), commonly known (and hereafter referred to) as drones, drones have seen widespread adoption in industry [1] and are being actively considered for military missions [2], [3]. This is because drones can perform tasks (e.g., delivery services, disaster-relief, and agricultural applications) that human beings cannot easily perform. Thus, major companies such as Google and Amazon are currently developing new drone technologies.

Lithium-ion batteries or lithium-ion polymer batteries are generally used for drones. Several alternative power sources (e.g., solar energy and hydrogen energy) have been developed, but rechargeable batteries continue to be the main power sources of drones [4]–[6]. For long-distance flying, however, drones often need to charge their batteries at Quick battery-Charging Machines (QCMs) because of their limited battery capacities. According to the type of drones and the battery used, the max service distance without battery recharging ranges from 3 km to 33 km [7]. For this reason, drone batteries should be periodically recharged at QCMs in order to support long-distance drone flying. Consequently, it will become important to avoid QCM congestion, which occurs when multiple drones simultaneously arrive at the same QCM; QCM congestion leads to an increasing traffic delay.

In order to solve this problem, we propose a Cloud-Based Drone Navigation (CBDN) system to efficiently determine drone routes so as to minimize the overall QCM congestion level for all of the QCMs in a target drone network. To the best of our knowledge, this paper is the first attempt for drone navigation services considering battery charging using cloud-based management. The proposed smart drone navigation system consists of QCMs, a Traffic Control Center (TCC) [8], and drones. The QCM is a station that charges used drone batteries so that they become fully charged. The TCC is the core computing and storage node in the drone cloud that collects trajectories of the drones, along with their average speeds, source positions, and destination positions, as well as their QCM statuses (i.e., battery-charging locations and schedule tables) for location management using cellular networks, such as the fourth-generation Long-Term Evolution (4G-LTE and 5G) [9], [10]. The TCC also maintains up-to-date traffic statistics of the drones and QCMs in the drone networks. In this environment, it is possible to realize a CBDN system for the cloud-based drone-traffic near optimization in order to minimize the overall QCM congestion level. In this paper, the QCM congestion level is defined as the waiting time of a drone for battery charging at a QCM due to the increased number of drones charging their batteries at that QCM.

The CBDN utilizes Dijkstra’s shortest path algorithm in consideration of the charging time and queuing time (i.e., waiting time) at a QCM, as well as the flight time on the end-to-end (E2E) path for the drone services. The CBDN allows drones to interact with the TCC to compute navigation paths in

Digital Object Identifier 10.1109/TITS.2018.2883058
terms of the drone-network-wide traffic optimization of all of the drones in the drone network. Before starting its service, each drone reports its source position, destination position, and average speed. Based on this drone-reported information, the TCC estimates the arrival, charging, and waiting time of each drone at a given QCM, as well as the E2E-path flight time and E2E-path travel time (i.e., the sum of the flight time, waiting time, and charging time). The TCC uses the estimated travel time to allocate the most suitable path to each drone, and also reserves the QCM-usage time of each drone so as to reduce the average travel time.

In addition, the CBDN can predict the near-future-congested QCMs along with the reported drone trajectories over time. Using this prediction information, whenever a new drone requests its navigation path from the TCC, the TCC sends the drone a drone-network-wide navigation path which does not select highly congested QCMs for the drone. Therefore, this TCC coordination can disperse the drone traffic throughout the drone network, leading to the minimum average E2E travel time. Consequently, the CBDN can provide coordinated drone-network-wide routes in order to reduce waiting times at QCMs for drones flying in a given target drone network. Note that this paper is an enhanced version of [13].

The main contributions of this paper are summarized as follows:

- **A drone-network architecture for drone battery charging**: The proposed architecture provides each drone in a drone network with an optimized path in terms of drone battery charging during the drone service (see Section III-A).

- **A globally optimized drone route finding algorithm**: The proposed optimization algorithm can minimize the average E2E travel time of drones for battery charging via QCM reservation. This strategy considers the efficiency of the drone-network-wide rather than the efficiency of each individual drone (see Sections III-C, IV, and V).

- **A simulation-based evaluation of the proposed system**: In order to show the effectiveness of the CBDN, we implemented a simulation of drone networks under various network conditions. The simulation results show that the proposed system outperforms the baseline schemes in terms of E2E-path travel time and battery-charging time (see Section VI).

The remainder of this paper is organized as follows. An explanation of existing drone related work is given in Section II. Section III explains the problem formulation of the efficient battery charging. Section IV describes the drone travel-time prediction. Section V presents the proposed drone battery-charging scheme along with the drone travel-time prediction. Section VI evaluates the performance of the CBDN system through comparisons with other baseline systems. Section VII concludes this paper and presents suggestions for future work.

II. RELATED WORK

The emergence of the new drone (i.e., UAV) industry is a recent phenomenon. Originally, drones were developed for military uses (e.g., reconnaissance, monitoring, and bombing). Recently, however, drone technology has rapidly come to be developed for various industrial purposes, because major enterprises such as Amazon and Google have introduced commercial drone applications like delivery, agriculture, infrastructure management, data sharing, outdoor/indoor navigation, and rescue services [14], [15]. Amazon unveiled a conceptual drone-based delivery system called “Prime Air” in December 2013 [16]. Prime Air comprises a drone delivery service, in contrast to a human delivery service. In addition, for this drone-based delivery system, Amazon is developing a charging infrastructure in order to extend the delivery ranges of its drones by providing them with charging stations [17], [18]. Google is working on the “Xing” project to build a delivery system using autonomous drones. Through this project, drones can be used to transport everything from groceries to emergency medicine in a faster, cheaper, and more environmentally friendly way [19], [20]. Such developments are not exclusive to major companies, as other businesses like Parrot [21], Nixie [22], Dji [23], and Airinov [24] have invested great efforts into drone-related research (e.g., software platform, battery, battery-charging machine, communications, and privacy security).

As the drone industry continues to progress, various drone energy sources (e.g., battery, solar energy, and hydrogen energy) are being developed for the provision of smooth drone services. Among the various drone energy sources, batteries are the main source because of their stability; however, the major disadvantage of batteries is their running times, which are typically inadequate for the execution of various drone services [1]. Due to this running-time issue, many enterprises are currently developing autonomous charging systems and battery-charging machines for drones to charge their batteries at battery-charging machines during the execution of long-distance drone services [5], [17], [25]–[28]. However, while charging systems and battery-charging machines are being actively developed, drone navigation paths requiring multiple battery charge do not currently consider congestion at battery-charging machines. For example, Amazon drones consider only the shortest distance from a source to a destination, without considering possible congestion at the charging stations [17], [18]. In the case of Google, their drones only consider the furthest distance they can deliver without battery charging [20]. Another paper focusing on drone charging systems considered power consumption for drones as affected by the carried baggage and wind-blown environment in order to increase the capacity of the battery [29]. However, the systems of Amazon, Google, and the other paper described above do not consider the congestion that may occur at battery charging stations. Note that, to the best of our knowledge, this paper is the first attempt for drone navigation services considering battery charging and drone congestion at battery charging stations using cloud-based management.

