國 立 成 功 大 學

工 業 與 資 訊 管 理 研 究 所

碩 士 論 文

執行區域覆蓋任務之可充換電無人機及其載具之最佳聯合路線規劃問題研究

An Optimal Joint UAV and Ground Vehicle Path Planning Problem for Area Coverage with Energy Consideration

研 究 生 ：馬哈曼 Muhammad Meidy Nur Hafidz

指導教授：王逸琳 教授

中 華 民 國 一 百 零 八 年 五 月

doi:10.6844/NCKU201901976

國立成功大學

碩士論文

執行區域覆蓋任務之可充換電無人機及其載具之最佳聯合路線規劃問題研究

An optimal joint UAV and ground vehicle path planning problem for area coverage with energy consideration

研究生：馬哈曼

本論文業經審查及口試合格特此證明

論文考試委員：王逸琳 莊坤達 陳穆 吳政翰

指導教授：王逸琳

系(所)主管：

中　華　民　國　108　年　5　月　26　日

doi:10.6844/NCKU201901976

摘要

無人機（UAV）可能是欲在短期間內完成大片地域全面搜索掃描（亦即覆蓋）任務的唯一手段，這些區域覆蓋任務若僅靠地面人力將難以快速完成，這是因為地面區域之間可能沒有方便的道路連結；反之，空中 UAV 之移動不受地域連結的障礙，較可輕易地自空中對目標地面區域展開搜索、監視或拍攝照片。然而 UAV 必須擔心續航力是否足夠，若欲持續其覆蓋任務，必須同時規劃其充換電行程；本研究擬以載有充換電設施的車輛（GV）當成行動式的動態充換電基地，讓 UAV 可在 GV 上進行充換電作業。因此，GV 與 UAV 兩者的路線必須聯合規劃。

假設我們將空域劃分割成數個可覆蓋所有目標地面區域的虛擬空間單元（空域節點），而相鄰的空域節點間構成空域節線，則目標地面區域上的掃描任務，將可使用 UAV 在其相應的空域節點滯空巡邏（亦即覆蓋）一段特定時間來完成。因此，在給定必須被覆蓋的地面區域後，本研究擬規劃最佳無人機路線，以在最短時間內可覆蓋所有必要的空域節點，同時其飛行路線必須隨時顧及足夠的續航力。由於無人機的電池功率有限，它們可先由 GV 攜帶，GV 在地面道路可停靠之點為地域節點，而 GV 在地面可能移動的路網稱為地域網路。UAV 可由 GV 在其地域網路的某些地域節點上發射、降落以充換電池。以網路圖形而言，共有 UAV 的空域網路與 GV 的地域網路，而這兩層網路間再由 UAV 可能的起飛或降落之地空節線所串接。因此，GV 路線僅在地域網路上，而 UAV 路線則橫跨空域與地域網路，UAV 與 GV 路線重合即可充換電。

本研究探討 GV 和 UAV 的最佳路線規劃方式，以便 UAV 在最短總時間內能掃描完所有目標地面區域，並考量 UAV 的續航力限制。為了模擬 UAV 和 GV 的聯合作業方式，我們將原問題轉換成在空地兩層網路與其間連結上的「覆蓋路徑規劃問題」以及「二階層式路徑規劃問題」，以時空網路為基礎提出兩種整數規劃數學模型：單一 GV 配單一 UAV（$M_{1\text{–}V}^{1\text{–}D}$）、以及多台 UAV（$M_{1\text{–}V}^{k\text{–}D}$）等兩種情境。其中，$M_{1\text{–}V}^{k\text{–}D}$ 可能是第一個可以同時處理單一 GV 配多台 UAV 路線的數學規劃模型。雖然上述模型可精準地規劃出最佳 GV 與 UAV 的詳細動態，但其求解十分耗時。

為了在更短的時間內計算好路線，我們針對 $M_{1\text{–}V}^{1\text{–}D}$ 設計了一個在時空網路中以深

doi:10.6844/NCKU201901976

度優先演算法為基礎的貪婪算法，試圖在 T 期限內為 UAV 找到一條可覆蓋所有目標空城節點的可行路線，再用類似二分搜尋法概念以逼近最佳之 T 值。數值測試結果顯示，我們的貪婪算法通常能在短時間內獲得不錯的可行解，至於多台 UAV 的啟發式演算法則有賴未來能發展更有效率的責任區域劃分方式來加以處理。

關鍵詞：覆蓋路徑規劃，時空網路，聯合作業，整數規劃，深度優先。

doi:10.6844/NCKU201901976

Abstract

The Unmanned Aerial Vehicles (UAVs) may be the only means by which to conduct coverage missions that search, monitor, or take photos from the air space over target ground areas that are too difficult to reach by ground manpower. Suppose we divided the air space into several virtual cells that cover all the target ground areas. Then, a scan mission on a target ground area can be done by a UAV flying over (i.e., covering) its corresponding virtual cell in air space for a specific period of time. In this research, we plan to route UAVs to cover all necessary virtual cells. Since UAVs have limited battery power, they can be carried by some mobile Ground Vehicles (GVs) and then be launched from and retrieved by GVs on some nodes in the ground road networks.

This thesis investigates how to calculate optimal routes for both GVs and UAVs so that all the target ground areas can be scanned by UAVs within a minimum total time subject to the battery limitation of the UAVs. To model the combined operations by the UAVs and GVs, we use the coverage path planning (CPP) and two-echelon routing problems as our approaches. We propose two mathematical models based on integer programming techniques over a time-space network. We consider one GV with one UAV case (M_{1-V}^{1-D}) and one GV with multiple UAVs case (M_{1-V}^{k-D}). To the best of our knowledge, M_{1-V}^{k-D} might arguably be the first mathematical programming model that can deal with routings for both GV and multiple UAVs at the same time. Our proposed model can calculate the detailed movement for both UAVs and GV, yet it is very time-consuming.

To calculate good routings in shorter time, we also designed a greedy algorithm for the one GV with one UAV case, where a depth-first search mechanism is exploited to find a feasible UAV route in the time-pace network that covers all target areas within the time limit *T*. The computational experiments indicate our greedy algorithm calculates a good solution faster than the proposed integer programming model.

Keywords: coverage path planning, time-space network, combined operations, integer programming, depth-first search.

doi:10.6844/NCKU201901976

Acknowledgments

First of all, I would like to express my gratitude to my advisor, Prof. I-Lin Wang (王逸琳) that accepts me to be part of his lab and for his dedication, guidance and patience so that this thesis is completed. There are many things that I got from him during my study at IIM NCKU which made my study progress smoother and it is also valuable and useful for my future. I also would like to thank all committee members, Prof. Kao-Chiang (高強), Prof Yeu-Shiang Huang (黃宇翔), Prof Cheng-Han Wu (吳政翰), Prof Kun-Ta Chuang (莊坤達) and Prof Sheng-I Chen (陳勝一) who are willing to take the time to review the paper and put forward many valuable suggestions to make this thesis better.