Drone software platforms are also being developed for the drone utilization regarding a variety of services [30]. These software platforms were initially only developed for flight control, but recent developments have led to other services being researched [14], [15].
Research on drone communication and control has also been performed [31], where drones can communicate with each other using satellites and wireless communication forms (e.g., WiFi and 4G-LTE). For efficient drone communication, Qualcomm, best known for inventing mobile technologies, is developing a cellular drone communication system for safe and autonomous drone navigation. Qualcomm is accelerating the development of drone-communication technology by conducting drone-operation testing over commercial LTE networks as well as optimizing present LTE networks for safe and low-altitude drone operations [32]. Furthermore, many research activities on security and privacy are currently being performed in order to prevent malicious users from illegally accessing and controlling drones [33], [34].

III. PROBLEM FORMULATION

This section presents our goal, drone network architecture, assumptions, and the basic concept of the proposed drone navigation system. Given the drone trajectories in a drone network, our goal is to provide efficient drone routes for the drones in order to minimize the average E2E-path total travel time of all of the drones rather than the individual E2E-path total travel time of each drone. In other words, we aim at providing the drones with efficient navigation paths including battery charging stations, while considering the paths of all the drones from the viewpoint of a drone network rather than from the viewpoint of per-drone optimization.

A. Drone Network Architecture

In this section, the proposed drone-network architecture for the drone networks is described. Fig. 1 shows an example drone network with QCMs. As shown in Fig. 1, it shows the enlarged drone network in the main area and shows the possibility of this network expanding to other districts as well. Note that the drones around the QCM are the drones that are waiting to charge their batteries. The proposed drone network architecture consists of the following components:

- **Traffic Control Center (TCC)**: As the core computing and storage node of the drone cloud, the TCC is a cloud-based management system that maintains the drone and QCM statuses for location management using cellular links, such as the 4G-LTE and 5G networks [9], [10]. The TCC can provide up-to-date traffic statistics of the drones and QCMs, such as the average speed, and the drone-speed deviation along with the source, destination, current position, and moving direction.

- **Quick battery Charging Machine (QCM)**: A QCM station charges used drone batteries. The QCMs report their latest queuing information to TCC using cellular links, such as the 4G-LTE and 5G networks [9], [10]. It is assumed that the QCMs are randomly deployed throughout the drone network; QCM placement is left for a future study.

- **Drones**: The drone is a UAV flying from a source position to a destination position. It is assumed that the drones know the waiting delay for the battery charging at each QCM through communicating with the TCC via cellular links, such as the 4G-LTE and 5G networks [9], [10]. In order to decrease battery consumption from frequent communication for navigation, except for in the case of an emergency (e.g., the failure or power outage of a QCM in the drone’s route), the drones communicate with the TCC only once when starting their service.

B. Assumptions

We have the following assumptions regarding the design of an efficient path-finding algorithm including drone battery charging in the drone networks:
The TCC, the QCMs, and the drones use the cellular links, such as the 4G-LTE and 5G networks [9], [10], because the cellular links can be used for secure and long-distance communication through cloud-based security service systems [11], compared to other communication technologies such as Bluetooth and Wi-Fi [12].

In order to decrease battery consumption, the drones only once report their sources, destinations, current positions, and mobility information (i.e., direction and speed) to the TCC in order to acquire their path.

The QCMs periodically report their statuses and service statistics (e.g., reservation information and waiting queue) to the TCC.

A QCM can only charge one used drone battery at a time. In addition, this charging scheme can be easily extended to a multiple-charging scheme through equipping a QCM with multiple chargers and multiple QCMs with a single charger.

The drone can arrive at both the QCM and the destination at the estimated arrival times, like an airplane; this capability is because the drone can control its own speed in the drone network so as to arrive at the estimated arrival times, and this is even the case under some resistance, such as air resistance from wind.

Based on the characteristics of vehicle networks in previous studies (e.g., [35] and [36]), we assume that the travel time of the drones follows a Gamma distribution.

The battery capacities, weights, and motor abilities of all of the drones are the same, but the weights of the carried baggage are not the same.

The battery consumption of each drone is assumed to be dependent only on weight of the carried baggage.

No drones are discarded or replaced in the QCM.

There are already authentication mechanisms between the drone and the QCM, so that only authorized drones can be serviced.

C. Concept of Cloud-Based Drone Navigation

This section explains the concept of the proposed CBDN that uses a charging reservation system at each QCM. The aim of the CBDN is to reduce the travel time (i.e., the sum of the waiting time and the charging time at the QCM, and the flight time along the E2E path) through the TCC-coordinated provision of the drone with a drone-network-wide navigation path in the drone network, rather than through the provision of a per-drone navigation path for each individual drone. The TCC identifies near-future-bottleneck QCMs using the drone trajectories, then provides each drone with an efficient navigation path in consideration of the QCM aggregated congestion and with respect to the drone battery charging.

The CBDN system considers the navigation paths of all of the drones in a network rather than the navigation path of only an individual drone. Fig. 2(a) shows a drone network for which the drone-network-wide navigation path of the navigating drone is considered. Suppose that the drones $d_1$, $d_2$, $d_3$, $d_4$, and $d_5$ sequentially began to move for the simultaneous provision of drone services along their trajectories, but then the $d_5$, $d_4$, $d_3$, $d_2$, and $d_1$ drones sequentially arrived at the $q_1$. At first, when it began, $d_1$ received the shortest navigation path including the $q_1$ from the TCC. The path travel time which is the sum of the flight time, the waiting time, and the charging time, is the shortest for $d_1$ here. Next, when the $d_2$ started to move, it received the shortest navigation path including the $q_1$ from the TCC. Likewise, when the $d_3$, $d_4$, and $d_5$ started to move, the drones respectively received the shortest navigation paths including the $q_1$ from the TCC. Accordingly, although the drones received the shortest paths from the TCC, except for the $d_5$, the drones may not have received the shortest navigation paths in reality, because these paths may cause congestion at the $q_1$ within a very short time. This potential congestion is caused by each drone only considering its own navigation path without considering the navigation paths of the other drones. In actuality, when each drone starts, the estimated travel time of each drone is expected to be the shortest travel time, because the waiting time of each drone at the $q_1$ is 0.

However, due to the unexpected waiting time at the $q_1$, the drone travel times will be longer than expected, and this is because the late-start drones arrive at the $q_1$ first. As a result, the average travel time in the drone network may increase with individual-drone traffic optimization, which does not consider the drone-network-wide traffic optimization of all of the drones.
In order to solve this flight-optimization problem, we propose our CBDN system. The main contributions of the CBDN include dispersing the drone traffic throughout the drone network, as well as minimizing the average travel time using the drone-network-wide traffic optimization approach. For this drone-network-wide optimization, the CBDN uses the aggregated waiting time (i.e., global waiting time) of all of the drones at each QCM; the global waiting time means the whole aggregated waiting time of not only the starting drone but also of all other drones at the same QCM. Fig. 2(b) shows a drone network where all of the drone navigation paths are considered for the drone-network-wide traffic optimization in selecting the QCMs for the battery charging. As shown in Fig. 2(b), the $d_1$, $d_2$, $d_3$, $d_4$, and $d_5$ are attempting to charge their batteries at the $q_1$, $q_2$, and $q_3$. In the drone-network-wide traffic optimization, the $d_2$, $d_3$, and $d_4$ may experience a slightly increased E2E-path flight time due to the detours in their flight paths for the purpose of avoiding QCM congestion. The average E2E travel time in the drone network, however, will decrease, since these $d_2$, $d_3$, and $d_4$ detours help reduce the drone traffic at the $q_1$. As a result, the average travel time of the drone-network-wide navigation paths will be shorter than the average travel time of the per-drone navigation paths. Therefore, the CBDN can achieve drone-network-wide traffic optimization in the drone network. A detailed explanation of the global waiting time is provided in Section IV-D.

**D. NP-Hardness of Battery Charging Scheduling**

We prove that the battery charging scheduling for efficient drone navigation is an NP-hard problem. In order to prove this, we consider a special case of the drone navigation problem (see Fig. 3). In this case, each drone has a different charging time because the destinations of all drones are generally different from each other, as shown in this figure. Here, we assume that the charging time of a drone is determined by the travel distance from its source to its destination. Under this case and assumption, we show that our drone navigation problem is NP-hard by using a straightforward reduction from a job-shop scheduling problem [37], which is a well-known NP-hard problem. Given two finite sets of machines and jobs in the job-shop scheduling problem, we can construct the set of QCMs and the set of charging times required for the drones where the machines are the QCMs and the job processing times are the charging times. That is, we can see that the job-shop scheduling problem is a special case of our drone navigation problem.

Since the drone navigation problem is NP-hard, we focus on developing a practically working heuristic algorithm to efficiently solve this problem instead of using an optimal algorithm to find a best answer. Thus, for a given drone network, our heuristic algorithm aims at a drone-network-wide optimization rather than an individual drone optimization for overall good performance.

**IV. Travel-Time Prediction**

This section contains the modeling of the travel-time (i.e., sum of the flight time, global waiting time, and charging time) prediction for the identification of the drone-network-wide E2E path. Upon commencement of the drone service, each drone reports its own mobility information (e.g., average speed, direction, source position, and destination position) to the TCC. With this information, the TCC can calculate the flight time based on the link, as well as the waiting and charging times at the QCMs along the E2E path. Thus, it is possible to acquire the most efficient navigation path using the calculated travel time.

**A. Battery Capacity for Flight Time of Individual Drones**

This section explains how to calculate the flight time that an individual drone can fly based on the remaining battery capacity. Note that this flight time is based on several factors such as the LiPo (lithium polymer) battery capacity, the wind speed, flight load, average amperage draw (e.g., drain), weather conditions, and so on [38]. According to the result [38], in order to calculate the flight time of individual drones, three factors are used. The three factors for the flight time calculation are as follows:

- **Battery Capacity**: It is known that the bigger the battery capacity is, the longer the flight time of a drone is. In this paper, we assume that the battery capacity of all the drones is the same.
- **Battery Discharge**: For the flight time calculation, the battery discharge is an important factor. In general, in the case of a drone that uses a lithium-ion battery or lithium-ion polymer battery, if the battery’s voltage level is less than 20%, it has same possibility of permanent damage [39]. In addition, for an emergency (e.g., a failure or power outage of a QCM in its own route), the considering the total capacity of the battery. For these reasons, in our paper, we calculate the available battery capacity for the flight time with only 80% of the actual battery capacity.
- **Average Amperage Draw**: In order to calculate the average amperage (amp) draw, we need to know the sum of the weights of the drone and the carried baggage, as well as the motor ability of the drone. For example, if the motor ability of the drone draws 1 amp in order to
produce 100g of thrust, and the weight of the sum of the drone and the carried parcel is 2000g, the average amp draw of the drone is 20 amps. In this paper, we assume that the weights and motor abilities of all of the drones are the same, but that the weights of the carried baggage are not the same. Therefore, note that the average amp draw depends on the weight of the carried baggage.

Using these three factors, the flight times of individual drones can be computed as follows:

\[
\text{Flight Time} = \text{Capacity} \times \frac{\text{Discharge}}{\text{Average Amp Draw}} \times 60.
\]

B. Flight Time Prediction Considering Charging Its Battery on End-to-End Path

This section explains the modeling of the flight time in terms of both the link between two QCMs and the source-destination E2E flight path. Note that this flight-delay modeling is based on [35] and [36].

1) Flight-Delay on the Link Between Two QCMs: Many studies on flight have demonstrated that the flight delay of a moving vehicle, airplane, or UAV over a fixed distance follows a Gamma distribution. Therefore, the flight delay through a node-to-node link \(i\) in the drone network (called link flight delay \(f_i\)) is modeled as: \(f_i \sim \Gamma(\kappa_i, \theta_i)\) where \(\kappa_i\) is a shape parameter and \(\theta_i\) is a scale parameter.

The means of those values can be used to compute the parameters \(\kappa_i\) and \(\theta_i\) by the mean \(\mu_i\) and the variance \(\sigma_i^2\) of the link flight delay \(f_i\). In order to compute \(\kappa_i\) and \(\theta_i\), we define the mean of \(f_i\) \(E[f_i] = \mu_i\) and the variance of \(f_i\) \(Var[f_i] = \sigma_i^2\). Thus, \(\kappa_i\) and \(\theta_i\) can be computed as follows:

\[
\theta_i = \frac{Var[f_i]}{E[f_i]} = \frac{\sigma_i^2}{\mu_i}, \quad (2)
\]

\[
\kappa_i = \frac{E[f_i]}{\theta_i} = \frac{\mu_i^2}{\sigma_i^2}. \quad (3)
\]

2) Flight Time on End-to-End Path: This section explains the delay model of an End-to-End (E2E) flight path (i.e., drone trajectory) from one position to another in a given drone network. As mentioned above, the link flight delay is modeled as the Gamma distribution of \(f_i \sim \Gamma(\kappa_i, \theta_i)\) for a node-to-node link \(i\). Note that as shown in Fig. 4, two contiguous edges (e.g., \(f_1\) and \(f_2\)) are connected via a common vertex (e.g., \(QCM_1\)) corresponding to a QCM, such that there is some QCM waiting delay at the common vertex. For the sake of simplicity, we assume that the QCM waiting delay \(a_{ij}\) at QCM \(f_j\) for an edge \(f_{ij}\) (denoted as \(f_i\)) is included in link flight delay \(f_i\).