In addition, I would like to thank everyone at I-Lin lab. To my seniors, B.K. (歐柏寬), Si-Han (方思涵) and Guan-Wei (何冠緯). To my classmates, Jib (李寶玉), Chia-Hao (廖嘉豪), Garry (呂昀軒), and Xue-Mei (孫雪湄). To my juniors, 岳晏慈, 陳彥瑋, 王宗瀚, and Ari. Thank you all for making me being able to face many struggling and hardship, willing to have a discussion and sharing, giving me suggestions or making this lab lively so that I can forget my stress and worry. I must say that with you guys here, I am able to have an enjoyable life here in Taiwan and it becomes one of the precious moments in my life.

Finally, I want to express my gratitude to my mother and my family for their never-ending support. They are my strongest backing and a reason to keep moving forward no matter how big the difficulties that I have during my study. With many things that they worried about, they still support my decision to pursue my degree in Taiwan. I am feeling happy and grateful for their understanding so that I can focus more on my study. I also thank all of you whom I can't mention one by one that also has a contribution to this thesis.

Muhammad Meidy Nur Hafidz (馬哈曼)

Department of Industrial and Information Management

National Cheng Kung University, Tainan City, Taiwan R.O.C.

May 2019

doi:10.6844/NCKU201901976

Table of Contents

doi:10.6844/NCKU201901976

doi:10.6844/NCKU201901976

List of Figure

doi:10.6844/NCKU201901976

List of Table

doi:10.6844/NCKU201901976

Chapter 1 Introduction

1.1. Research Background

Recently, unmanned aerial vehicles (UAVs) became popular for both civilian and public purposes, (Vachtsevanos & Valavanis, 2015) because they are reliable, inexpensive, and easy to use and maintain. Applications of UAVs vary, from individual and business needs including self-entertainment, photography, package delivery, and precision agriculture to the public domain in areas such as law enforcement and disaster management.

One of the uses of UAVs is to take aerial images or videos. This images and videos provide information that is useful for several requisites, including mapping, surveying, and disaster assessment. Satellite and traditional aerial systems (e.g., helicopters) can also be used for these purposes. Each tool has its advantages, and they can even be used together to increase the quality of information.

Both satellite and traditional aerial system imagery have advantages in terms of the coverage area but usually have lower resolutions than UAVs. Figure 1.1 provides a comparison of UAV and satellite image resolution. In addition to its advantages in terms of image resolution, UAV also has more flexibility and a faster response compared to satellites. Another advantage is that UAVs can obtain more detailed, clearer images than satellites and traditional aerial systems, especially when there is cloud cover.

Figure 1.1. Comparison of UAV and satellite imagery (FSD, 2016)

doi:10.6844/NCKU201901976

However, UAVs have an operational limitation compared to satellites and traditional aerial systems because those used for civilian and public purposes mostly use a battery as the power source and are small in size. These batteries usually last for less than an hour on average for each flight. This becomes a disadvantage for UAVs when the coverage area is large or there is a need for a long monitoring period.

Satellite images usually have a lower resolution than UAVs, it is possible to obtain a high-resolution image although it may not be efficient. High-resolution images that are several days to a week-old cost around euro 250-400 per 25 km^2, and the lead time is relatively long, ranging from a half a day to four days (FSD, 2016). This makes UAVs a better option than satellites, especially if the high resolution, up-to-date images are a priority.

Combining ground vehicles (GVs) with UAVs has been suggested to enhance the UAV flight range. GVs are also able to serve as mobile battery recharge stations that can be used to increase UAV operational time. The GV can traverse along a road and launch UAVs at possible launch points, such as parking spaces. For civilian and public purposes, combining GVs with UAVs is common in logistic, law enforcement, and search and rescue operations. Figure 1.2 gives two examples: (a) the left picture shows a drone parcel delivery done by Amazon and UPS for a logistic application. (b) the right picture is that the Red Cross partnered with Land Rover to develop a car that could carry UAVs and help in search and rescue operations as part of Project Hero. In addition to these two examples, in the area of law enforcement, the Dubai Police Force have utilized a drone for traffic management.

Figure 1.2. Application combining UAV and GV operation

The combined operation of UAVs and GVs can also be implemented for other applications, such as mapping and surveying. This has, especially in the domain of logistics, attracted many researchers. However, very few studies have examined the area coverage domain (Otto, Agatz, Campbell, Golden, & Pesch, 2018). These facts and our interest in combined operation of UAVs and GVs motivated this research.

doi:10.6844/NCKU201901976

1.2. Research Objective

The research objective was to develop mathematical models for the combined operation of UAVs and GVs for monitoring or taking an aerial image of some target areas. These mathematical models were developed for two scenarios. The first is one GV carrying one UAV, and second is one GV carrying multiple UAVs. These mathematical models are expected to have these capabilities:

1. Determine the route and schedule for both a GV and UAV simultaneously.
2. Minimize the total time of the task subject to the UAV battery power limitation.

1.3. Research Scope

In this research, we focus on the combined operation of UAVs and GVs where the combined operation is defined as follows: the UAV takes an aerial image of some given area while GV act as a supporter of the UAV. The UAV freely flies in the airspace and moves with the GV as it moves from one point to another point in the ground space, where the GV is moving in the ground space only. The combined operation goal is where the UAV performs a monitoring mission or takes an aerial image of a given area with the time for task completion minimized.

In order to model the combined operation, we will combine two methods. The first method is a coverage path planning (CPP) approach to model the UAV route. In this approach, we will use a grid-based method to divide the airspace workspace and change it into a graph. Next, we will combine the ground space and airspace into one graph and then use a two-echelon routing problem concept for routing the UAV and the GV simultaneously. The original graph will be changed into the time-space network method to solve the problem. Details of the model and the theoretical relationships are explained in Chapter 3.

1.4. Thesis Outline

The rest of this thesis proposal is organized as follows: In Chapter 2, we provide some prior studies and theories that support the current research, together with our research contribution. The problem definition and formulation including the mathematical models will be explained in Chapter 3. In Chapter 4, we present the proposed heuristic algorithm. In Chapter 5, we present our computational experimental results for the mathematical models described in Chapter 3. Lastly, in Chapter 6 we present the conclusions of our research.

doi:10.6844/NCKU201901976

Chapter 2 Literature Review

In this chapter, we discuss UAVs, theories related to coverage path planning (CPP), combined UAV and GV operation, and the concept of the two-echelon routing problem with some relevant research papers used as research references, and the contribution of our research.

2.1. Unmanned Aerial Vehicles (UAVs)

Dalamagkidis (2015) defined an unmanned aerial vehicle (UAV, also known as a drone) as a pilotless aircraft or a flying machine without an onboard human pilot or passenger. The term "unmanned" implies the total absence of a human who directs and actively pilots the aircraft. Control functions for unmanned aircraft may be either onboard or off-board.

Originally, UAVs were developed and used exclusively for military purposes. Nevertheless, UAVs eventually became popular for civilian and public purposes. According to Vachtsevanos & Valavanis (2015), 2013 was a critical turning point for UAV applications for civilian use. In this section, we provide information related to the characteristics of UAVs, as well as their applications for civilian and public purposes.

2.1.1. UAV Characteristics

Most of the UAVs used for civilian and public purposes are categorized as either fixed-wing or multirotor. There is also another UAV with the inherent characteristics of both fixed-wing and multirotor UAVs. In this section, we only explain the fixed-wing and multirotor UAVs based on Vergouw, Nagel, Bondt, & Custers (2016).

A fixed-wing UAV usually refers to an aircraft that uses fixed, static wings in combination with a forward airspeed to generate lift (Vergouw et al., 2016). Fixed-wing UAVs required a large space for take-off and landing compared to multi-rotor UAVs. However, fixed-wing UAVs usually have better endurance and are thus more suitable for use in a large area. Examples of fixed-wing UAVs are the Ebee SQ- SenseFly, Husban H301S Spy Hawk, Skywalker FPV Black X8, and Parrot Disco.

Multi-rotor UAVs are a subset of rotorcraft. A rotorcraft is an aircraft that use rotary wings to generate lift. Rotorcraft can have one or multiple rotors. Most rotating system UAVs use multiple small rotors, which are necessary for their stability (Vergouw et al., 2016). This is why the term multi-rotor is used even though there are one rotor UAVs.

doi:10.6844/NCKU201901976

Examples of multi-rotor UAVs are the DJI Phantom Pro, DJI Mavic Pro, Yuneec Drone Tornado H920, and the Walkera F210 Deluxe Racer. Figure 2.1 shows the difference between fixed-wing and multi-rotor UAVs.

Figure 2.1. UAV type: fixed-wing UAV (left), multi-rotor UAV (right)

2.1.2. UAV Applications

UAVs are beneficial for many civilian and public purposes. However, they require special clearance and requirements in many countries. This has caused some UAV-related research to be conducted based on simulation scenarios rather than a real case scenario. Even so, the possibility of using UAVs in the future is promising. Figure 2.2 shows the potential applications of UAVs for civilian and public purposes.

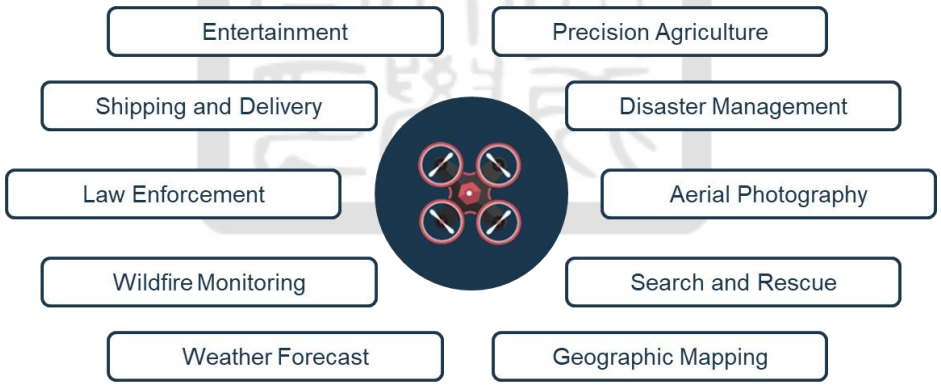

Figure 2.2. Application of UAVs for civilian and public purposes.

In this research, we consider a situation where UAVs will be launched to take aerial images or videos in a given area. This situation is known as area coverage since the UAV has to cover some area with a limited footprint. The developed model is suitable for several applications, including precision agriculture, disaster management, search and rescue, aerial photography, and geographic mapping.

doi:10.6844/NCKU201901976

2.2. The Combined Operation of GVs and UAVs

UAVs invariably have operational constraints in terms of both range and time, especially for micro, mini, and small UAVs typically used for civilian and public purposes. Otto et al. (2018) suggested that utilizing another means of transportation and combining it with UAVs could enhance the effectiveness and efficiency of UAV operation. This transport refers to a robot or carrier. We call this transport as a ground vehicle (GV).

According to Otto et al. (2018), most studies involve the combined operation of vehicles and UAVs to determine a route for a set location, meaning that the UAV must visit a discrete set of addresses. Only a few research papers have studied different operations such as area coverage, where UAVs have to cover a specific area with a sensor that has a limited footprint or search operation, where UAVs must discover a stationary or moving target.

According to the UAV and the GV interaction characteristics in the combined operation described by Otto et al. (2018), the current research falls into the GV supporting UAVs operation category. In this category, a UAV acts as the main performer while the GV acts as the UAV supporter. Some studies, including our research, assume that the UAV can recharge each time it lands on the GV, where other studies do not consider recharging during each landing. Figure 2.3 illustrates the combined UAV and the GV interaction operation where the interaction characteristic is the GV supporting UAV operation.

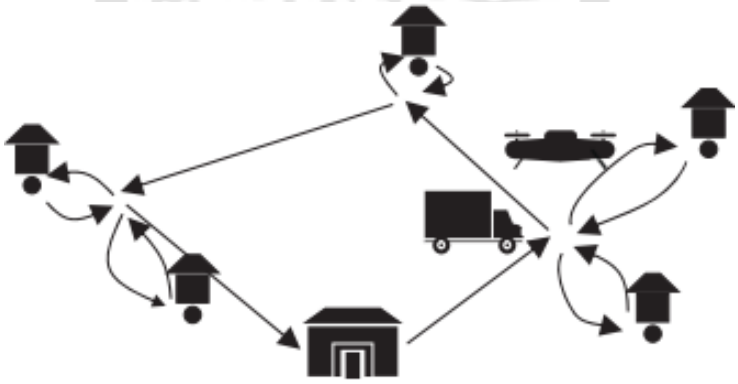

Figure 2.3. The GV supporting operation of UAVs (Otto et al., 2018)

There are several studies related to the combined operation of the GV and UAV where the GV supports the UAV operation. Some of them also use a two-echelon routing problem concept. In this section, we only provide descriptions of those not using a two-echelon routing problem as their approach.

doi:10.6844/NCKU201901976

Garone, Naldi, Casavola, & Frazzoli (2010) researched path planning for a class of carrier vehicles in which a slow carrier with an infinite operating range cooperated with a carrier vehicle, which, on the contrary, was faster but had a limited range. They describe a sea rescue scenario in which a drone must visit a given set of visit point and develop a heuristic algorithm based on the Euclidean TSP, called the Carrier/Carried-TSP (CC-TSP) to determine an optimal sequence necessary to minimize the sum of the Euclidean distances among consecutive points.