Given a specific flight path (i.e., drone trajectory), we assume that the link flight delays of different node-to-node links on that path are independent. Under this assumption, we approximate the mean (or variance) of the E2E flight delay as the sum of the means (or variances) of the link flight delays for the links along the E2E path. In the case where the drone trajectory consists of \(QCM_{n-1}\) node-to-node links, as shown in Fig. 4, the mean and variance of the E2E flight delay \(F\) are computed based on the link flight delay independence as follows:

\[
E[F] = \sum_{i=1}^{n-1} E[f_i] = \sum_{i=1}^{n-1} \mu_i. \quad (4)
\]

\[
Var[F] = \sum_{i=1}^{n-1} Var[f_i] = \sum_{i=1}^{n-1} \sigma_i^2. \quad (5)
\]

From (4) and (5), we model the E2E flight delay \(F\) as a Gamma distribution as follows: \(F \sim \Gamma(\kappa_F, \theta_F)\), where \(\kappa_F\) and \(\theta_F\) are computed by \(E[F]\) and \(Var[F]\) using the formulas of (2) and (3). It is noted that our flight delay prediction can accommodate any superior E2E path delay estimation if one is available from either another mathematical model (considering traffic congestion at a QCM) or empirical measurement (e.g., drone’s flight experience in real time). Therefore, we can compute a drone’s E2E flight delay from source to destination for a given drone trajectory.

C. Charging Time at QCMs

This section presents the requisite charging time at a QCM for fully charging a used battery. The charging time differs according to the amount of battery used. Let the QCM charging time, the full-battery charging time, the full-battery amount, and the used-battery amount be \(C_{qcm}\), \(C_{full}\), \(a_{full}\), and \(a_{used}\), respectively. The the formula for the drone-charging time at the QCM is as follows:

\[
C_{qcm} = C_{full} \times \frac{a_{used}}{a_{full}}. \quad (6)
\]

D. Global Waiting Time at QCMs

This section describes the global waiting time at QCMs. The global waiting time considers not only the individual waiting
time of the starting drone, but also the additionally generated waiting times of the other drones with the reservation schedules at the QCM; this is because the starting drone may have to wait for the drones currently at the QCM to finish charging. In order to estimate the global waiting time, up-to-date traffic-statistic information of the drones and the QCMs were collected in the drone network under the management of the TCC. Using this updated statistical information, it is then possible to calculate the estimated global waiting time of the starting drone at the QCM. Fig. 5 shows the schedule chart of the battery charging at the QCM; here, the zone \( Z \) is defined, which is the group of consecutive charging times, along with the gap \( G \), which is the inter-zone empty space. As shown in Fig. 5, the global waiting time should be calculated differently according to the arrival points of the starting drone at the QCM. For this global waiting time, the calculations of the individual waiting time and the additional waiting time were performed according to the following three cases of the starting-drone arrival points: (i) the case where the starting drone arrives at the QCM than the reserved charging time of the first drone in the zone, (ii) the case where the starting drone arrives at the QCM later than the reserved charging time of the last drone in the zone, and (iii) the case where the starting drone arrives at the QCM between the reserved charging times of the first drone and the last drone in the zone.

1) Individual Waiting Time: The description of the individual waiting time of the starting drone at the QCM is provided here. Fig. 6 shows the schedule chart for each case where the starting drone reserves the charging time at the QCM. As shown in Fig. 6(a), in the first case, the starting drone can immediately recharge its own battery at the QCM; here, the individual waiting time of the starting drone is zero, and therefore the starting drone can recharge its battery upon arriving at the QCM without waiting. Alternatively, in the other cases, the starting drone needs to wait for the overlapped charging time that has been reserved by the other drones, as shown in Figs. 6(b) and 6(c). In these cases, the individual waiting time of the starting drone is the overlapped time. The individual waiting time, \( W_i \), can be calculated as the difference between the charging end time of the drone being charged ahead of the starting drone and the arrival time of the starting drone. Therefore, the starting drone can charge its battery after the battery of the currently-charging drone is fully charged.

2) Additional Waiting Time: The description of the additionally generated waiting time at the QCM is given here. For the global waiting time, we consider not only the individual waiting time of the starting drone, but also the additionally generated waiting time that is reserved by the other drones. This consideration is because the charging schedule of the starting drone may conflict with the charging schedules of the other drones being charged due to the overlapped charging time, as shown in Figs. 6(a), 6(b), and 6(c). Therefore, the other drones need to additionally wait for the overlapped time. The additional waiting times of the other drones can be calculated as the difference between the charging end time of the starting drone and the charging start time of the first drone in the first zone. The additional waiting time in each zone will continually occur until the difference is less than zero.

With the calculated individual and additional waiting times, the global waiting time of the starting drone can be calculated as the sum of the additional waiting times of the other drones in each zone and the individual waiting time of the starting drone. The formula for the estimated global waiting time (denoted as \( W_g \)) is as follows:

\[
W_g = W_i + \sum_{j=1}^{M} \left[ Z_j (W_o - \sum_{i=1}^{j-1} G_i) \right],
\]

where \( Z_j \) and \( M \) are defined as the number of drones affected by the new drone in the \( j \)th zone and the index of the last zone among the zones affected by the charging of the starting drone, respectively.

3) An Example of Global Waiting Time: This section explains how to calculate the global waiting time with (7) and an example. Fig. 7 shows a schedule chart including zones, gaps, and schedule time. With the schedule chart in Fig. 7,
we can acquire the $i$th gap $G_i$, the number of the drones affected by the charging of the new drone in $j$th zone $Z_j$, the index of the last zone among the zones affected by the charging of the new drone $M$, the individual waiting time of the new drone $W_i$, and the overlapped time $W_o$. $G_1$, $G_2$, and $G_3$ are 30, 40, and 90, respectively. $M$ is 3 because the overlapped time affects the first zone, second zone, and third zone. $Z_1$, $Z_2$, and $Z_3$ are 3, 2, and 2, respectively. $W_i$ is 50 (i.e., difference between the charging end-time of the first drone in the first zone and the arrival time of the new drone) and $W_o$ is 75 (i.e., the overlapped time by the charging time of the new drone). Therefore, using these values, we can calculate the global waiting time (i.e., $50 + (3 \times 75 + 2 \times (75 - 30) + 2 \times (75 - (30 + 40))) = 375$).

In the next section, based on the travel-time prediction, an explanation of the CBDN design is provided.

V. CLOUD-BASED DRONE NAVIGATION DESIGN

In this section, an explanation of the CBDN design for the efficiency of the drone battery charging is provided. The purpose of the CBDN is to minimize the average travel time of the drones through dispersing the drone traffic throughout the drone network, and by assigning each drone to an appropriate QCM for battery charging. In this section, the following processes are explained: (i) The CBDN design and algorithm, (ii) CBDN system comparison with the other baseline systems, and (iii) CBDN optimization.

A. CBDN Design and Algorithm

Fig. 8 shows four trajectories according to various algorithms. As shown in this figure, a variety of drone source and destination trajectories are possible, and it is important to select the most efficient trajectory for long-distance drone services. If the waiting time and charging time at the QCM are not considered, the shortest-distance path (or shortest-flight-time path) can be found using Dijkstra’s algorithm. However, because of the short drone running time caused by the battery capacity, both the waiting time and charging time at QCMs should be considered.