Mathew, Smith, & Waslander (2013) studied how to maintain continuity in persistent coverage and surveillance tasks with coordinated teams of autonomous robots. These robots consist of charging robots and drones, where the objective was to find the paths of the recharging robots with the minimum total cost so that each drone met one of the robots exactly once within its specified time window. They formulated the problem as mixed-integer linear programming (MILP) and then converted it into a partitioned directed acyclic graph. The Noon-Bean transformation and the LKH solver were used for a single charging robot case with larger instances.

Mathew, Smith, & Waslander (2015a) extended the problem discussed in Mathew et al. (2013). They investigated a cooperative replenishment strategy for a team of drones performing a surveillance task using one or more mobile charging robots. The problem extension was a longer planning horizon using a receding horizon (RH) and a fixed horizon approach.

Mathew, Smith, & Waslander (2015b) researched a task scheduling and path planning problem for a team of cooperating vehicles performing autonomous deliveries in an urban environment. The vehicle consisted of trucks and quadrotor drones. They formulated a heterogeneous delivery problem (HDP) as a discrete optimal path planning problem. The solution consisted of routes computed for the truck and the quadrotor through the graph that minimized the total cost of deliveries.

Tokekar, Hook, Mulla, & Isler (2016) studied planning for the symbiotic vehicle-drone paths used to obtain aerial measurements in a precision agriculture application. An unmanned ground vehicle (UGV) can carry a drone, which must take aerial images of as many points of interest as possible to gather information for classification of soil based on the nitrogen level. The drone does not refuel on the UGV. The problem was modeled as an orienteering problem by constructing a complete graph with metric edges.

doi:10.6844/NCKU201901976

Ferrandez, Harbison, Weber, Sturges, & Rich (2016) investigated the effectiveness of using a truck-drone combination in a delivery network. They use time and energy as a parameter. In their problem, the drone can only carry at most one package per sortie. They proposed an algorithm using K-means and a genetic algorithm to determine an optimalnumber of launch sites and location, delivery requirements, and drones per truck.

Table 2.1 summaries the studies related to the vehicle-drone operation.

Table 2.1. Vehicle-drone combined operation-related studies

Literature	**Method**	**Drone Operation**	**Refuel**	**Application**
Garone et al. (2010)	M, H, Exp	Routing set of location	v	SAR
Mathew et al. (2013)	M, H, Exp	Routing set of location	v	ISR
Mathew et al. (2015a)	M, H, Exp	Routing set of location	v	ISR
Mathew et al. (2015b)	H, Exp	Routing set of location	v	Delivery
Tokekar et al. (2016)	H, Exp	Area Coverage		Agriculture
Ferrandez et al. (2016)	H, Exp	Routing set of location	v	Transport
Our research	**M, H, Exp**	**Area Coverage**	**v**	**General**

Table 2.2 lists the description of notations used in Table 2.1 and later in tables 2.3 – 2.4.

Table 2.2. Glossary of abbreviations

BnC	Branch-and-cut algorithm
M	Mathematical model ready for input into an off-the-shelf solver (e.g., Gurobi).
H	One or several customized heuristic algorithms. Literature may provide memory and computational complexity analysis of the proposed algorithms.
Exp	Computational experiments on artificial or real-world data

2.3. Two-Echelon Location Routing Problem

According to Cuda, Guastaroba, & Speranza (2015), multi-echelon refers to a supply chain system situation where the movement of goods from the origin to the destination usually pass through several stages, which represents one level of the distribution network

doi:10.6844/NCKU201901976

known as an *echelon*. A multi-echelon network with only two stages is called a *two-echelon* network. In the two-echelon case, after goods leave their origin, they pass through exactly one level of the intermediate facility before arriving at their destination.

Cuda et al. (2015) defined the two-echelon location routing problem (2E-LRP) as two-echelon distribution networks consisting of three disjoint sets of vertices corresponding to potential locations for the origins (i.e., the depots), the satellites (i.e., the intermediate facilities), and the destination (i.e., the customers), respectively. The destination location is assumed to be fixed and known beforehand, but the origins and satellites that are used are not determined a priori.

This network then is decomposed into two echelons. The first echelon comprehends the links between the origins and the satellites and the connecting pairs of satellites. The second echelon consists of satellites and the destinations with the connecting pairs of satellites. There are two kinds of vehicles in the network, one at each echelon. Vehicles belonging to the first echelon are called *primary vehicles*, while those in the second echelon are called *secondary vehicles* (Cuda et al., 2015).

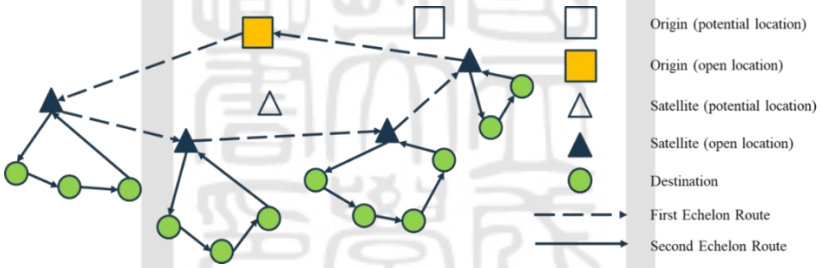

Figure 2.4. The concept for the two-echelon location routing problem (Luo et al.,2017)

We are using 2E-LRP as our reference since it has similarity concept with our problem. According to a 2E-LRP concept, where the GV is the primary vehicle, and the UAVs comprise the secondary vehicle. They start from the depot together and then move to a rendezvous node (i.e., parking space), where the UAV is launched into airspace so the UAV can begin to cover all areas. The UAV is then retrieved after the task is finished. The network for this problem is similar to the 2E-LRP concept, where the ground space is the first echelon, and the airspace is the second echelon. Another similarity is the selection of nodes from a set of rendezvous nodes. In the UAV and GV combined operation, all the nodes in the ground space may not need to be visited; thus selecting the nodes is necessary.

While various 2E-LRP approaches are similar, there are some differences between the

doi:10.6844/NCKU201901976

classic 2E-LRP and a 2E-RP used for the vehicle-drone operation. Luo, Liu, & Shi (2017) identified two different points, spatial cooperation, and temporal cooperation, where a spatial cooperation difference implies that each vehicle has a spatial route relationship. In the classic 2E-LRP, these routes are independent of each other, meaning that a route change for one vehicle will not affect the others. However, in a vehicle-drone operation, they are dependent are one another.

A temporal cooperation difference refers to the vehicle routing periods. For the classic 2E-LRP, usually, the vehicle that serves the first echelon is required to distribute large amounts of products in a low-period cycle. In contrast, in the second echelon, there is a longer period with a smaller amount of product. In this case, the routing period for each vehicle is independent. However, for the vehicle-drone operation, the routing period is simultaneous since there are no period cycle differences.

Manyam, Casbeer, & Sundar (2016) studied a cooperative vehicle routing problem (VRP) for a drone and vehicle on a surveillance and reconnaissance (ISR) mission. In their model, the drone does its job within a given range while the vehicle stays in its initial position until the task is completed in each flight. After that, they move together to other regions and repeat the process until all jobs are finished. This model is proposed for the one-vehicle-one-drone case.

Luo et al. (2017) developed a mathematical model called a two-echelon ground vehicle and UAV routing problem (2E-GU-RP) for an ISR mission. In their research, they assumed that there were enough rendezvous nodes in the road network. The vehicle can thus always find a rendezvous node on the following trip to meet the drone after it launches the drone from some given node. This means that each road arc traversed by the vehicle corresponds to a flight route of the drone. They also proposed two heuristics to solve a large case.

Luo et al. (2018) studied the same problem as in Luo et al. (2017), where the assumption for the vehicle was relaxed. The vehicle could wait in the current node while waiting for the drone to execute the task or while it was heading to the next rendezvous node. In the previous study, waiting for the drone in the current node was prohibited. Also, in this study, they did not provide any heuristic model.

Hu et al. (2018) also researched vehicle-drone routing for an ISR mission. They relaxed the drone number from one into multiple drones. To the best of our knowledge, this research was the first to consider a case of one vehicle with multiple drones. They developed an efficient algorithm without a mathematical model.

doi:10.6844/NCKU201901976

Table 2.3. provides a summary of the research related to the two-echelon routing problem vehicle-drone operation.

Table 2.3. 2E-RP related researches

Literature	Method	GV	UAV	Application
Manyam, et al. (2016)	M, BnC, Exp	Single	Single	ISR
Luo, et al. (2017)	M, H, Exp	Single	Single	ISR
Luo, et al. (2018)	M, Exp	Single	Single	ISR
Hu, et al. (2018)	H, Exp	Single	Multiple	General
Our research	**M, H, Exp**	**Single**	**Single, Multiple**	**General**

2.4. Coverage Path Planning

Coverage path planning (CPP) is a task defining a path that passes over all points of an area or volume of interest while avoiding obstacles (Galceran & Carreras, 2013). Barrientos et al. (2011) mentioned that CPP has two approaches, online and offline. Online approaches are mostly sensor-based navigation schemes, where the robot has to cover an unknown environment. Offline approaches typically require the robot to cover areas where boundaries and obstacles in the area are known to the path planner in advance.

CPP has two major steps: area decomposition, and optimization. Area decomposition is a process in which a given area is divided into obstacles and a target configuration. Most CPP algorithms decompose the target space into sub-regions called cells. Galceran and Carreras (2013) explained area decomposition techniques that were used by literature related to CPP. In the current study, referring to their study, grid-based methods are used.

Grid-based methods use a representation of the environment decomposed into a set of uniform grid cells. This method was introduced by Moravec & Elfes (1985) for mapping an indoor environment using a sonar ring mounted on a mobile robot. Typically, each grid cell is a square. Because a grid representation only an approximate the shape of the target region and its obstacles, this approximation makes the grid-based method “resolution-complete,” meaning that its completeness depends on the resolution of the grid map (Galceran & Carreras, 2013). In some literature, the grid-based method is known as ‘approximate cellular decomposition.’ This term comes from the classification done by Choset (2001). An example of a grid-based map is shown in Figure 2.5.

doi:10.6844/NCKU201901976

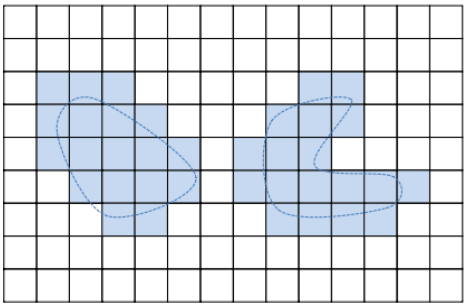

Figure 2.5. Grid-based method (Galceran & Carreras, 2013)

After decomposition, the optimization step can be started. A common approach used to solve the CPP problem is to mimic the classic traveling salesman problem (TSP) to cover the sub-regions within a decomposed area (Khan, Noreen, & Habib, 2017). This is the reason some of the literature like the study by Galceran & Carreras (2013) mentioned that CPP is related to the TSP or the covering salesman problem (CSV), a variant of the TSP.

Avellar, Pereira, Pimenta, & Iscold (2015) researched the minimum time coverage of ground areas using multiple drones. They used an optimal coverage method as the area decomposition technique and the vehicle routing problem (VRP) approach to model the problem after decomposition. In their problem, they considered using multiple fixed-wing drones. The objective was to minimize mission time. Since there were multiple drones, the goal could be reached by minimizing the time of the longest route among the routes of all of the drones.

Nam, Huang, Li, & Xu (2016) studied coverage path planning for a drone conducting a survey mission. They used the grid-based method as the area decomposition technique and a wave-front algorithm to solve the problem. In their research, they considered the drone's camera field of view (FOV). This factor was used to obtain the size of the rectangles for the grid cell, together with the overlap rates required in the mosaicking process. The objective of this research was to minimize the completion time for the drone. In their problem, they only considered one drone.

Nedjati, Izbirak, Vizvari, & Arkat (2016) conducted a rapid damage assessment using multiple-drones in a post-earthquake response system. Drones were deployed to gather information from the earthquake site by taking the images around the affected area. They use the grid-based method as the area decomposition technique and mixed-integer linear programming (MILP) to model their problem. In their problem, they considered multiple

doi:10.6844/NCKU201901976

usages of a drone. The proposed problem had a minimax objective function that minimized the maximum traveling time of each drone to finish the job.

Pham, Bestaoui, & Mammar (2017) attempted to maximize the CPP while minimizing the path length of a drone in an agriculture application with a concave obstacle. In their research, they proposed a new cellular decomposition based on a generalization of the Boustrophedon variant, using Morse functions, with an extension of the representation of the critical points as the area decomposition technique. To solve the problem, they used the TSP algorithm and a genetic algorithm (GA) to obtain the shortest path.