In order to solve this problem, we propose an enhanced shortest-path algorithm that considers the waiting and charging times at QCMs as well as the E2E-path flight time. The core of this algorithm is using the TCC-reservation concept in order to acquire the most efficient shortest path in terms of the drone-network-wide traffic optimization of all of the drones in the drone network. For the drone-network-wide traffic optimization, a drone sends its own position information to the TCC prior to starting its flight. Using this position information, the TCC can construct a reachability graph based on the shortest-flight distances among the QCMs and the destination with the fully charged battery.

The drone reachability graph is shown in Fig. 9. Let the service time be the sum of the waiting and charging times. With this drone reachability graph, the TCC calculates the estimated travel time using the estimated service and flight times along the E2E path. With this calculated travel time, the TCC selects the most efficient navigation path from the source to the destination for each drone. A detailed explanation of how the CBDN algorithm finds the appropriate QCMs is as follows:

Regarding Algorithm 1, the most appropriate path $P_{u,v}$ is returned for the input of the starting drone $D_{cur}$, the capacity of the battery of the starting drone $C_{battery}$, the weight of the carried baggage $B_w$, the set of all the QCMs $Q_{all}$, the source position $u$, and the destination position $v$. 
Algorithm 1 CBDN Algorithm

1: function GET_PATH(Dcur, Cbattery, Bw, Qall, u, v)
2:     T,u,v ← ∞
3:     if Distanceu,v < Fcapa * Savg then ▷ check if the starting drone can arrive at destination v from source u without battery charging or not. The flight time Fcapa for the battery capacity that the drone Dcur can fly is dependent on the weight of the carried baggage Bw.
4:         Pu,v ← Pdirect ▷ Pdirect means a direct path without the battery charging from u to v.
5:     return Pu,v ▷ return a direct path from u to v.
6:   else
7:       Qreachable ← Get_Reachable_QCM(Dcur, Cbattery, Bw, Qall) ▷ get a set of reachable QCMs Qreachable which the drone Dcur can arrive at QCMs with its own battery.
8:       Gau,v ← Get_Graph(Dcur, Qreachable, u, v) ▷ get a reachability graph Gau,v.
9:       P1,k ← Get_K_Shortest_Paths(Gau,v, k) ▷ get k shortest paths considering the E2E path flight time and charging time at QCMs in Gau,v.
10:   for i ← 1, k do
11:       Ti ← Compute_Travel_Time_of_Path(Pi) ▷ compute the estimated travel time (i.e., the sum of the flight time, the charging time, and the global waiting time) for the i'th path.
12:       if Ti < T,u,v then ▷ check if the estimated travel time for the i'th path is short or not.
13:           Tu,v ← Ti
14:           Pu,v ← Pi ▷ change the appropriate path.
15:     end if
16:   end for
17:   end if
18: return Pu,v ▷ return the most appropriate path from u to v.

end function

In lines 3–5, the TCC calculates whether or not the drone is operated within the distance Distanceu,v from which it can arrive at its destination v from source u without the need for battery charging. This distance is the product of the average speed Savg of the drone and the flight time for the remaining battery capacity Fcapa calculated from (1). The flight time for the remaining battery capacity that the drone Dcur can fly is dependent on the weight of the carried baggage Bw. If the drone can reach its destination without battery charging, the TCC assigns the drone with a direct path Pdirect from the source to the destination. If the drone cannot reach its destination without the battery charging, the TCC calculates the QCMs that are reachable from the source to the destination considering the flight time for the remaining battery capacity that the drone Dcur can fly in line 7. In line 9, with these reachable QCMs, the TCC obtains the reachability graph Gau,v (i.e., the available paths). The Get_K_Shortest_Paths() gets k shortest paths in terms of the sum of the E2E path flight time and charging time at QCMs in Gau,v in line 10. The paths increase the minimum flight time in the flight-time matrix M for the target drone network; they are computed using Yen’s k-shortest-path algorithm [44] along with matrix M, and are then stored into the set k. In lines 11–18, the TCC computes the estimated travel time Ti for the i'th path in consideration of the estimated E2E-path flight time, the charging time, and the global waiting time, and then selects the path with the shortest travel time. The estimated travel time Ti is calculated as follows:

\[ T_i = F_i + C_i + W_i, \]

where

- \( G_{u,v} \): the reachability graphs from the source u to the destination v including the QCMs,
- \( P_i \): the i'th shortest path in terms of the sum of the E2E path flight time and charging time at QCMs among the \( G_{u,v} \),
- \( T_i \): the estimated travel time for the \( P_i \),
- \( F_i \): the estimated E2E path flight time for the \( P_i \),
- \( C_i \): the estimated sum of charging times at the QCMs for the \( P_i \), and
- \( W_i \): the estimated sum of global waiting times at the QCMs for the \( P_i \).

With this calculated travel time, the TCC assigns the corresponding flight path to the drone Dcur in line 19.

Note that the time complexity of Algorithm 1 is \( O(kN(M + N\log N)) \), where \( N = |V|, M = |E| \), and k is the number of paths in the k-shortest-path algorithm called “Get_K_Shortest_Paths()” in line 10 [44]; this is because the k-shortest-path algorithm is the dominant function in Algorithm 1, and its time complexity is \( O(kN(M + N\log N)) \) [44].

B. CBDN System Comparison With the Other Baseline Systems

In this section, an explanation of the drone-navigation using the CBDN system is given through comparing it with the other baseline systems (i.e., Dijkstra, statistic, and individual reservation navigation); the CBDN is shown to be the most efficient system for drone services.

Fig. 10 shows the travel times as the sum of the flight time, the waiting time, and the charging time. In addition, the subfigures show each case of the travel time in consideration of the individual waiting time and the global waiting time. As shown in Figs. 10(a) and 10(b), the starting drone can acquire the four trajectories using the Shortest-Flight-Wait-Time (SFWT), Shortest-Flight-Time (SFT), Individual Reservation Navigation (IRN), and Cloud-Based Drone Navigation (CBDN) systems, respectively. These systems can be defined as follows:

- **Shortest-Flight-Wait-Time System (SFWT):** The SFWT system considers the flight time on an E2E-path and the statistical waiting time and charging time at QCMs. In order to acquire the statistical waiting time, each QCM calculates the average waiting time with the drone waiting times and reports it to the TCC. With this reported average waiting time, the TCC calculates
the estimated travel time of the starting drone using Dijkstra’s algorithm. Then, the TCC selects the most efficient path with the shortest travel time and assigns this path to the drone.

- **Shortest-Flight-Time System (SFT):** The SFT system considers only the flight time on an E2E-path and the charging time without the waiting time at QCMs; that is, the TCC calculates the estimated travel time of the drone with only the flight and charging times using Dijkstra’s algorithm. Then, the TCC selects the most efficient path with the shortest flight time and assigns this path to the drone.

- **Individual Reservation Navigation System (IRN):** The IRN system considers the flight time on an E2E-path and only the individual waiting time of the starting drone and the charging time at QCMs. This system uses the reservation schedule at each QCM. With these reservation schedules, the TCC calculates the estimated individual travel time of only the starting drone for each individually-possible shortest travel path with the individual waiting time at each QCM along the path. Then, the TCC selects the most efficient path with the shortest travel time and assigns this path to the drone.

- **Cloud-Based Drone Navigation System (CBDN):** The proposed CBDN system, which considers the global waiting time, considers the flight time on an E2E-path as well as the waiting time and charging time of all of the drones at QCMs. This system is similar to the IRN system in that it considers the individual waiting time. However, unlike the IRN system, the CBDN system calculates the global waiting time in consideration of the additionally generated waiting times of the already reserved drones, because the starting drone may interfere with the charging of the other drones at the QCMs. With this calculated global waiting time, the TCC computes the estimated travel time of the starting drone. Then, the TCC selects the most efficient path with the shortest travel time and assigns this path to the drone. A detailed explanation of the global waiting time is given in IV-D.

Fig. 10(a) shows the travel times of the four trajectories in consideration of the drone individual waiting time. In this figure, the travel time of the path for which the IRN system was used is shorter than those of the other possible paths. From the viewpoint of the average travel time of all of the drones, however, this may not be the shortest average travel time, because the navigating drone may interfere with the charging of the already reserved drones at the QCMs.

Fig. 10(b) shows the travel times of the four trajectories in consideration of the drone global waiting time. In this figure, the travel times of all of the trajectories, except for the trajectory using the CBDN system, are shown to be increasing, because the other trajectories do not consider the waiting time of the already reserved drones at the QCMs, and instead consider only the individual waiting time of the starting drone. Therefore, from the viewpoint of the average travel time of all of the drones, the average travel time of the CBDN system is the shortest, although the starting drone can experience detours to a certain degree.

**C. Considerations in CBDN System**

This section describes the considerations in our CBDN system. For effectiveness of the CBDN system, we examine two considerations as follows: (i) Battery charging level at the QCMs and (ii) Failure or power outage of a QCM.

1) **Battery Charging Level at the QCMs:** In this section, we explain a battery charging strategy for efficient navigation. For battery charging, we consider two charging methods as follows: (i) the Maximum-Charging-Level Method and (ii) the Minimum-Charging-Level Method. In the maximum-charging-level method, the used battery of each drone is fully charged at all QCMs, except for the last QCM; in the case of the last QCM, the battery of the drone is charged considering the distance to the destination. In contrast, in the minimum-charging-level method, the used battery of each drone is charged to a level with which it can reach the next-hop QCM or
Algorithm 2: CBDN Algorithm for Emergency

function GET_ALTERNATIVE_PATH(D_{car}, C_{battery}, B_{w}, Q_{all}, u, v, Q_{failure})
1. \( T_{u,v} \leftarrow \infty \)
2. \( Q_{reachable} \leftarrow \text{Get}_\text{New}_\text{Reachable}_\text{QCM}(D_{car}, C_{battery}, B_{w}, Q_{all}, Q_{failure}) \) \( \triangleright \) get a new set of reachable QCMs \( Q_{reachable} \) except for the failure QCM \( Q_{failure} \) which the drone \( D_{car} \) can arrive at QCMs with the remaining battery \( B_{w} \).
3. \( G_{u,v} \leftarrow \text{Get}_\text{Graph}(D_{car}, Q_{reachable}, u, v) \triangleright \) get a reachability graph \( G_{u,v} \), \( u \) and \( v \) mean current location and destination.
4. \( P_{1,k} \leftarrow \text{Get}_\text{K}_\text{Shortest}_\text{Paths}(G_{u,v}, k) \triangleright \) get \( k \) shortest paths considering the E2E path flight time and charging time at QCMs in \( G_{u,v} \).
5. for \( i \leftarrow 1, k \) do
6. \( T_i \leftarrow \text{Compute}_\text{Travel}_\text{Time}_\text{of}_\text{Path}(P_i) \triangleright \) compute the estimated travel time (i.e., the sum of the flight time, charging time, and the global waiting time) for the \( i \)th path.
7. if \( T_i < T_{u,v} \) then
8. \( T_{u,v} \leftarrow T_i \) \( \triangleright \) check if the estimated travel time for the \( i \)th path is short or not.
9. \( P_{u,v} \leftarrow P_i \) \( \triangleright \) change the appropriate path.
10. end if
11. end for
12. return \( P_{u,v} \) \( \triangleright \) return the most appropriate path from \( u \) to \( v \).
13. end function

its destination. Although these two cases seem to have no big difference because the sums of the charging times at the QCMs are the same, there is a significant difference in our CBDN system. This is because the CBDN system employing the minimum-charging-level method makes drones have more opportunities to choose less busy QCMs for their paths. In other words, the CBDN system with the maximum-charging-level method may let the charging time of each drone be longer at the other QCMs (except for the last QCM) than that with the minimum-charging-level method. Therefore, when other new drones start a new service, the CBDN system with the minimum-charging-level method has a higher probability of providing the drones with better paths than the CBDN system with the maximum-charging-level method, because the charging times of the drones are distributed at the QCMs, except for the last QCMs, along their paths. Note that the simulation results in Section VI-C show that the CBDN system with the minimum-charging-level method leads to a shorter average travel time than the CBDN system with the maximum-charging-level method.

2) Failure or Power Outage of a QCM: In this section, we consider emergencies caused by the failure or power outage of a QCM. In our CBDN system, the state (i.e., normal, failure, or power outage) of each QCM is crucial. When drones are flying on their scheduled paths, they should have alternative paths in the case of the failure or power outage of a QCM along its path. In order to obtain such alternative paths, we should execute another CBDN algorithm for emergencies and for new drone navigation paths.

- **CBDN Algorithm for Emergency**: Algorithm 2 shows the CBDN algorithm for emergencies. As shown in Algorithm 2, in the case of the failure or power outage of a QCM, we can use Algorithm 2 (a slight modification of Algorithm 1) in order to regain the paths of drones. The main difference is that the reachable QCMs are calculated by excepting the failure or power outage QCMs, and the flight time for a given battery capacity is calculated with the current remaining amount of the battery in order to select the first QCM among the reachable QCMs. Therefore, using Algorithm 2, drones can obtain new paths in cases of emergency.

- **Scheduling Sequence for New Drone Navigation**: When drones regain their paths after the failure or power outage of a QCM, their performances may vary according to the sequence of the navigation rescheduling of the drones related to the QCMs. If we consider the scheduling sequences of all of the drones related to the QCM, we can always obtain optimal results. However, it is computationally infeasible to consider the scheduling sequences of all drones related to the failure or power outage of the QCM, because it is an NP-hard problem to consider the scheduling sequences of all of the drones like in Section III-D. Therefore, like in Section III-D, when the drones related to the failure or power outage of a QCM are scheduling, we do not use the optimal algorithm to find the best paths, instead we use a heuristic algorithm in order to regain their paths.