Choi, Choi, Briceno, & Mavris (2018) researched the CPP for a drone imagery mission. They proposed a vehicle-routing based approach using a column generation method to solve the problem. In their model, they considered a turn motion penalty in a cost function since the CPP in a conventional vehicle-routing-based approach cannot capture a turning motion.

A summary of research related to the two-echelon routing problem for vehicle-drone operation is provided in Table 2.4.

Table 2.4. CPP related studies

Literature	Method	Area Decomposition	Drone	Application
Avellar et al. (2015)	M, Exp	Optimal coverage	Multiple	General
Nam et al. (2016)	H, Exp	Grid-based	Single	General
Nedjati et al. (2016)	M, Exp	Grid-based	Multiple	Disaster Management
Pham et al. (2017)	H, Exp	Cellular decomposition	Single	Agriculture
Choi et al. (2018)	M, Exp	Column generation	Multiple	General
Our research	**M, H, Exp**	**Grid-based**	**Single, Multiple**	**General**

2.5. Research Contribution

According to Otto et al. (2018), most of the research that involves combined vehicle-UAV operation mainly have used routing for a set of locations as the UAV operation type. Few articles consider other operations, such as area coverage. In this thesis, we intend to solve the area coverage operation problem. However, the way we solve our problem may intersect with the routing of a set of location since we discretize the given area and represent it as a graph as an approximation.

doi:10.6844/NCKU201901976

In our research, we attempt to develop the mathematical models and solution to calculate the path and schedule for a UAV and GV in a combined operation. This path is expected to minimize the total time required for the UAV to cover or monitor the given target areas. Using this model, it is also expected that the path and schedule generated will not violate the UAV operational limit, which in this model is the battery power limitation.

Some research such as that conducted by Luo et al. (2017) assumed the battery limitation to be the maximum flight time. For example, a full battery allows the UAV to fly for 30 minutes. In our problem, we consider the battery limitation in a level form (e.g. a full battery is equal to 100 level units), so the travel time and energy consumption are not the same things. It is reasonable to separate time and energy since it is possible for the route to have the same travel time but to require different amounts of energy. Another contribution is that when most of the related research only models a case of one GV with one UAV, we also attempt to model a case with multiple UAVs.

doi:10.6844/NCKU201901976

Chapter 3 Mathematical Programming Models for Planning the Joint UAV and GV Paths

This chapter introduces the modeling of joint UAV and ground vehicle path planning problem for area coverage with energy consideration. In order to obtain the exact solution, we propose two mixed-integer programming models: (1) a one GV with one UAV model (M_{1-V}^{1-D}), and (2) a one GV with multiple UAVs model (M_{1-V}^{k-D}). Figure 3.1 provides a schematic diagram of the proposed mathematical models.

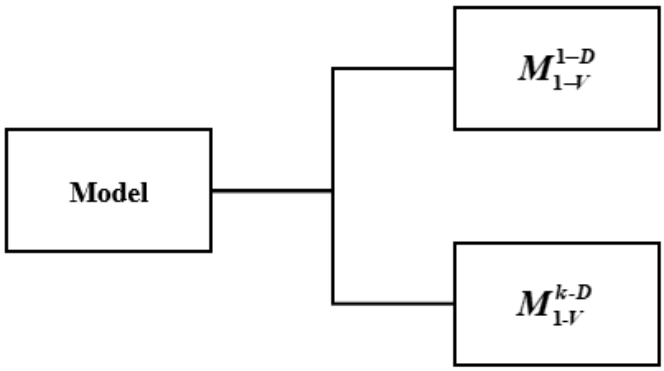

Figure 3.1. Schematic diagram of the proposed mathematical models

3.1. Modeling Approach

In this problem, there are two types of workspaces. The first is the airspace that is exclusively for the UAV, and the second is the ground space that is accessible for both the GV and the UAV. Under actual conditions, the UAV may fly freely in the airspace. In order to make a route for the UAV, we have to decompose the airspace area into sub-regions. We use a grid-based method that involves a decomposition technique that divides the airspace into a virtual uniform grid cell.

Suppose that we already have done the pre-processing step and obtained the grid map for the airspace and the network graph for the ground space. We can transform this grid map into a network graph by changing the virtual cells into nodes and the connections between each cell into an arc. As for the ground space, the nodes represent the potential places for the GV to launch the UAV and the arcs represent roads. We then combine these two workspaces. Figure 3.2 shows the network graph for the combined operation of the GV and UAVs.

doi:10.6844/NCKU201901976

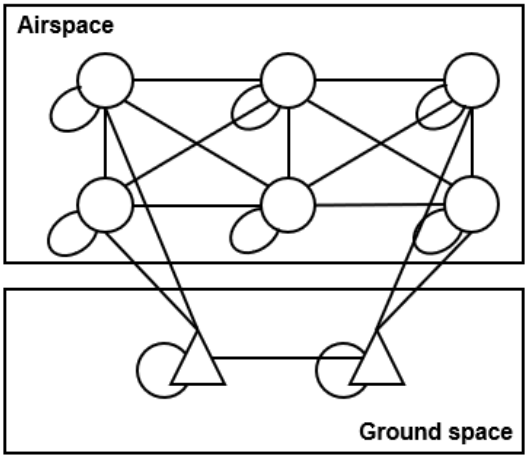

Figure 3.2. The schematic network for combined GV and UAV operation

Notice that in the network graph shown in Figure 3.2, there are arcs connecting the two workspaces. From a UAV routing perspective, these arcs turn two different workspaces into one workspace. However, from a GV routing perspective, there are still two different workspaces, and the GV is only able to access one of them. This network then has similarity in terms of properties with the two-echelon routing problem network. We can use this concept to help build our mathematical models.

In order to record both the time and battery power consumption for the UAV, together with the GV movement at each time period, we are using a time-space network method. Figure 3.3 shows the result of the transformation from the original into the time-space network.

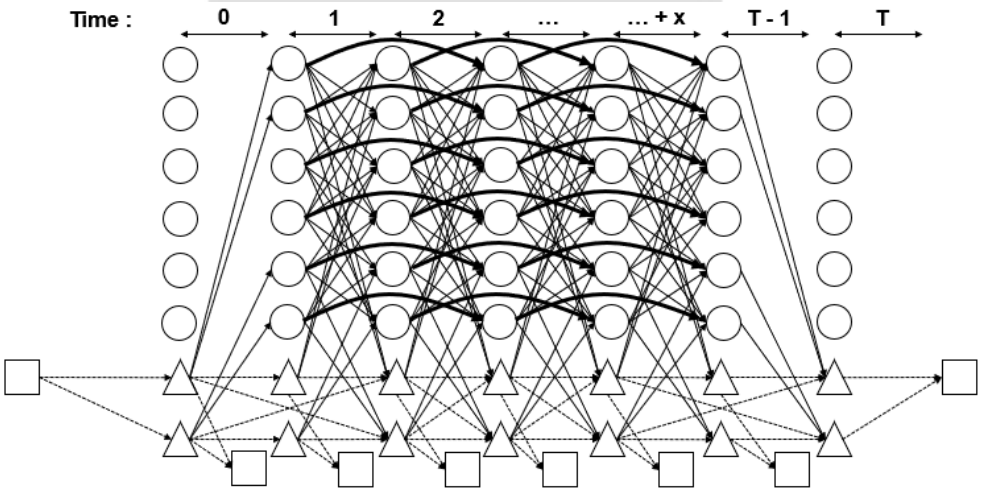

Figure 3.3. The schematic diagram for the proposed model

doi:10.6844/NCKU201901976

In our proposed model, we required both the UAV and GV to start and end their task together in the ground space. We made it possible for them to start at any node or only at several potential nodes, and even at only one node (e.g., depot). This condition also applies to choose the location to end the task. The UAV has to scan the entire airspace by serving all nodes at most once. During each movement or completion of any task, the UAV consumes some energy from the battery. When the battery is almost empty, the UAV has to go back to the ground space and meet with the GV to charge or swap the battery. This process will repeat until the UAV scan all nodes in the airspace. Figure 3.4 shows an example of the proposed method for the one GV with one UAV model with six cells and two ground nodes.

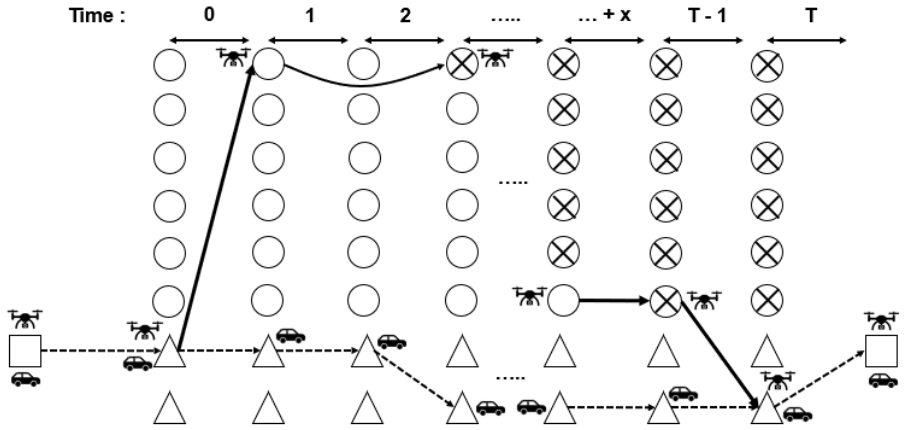

Figure 3.4. Example of the proposed method

In summary, we attempt to model a combined GV and UAV operation to cover/monitor specifically designated areas. To build our proposed model, we use several methods. To model how the UAV performs its task, we use a coverage path planning method. We adopt the two-echelon routing problem concept since we will determine the route for the GV and the UAV. By combining the coverage path planning and the two-echelon routing problem concept, we can develop the model for the combined GV and UAV operation. We also use a time-space network to track the movement of the UAV at each time point, including its energy consumption and the movement of the GV. Figure 3.5 shows how each method is related to building the proposed model.

doi:10.6844/NCKU201901976

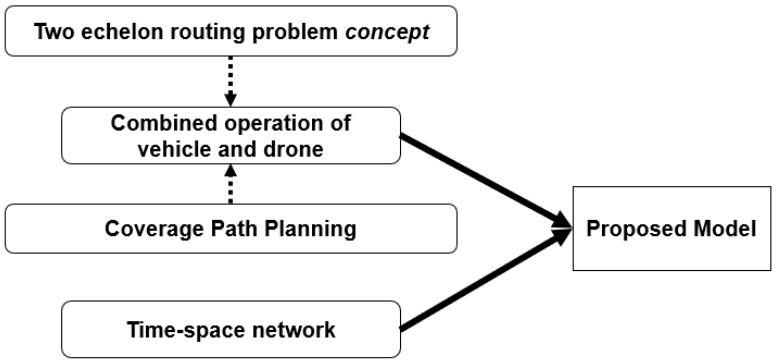

Figure 3.5. A schematic diagram of the relationships among the methods

3.2. Problem Definition

Let $G=(V,E)$ be a complete undirected graph, where V is the set of all nodes except the dummy nodes. Let $V_{ground}=\{1,2,...,n\}$ be a subset of V containing n possible rendezvous nodes in the ground space, and $V_{air}=\{1+n,2+n,...,m+n\}$ be a subset of V containing m nodes that represent the number of cells in the airspace. Each rendezvous node in the ground space represents a location in the road network where the GV can stop to launch the UAV. Let V_{start} be a subset of V_{ground} containing a possible starting node where the GV and UAV start their job and V_{end} be a subset of V_{ground} containing possible end nodes where the GV and UAV end their job.

Let A be the set of all arcs except the arc connecting to the dummy nodes. Let $A_{ground}=\{(i',j')\,|\,i',j'\in V_{ground}\}$ be a subset of A containing all arcs that connect each node in the ground space. Let $A_{air}=\{(i'',j'')\,|\,i'',j''\in V_{air}\}$ be a subset of A containing all arcs that connect each cell in the airspace. Let $A_{loop}=A^1_{loop}\cup A^2_{loop}$ be a set of self-loop arcs. Let $A^1_{loop}=\{(i',i')\,|\,i'\in V_{ground}\}$ be the self-loop arcs for the node in the ground space that allows the UAV to wait for the GV to arrive at that node and vice versa. Let $A^2_{loop}=\{(i'',i'')\,|\,i''\in V_{air}\}$ be the self-loop arcs for the node in the airspace that represent that the UAV serves the cell if passing these arcs.

Since there is a relation between the ground space and the airspace, where for each node in the ground space, there is a specific range of coverage to cover several cells in the air, then, let $R_i=\{j\,|\,d_{ij}\le\gamma, j\in V_{ground}\cup V_{air}\}$ be a set of nodes adjacent to the node

doi:10.6844/NCKU201901976

$i \in V_{ground} \cup V_{air}$ to represent this range coverage. Let γ be a range threshold for the UAV to travel that consumes half the battery level energy. Let $A_{connect} = A^1_{connect} \cup A^2_{connect}$ be a set of all arcs (i, j) that connects two workspaces. Let $A^1_{connect} = \{(i', j'') \mid j'' \in R_{i'}\}$ be the set of arcs with a direction from the node i' in the ground space to the node j'' in the airspaces, and let $A^2_{connect} = \{(i'', j') \mid j' \in R_{i''}\}$ be the set of arcs that is opposite the direction of the set $A^1_{connect}$.