- **Communication with TCC**: In the event of the failure or power outage of a QCM, drones cannot use that QCM, so the drones related to that QCM should receive information from the QCM in order to obtain an alternative path. However, frequent communication between the drones and the QCM has a significant impact on battery consumption. Therefore, in our CBDN system, rather than communicating directly between the drones and QCMs, the TCC periodically communicates with QCMs, and in the case of emergency with a QCM, the drones receive information from the TCC via QCMs. Of course, the drones can communicate directly with the TCC via cellular links such as 4G-LTE and 5G networks [9], [10].
TABLE I
SIMULATION CONFIGURATION

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drone network</td>
<td>The size of the map of drone network is 43.496 miles * 43.496 miles (i.e., 70 km * 70 km).</td>
</tr>
<tr>
<td>Number of drones</td>
<td>The default of the number of drones moving in drone networks is 80.</td>
</tr>
<tr>
<td>Average speed of drones</td>
<td>The average speed of drones is 30 MPH (i.e., 48.28 km/h) [46].</td>
</tr>
<tr>
<td>Drone-speed standard deviation</td>
<td>The drone-speed standard deviation is 3 MPH (i.e., 4.82 km/h) [46].</td>
</tr>
<tr>
<td>Number of QCMs</td>
<td>The default of the number of QCMs in drone networks is 40.</td>
</tr>
<tr>
<td>QCM performance</td>
<td>The default charging time at QCMs is 90 minutes from 0% to 100% [46].</td>
</tr>
<tr>
<td>Weight of delivery baggage</td>
<td>The default weight of delivery baggage and weight standard deviation is 2lb and 0.3lb [46].</td>
</tr>
</tbody>
</table>

Note that these considerations for the failure or power outage of a QCM may apply to other special emergencies (e.g., rerouting due to bad weather) as well.

VI. PERFORMANCE EVALUATION

In this section, the evaluation of CBDN performance is explained. A comparison of the CBDN system with the other baseline systems (i.e., SFT, SFWT, and IRN) was completed. The SFT system allocates a drone to the QCM in consideration of the flight time on the E2E-path as well as the charging time without considering the waiting times at QCMs. The SFWT system allocates a drone to the QCM in consideration of the flight time, the charging time, and the waiting time based on the QCM statistical information. The IRN system allocates a drone to the QCM in consideration of the flight time, the charging time, and the individual waiting time. The evaluation settings are as follows:

- **Performance metrics**: In order to evaluate the algorithmic performances, the average travel time of the drones and the average QCM utilization were used as the performance metrics.

- **Parameters**: In order to evaluate the algorithmic performances in various environments, the impact of (i) the number of drones, (ii) the drone speed, (iii) the drone-speed deviation, (iv) the number of QCMs, (v) the QCM performances, and (vi) the weight of the delivery baggage were investigated.

In order to prove the excellence of our CBDN, a drone simulation was implemented using the popular object-oriented modular discrete event network simulation framework called “OMNeT++” [45]. For this simulation, random source positions are applied for the drones, and a destination position and flight are applied according to these positions. The normal distribution $N(\mu_v, \sigma_v)$ of the drone speeds was used in the drone network for this simulation. Table I describes simulation configuration for a target drone network having drones and QCMs. Note that we used DJI’s “Matrice 200 Series” specification [46] for simulation configuration, because the “M200” ensures stable and long flight in strong winds through the use of high-performance motors and 17-inch propellers.

Fig. 11 shows snapshots of the drone simulation for the four battery-charging systems. As shown in Figs. 11(a), 11(b), 11(c), and 11(d), the dispersed drones using the CBDN system occupied more QCMs than the drones using the SFT, SFWT, and IRN systems for battery charging. As a result, according to the number of drones, drone speed, drone-speed deviation, the number of QCMs, QCM performance, and the weight of the delivery baggage in
the drone network, the CBDN obtained better results than the other baselines. The simulation confidence interval is 95%.

A. Behavior Comparison of the Battery Charging Systems

In this section, the behaviors of the four battery-recharging systems (i.e., SFT, SFWT, IRN, and GRN) are compared. For the behavior comparison, as shown in Fig. 12, the cumulative distribution functions (CDFs) were used to find the distributions of the drone travel times for these four systems. As shown in Fig. 12, the CDF of the CBDN increased more quickly than those of the other baselines. The CDF of the CBDN reached 1 when the travel time was approximately 769 min. Alternatively, the CDFs of the SFT, SFWT, and IRN reached 1 when the travel times were approximately 2870 min, 1048 min, and 1162 min, respectively.

B. Utilization of QCMs

In this section, the utilizations of QCMs for the four battery recharging systems (i.e., SFT, SFWT, IRN, and GRN) are compared. In comparing the utilizations of QCMs, as shown in Fig. 13, the average QCM utilization of the CBDN is shown to be the most uniform compared to the other systems, and this is due to the drone dispersal of the CBDN among multiple QCMs in order to prevent specific QCMs in the drone network from serving excessively heavy drone traffic for battery recharging.
C. Impact of the Battery Charging Level

This section investigates the impact of the battery charging level (i.e., Maximum Charging Level and Minimum Charging Level Methods) on performance in the CBDN system. Through the comparison of the battery charging levels, as shown in Fig. 14, the CBDN system with the minimum-charging-level method has a shorter average travel time than that with the maximum-charging-level method. This is because the CBDN with the minimum-charging-level method allows the drones to have more options in choosing less busy QCMs along their paths than that with the maximum-charging-level method.

D. Impact of the Number of Drones

This section investigates the impact of the number of drones on performance. As predicted, the average travel times of all four systems were lengthened as the number of drones increased. As shown in Fig. 15, the average travel times of the drones tended to increase according to the increasing number of drones; this is because a larger number of drones needs to use more QCMs to recharge the drone batteries. In the baselines for the SFT, SFWT, and IRN systems, long drone queues may form at the QCMs for battery charging, because these systems do not perform load balancing, as shown in Figs. 11(a), 11(b), and 11(c). These long drone queues lead to longer waiting times according to the increasing number of drones.

E. Impact of the Drone Speed

This section investigates the impact of the drone speed on performance. As in our prediction, as the drone speed increased, the performances of all four systems improved because the drones could quickly reach the QCMs and their destinations. According again to our predictions, the average travel times of all four systems reduced as the drone speed increased. As shown in Fig. 16(a), the average travel times of all four systems tended to decrease according to the increase in drone speed from 15 to 45 mph. This tendency was because the drones could quickly reach the QCMs and their destinations as the drone speed increased. In addition, for the QCM average utilization, as shown in Fig. 16(b), all four systems tended to increase according to the increase in drone speed from 15 to 45 mph due to the quick arrival at the QCMs as the drone speeds increased.

F. Impact of the Drone-Speed Deviation

This section investigates the impact of the drone-speed deviation on performance. Fig. 17 shows the impact of the
drone-speed deviation on the average travel time and the QCM utilization. The speed deviation is an important element because the IRN and CBDN systems use the reservation system with time in the QCMs. Contrary to our expectations, as shown in Figs. 17(a) and 17(b), the performances of all four systems were almost the same despite the increase or decrease of the speed deviation. This is because the minor deviation in the expected arrival time did not significantly affect the average travel time.

G. Impact of the Number of QCMs

This section investigates the impact of the number of QCMs on drone performance. As in our prediction, the average travel...
times of all four systems lengthened and the average QCM utilization of all of the systems decreased as the number of QCMs increased. As shown in Fig. 18(a), the tendency of the average travel time of all four systems is decreasing, because the drones could use a greater number of QCMs to charge their batteries according to the increase in the number of QCMs from 25 to 55. In addition, for the QCM average utilization, as shown in Fig. 18(b), all four systems tended to decrease, because the drones were scattered across more QCMs for battery charging according to the increase in the number of QCMs from 25 to 55.

H. Impact of the QCM Performance

This section investigates the impact of the QCM performance on drone performance. As in our prediction, the average travel time of all four systems lengthened, and the average QCM utilization of all of the systems increased as the QCM performance was reduced. As shown in Fig. 19(a), the average travel times of the drones tended to increase according to the reduction of the QCM charging performance. Alternatively, as shown in Fig. 19(b), the average QCM utilization tended to increase according to the reduction of the QCM performance. This tendency is because the duration of the drone battery charging is expected to be longer with the reduction of the QCM-charging performance.

I. Impact of the Weight of the Delivery Baggage

This section investigates the impact of the weight of the delivery baggage on drone performance. As in our prediction, the average travel times of all four systems lengthened as the weight of the delivery baggage increased. As shown in Fig. 20(a), the average travel times of the drones tended to increase according to the increasing weight of the delivery baggage. In addition, as shown in Fig. 20(b), the average QCM utilization tended to increase according to the increasing weight of the delivery baggage. This tendency is because the maximum flight time is shortened as the weight of the delivery baggage increases. Therefore, the drones use more QCMs as the weight of the delivery baggage increases.

Finally, through the simulation results, it is possible to conclude that the average travel time and average QCM utilization of the CBDN are the shortest and the highest, respectively, compared to those of the other three baselines. This finding is because the SFT does not consider the QCM waiting time, while the SFWT considers only the QCM statistical information, but not the QCM current states. Furthermore, the IRN system does not consider the additional waiting time required by the other drones due to the starting-drone battery charging. Therefore, the CBDN system can achieve superior performance compared to the SFT, SPWT, and IRN systems owing to its consideration of the flight time, charging time, and global waiting time at the QCMs.

VII. Conclusion

In this paper, we present a cloud-based drone navigation system called CBDN for finding coordinated globally drone routes in a drone network by avoiding QCM congestion for the battery recharging of drones. The CBDN system exploits the reservation schedule of QCM battery charging, along with the estimated travel time (i.e., the sum of the flight time, global waiting time, and charging time), based on the drone-traffic information using cloud-based management. We showed that the CBDN system outperformed the other baseline drone route finding systems through intensive simulations results. In this paper, we only focused on the feasibility of the proposed method on drone systems using battery charging because battery charging systems will be deployed for long-distance drone services. However, our technique is generic and can be applied applicable to drone systems using battery replacement. As future work, we will research how to strategically deploy QCMs within a specific area. In addition, based on the research on the deployment of QCMs, we will improve the accuracy of our simulation model by considering other environmental factors such as physical drone collision, wind and terrain (e.g., mountains, obstacles, infrastructure, and residential areas) which may significantly affect the drone flight time in the real world.

REFERENCES

Seokhwa Kim received the B.S. degree from the Division of Information and Communication, Baekseok University, in 2017. He is currently pursuing the M.S. degree with the Department of Computer Engineering, Sungkyunkwan University, under the supervision of Prof. J. Jeong. His research interests include Internet of Things and indoor positioning systems.

Jaehoon (Paul) Jeong received the B.S. degree from the Department of Information Engineering, Sungkyunkwan University, South Korea, in 1999, the M.S. degree from the School of Computer Science and Engineering, Seoul National University, South Korea, in 2001, and the Ph.D. degree from the Department of Computer Science and Engineering, University of Minnesota, Twin Cities, in 2009. He is currently an Associate Professor with the Department of Software, Sungkyunkwan University. His research areas include cyber-physical systems, Internet of Things, vehicular ad hoc networks, mobile ad hoc networks, wireless sensor networks, software-defined networking, and network functions virtualization. He is a member of ACM and the IEEE Computer Society. His two data forwarding schemes (called TBD and TSF) for vehicular networks were selected as spotlight papers in the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS in 2011 and in the IEEE TRANSACTIONS ON MOBILE COMPUTING in 2012.

Hyoungshick Kim received the B.S. degree from the Department of Information Engineering, Sungkyunkwan University, in 1999, the M.S. degree from the Department of Computer Science, KAIST, in 2001, and the Ph.D. degree from the Computer Laboratory, University of Cambridge, in 2012. He was a Post-Doctoral Fellow with the Department of Electrical and Computer Engineering, The University of British Columbia. He was a Senior Engineer with Samsung Electronics from 2004 to 2008. He also served as a member of DLNA and Coral Standardization for DRM interoperability in home networks. He is currently an Assistant Professor with the Department of Computer Science and Engineering, College of Software, Sungkyunkwan University. His current research interests are focused on social computing and usable security.

Hyoungshick Kim received the B.S. degree from the Department of Information Engineering, Sungkyunkwan University, in 1999, the M.S. degree from the Department of Computer Science, KAIST, in 2001, and the Ph.D. degree from the Computer Laboratory, University of Cambridge, in 2012. He was a Post-Doctoral Fellow with the Department of Electrical and Computer Engineering, The University of British Columbia. He was a Senior Engineer with Samsung Electronics from 2004 to 2008. He also served as a member of DLNA and Coral Standardization for DRM interoperability in home networks. He is currently an Assistant Professor with the Department of Computer Science and Engineering, College of Software, Sungkyunkwan University. His current research interests are focused on social computing and usable security.

Jung-Soo Park received the B.S., M.S., and Ph.D. degrees from the Department of Electronics Engineering, Kyungpook National University, in 1992, 1994, and 2013, respectively. He has been a Principle Researcher with the Electronics and Telecommunications Research Institute since 1994. His research interests are network security, Internet of Things, machine to machine, network functions virtualization, vehicular networks, wireless sensor networks, and mobile ad hoc networks and blockchain.

Taeho Kim received the B.S. degree from Sungkyunkwan University in 1995 and the M.S. and Ph.D. degrees from the Department of Computer Science, Korea Advanced Institute of Science and Technology, in 1997 and 2005, respectively. He is currently the Managing Director of the Dependable CPS Research Group, Electronics and Telecommunications Research Institute. His research interests are safety-critical and intelligent cyber-physical systems, system software, and software engineering.