In our model, we allow a condition where the task starts and ends at the depot, at some potential location, or any location in the ground space. To meet this condition, we introduce the set of starting nodes and the set of ending nodes and then connect them to the dummy nodes. Let V_{start} be a set of starting nodes, where V_{start} is a subset of V, with $A_{start} = \{(s', i') \mid i' \in V_{start}\}$ being a set of arcs that connects them to the dummy node. Let V_{end} be a set of ending nodes, where V_{end} is a subset of V, with $A_{end} = \{(i', s'') \mid i' \in V_{end}\}$ being a set of arcs that connects them to the dummy node.

Let $\bar{t}_{ij}$ be the time required to traverse from node i to node j, where $(i, j) \in A$ and is assumed to be constant. While passing through arcs $(i, j) \in A$ by itself, the UAV will consume energy from its battery. Then, let $\bar{b}_{ij}$ be the energy required to traverse from node i to node j, where $(i, j) \in A$, and B is the maximum battery energy level. $\bar{b}_{ij}$ and B are assumed to be constant. For each cell in the airspace, let $\bar{s}_i$ be a service time for visiting node i, where $i \in V_{air}$. Since for $(i'', i'') \in A^2_{loop}$ representing the UAV servicing node $i \in V_{air}$, then $\bar{s}_i$ has the same value as $\bar{t}_{i''i''}$, where $(i'', i'') \in A^2_{loop}$.

To develop our mathematical models for this problem, we make several assumptions as follows:

1. The GV moves in the ground space only, while the UAV moves at two workspaces with some restrictions while moves in the ground space.
2. The UAV serves each virtual cell (nodes) exactly once. However, the UAV can visit each node more than once.
3. The service times for all virtual cells (nodes) and the traversal time for all arcs, including the required energy to perform the service or travel are known beforehand and assumed to be constant.

doi:10.6844/NCKU201901976

4. The UAV has an energy limit, and there is no operational limit for the GV.
5. The time required to charge or swap the UAV battery is negligible.
6. There is no communication constraint between the UAV and the GV.

3.3. Mathematical Models

3.3.1. Notation

The following notations used in the mathematical model are listed below:

Sets

V	Set of all nodes. $V \in V_{air} \cup V_{ground}$
V_{ground}	Set of nodes in the ground space. $\forall i \in \{1, 2, ..., n\}$
V_{air}	Set of nodes in the airspace. $\forall i \in \{1+n, 2+n, ..., m+n\}$
A	Set of all arcs. $A \in A_{air} \cup A_{ground} \cup A_{connect} \cup A_{loop}$
A_{ground}	Set of arcs in the ground space. $A_{ground} = \{(i', j') \mid i', j' \in V_{ground}\}$
A_{air}	Set of arcs in the airspace. $A_{air} = \{(i'', j'') \mid i'', j'' \in V_{air}\}$
$A_{connect}$	Set of arcs connecting two workspaces. $A_{connect} = A^1_{connect} \cup A^2_{connect} = \{(i', j'') \mid j'' \in R_{i'}\} \cup \{(i'', j') \mid j' \in R_{i''}\}$
A_{loop}	Set of self-loop arcs. $A_{loop} = A^1_{loop} \cup A^2_{loop} = \{(i', i') \mid i' \in V_{ground}\} \cup \{(i'', i'') \mid i'' \in V_{air}\}$
R_i	Set of nodes adjacent to the node $i \in V_{ground} \cup V_{air}$, $i \in V_{ground} \cup V_{air}, R_i = \{j \mid d_{ij} \le \gamma, j \in V_{ground} \cup V_{air}\}$
T	Set of the time period. $\forall t \in T$
V_{start}	Set of potential start nodes in the ground space. $V_{start} \in V_{ground}$
V_{end}	Set of potential end nodes in the ground space. $V_{end} \in V_{ground}$
A_{start}	Set of dummy arcs for selecting the start node. $A_{start} = \{(s', i') \cup (i', s') \mid i' \in V_{start}\}$
A_{end}	Set of dummy arcs for selecting the end node. $A_{end} = \{(i', s'') \cup (s'', i') \mid i' \in V_{end}\}$
K	Set of UAVs. $k \in \{1, 2, ..., K\}$
L	Set of GVs. $l \in \{1, 2, ..., L\}$

doi:10.6844/NCKU201901976

Parameters

$\bar{t}_{ij}$ Traversal time to passing arcs $(i, j) \in A$.

$\bar{s}_i$ Time to serve the node $i \in V_{air}$, equal to a time required to pass the arcs $(i'', i'') \in A^2_{loop}$.

$\bar{b}_{ij}$ Energy consumption to pass the arcs $(i, j) \in A$.

γ Range threshold for the UAV to travel that consumes half the battery level power.

ε A small number.

B The maximum battery energy level of the UAVs.

Decision variables

x_{ij}^{kt} 1, if the UAV k at a time t starts to pass arc $(i, j) \in A$.
0, otherwise

y_{ij}^{lt} 1, if the GV l at a time t starts to pass arc $(i, j) \in A$.
0, otherwise

$\widehat{\beta}_{ij}^{kt}$ The UAV k outgoing battery level when pass arc $(i, j) \in A$ at time t.

$\breve{\beta}_{ij}^{kt}$ The UAV k incoming battery level when pass arc $(i, j) \in A$ at time t.

α_{it}^{kl} 1, if the UAV k meets with the GV l at time t on node i.
0, otherwise

λ_{ij}^{kt} 1, if the UAV k recharges the battery at time t on node i and departs to node j.
0, otherwise

ω_{ij}^{kt} 1, if the UAV k is carried by the GV from node i to node j.
0, otherwise

3.3.2. The one-GV-one-UAV model M_{1-V}^{1-D}

This model considers a situation where only one GV ($l \in \{1\}$) carrying exactly one UAV ($k \in \{1\}$) is required to cover the given area. The mathematical model M_{1-V}^{1-D} is as follows:

The objective function is to minimize the total time required to complete the task shown in Equation (3.1). Since the UAV performs the task, then it is only necessary to minimize the total time for the UAV. We can ignore the total time for the GV since the GV and the UAV will end their job together. Thus, we assign ε to ignore the total time for the GV.

doi:10.6844/NCKU201901976

$$Minimize: \sum_{i\in V}\sum_{j\in V}\sum_{t\in T} x_{ij}^{1t}\bar{t}_{ij} + \varepsilon \sum_{i'\in V_{ground}}\sum_{j'\in V_{ground}}\sum_{t\in T} y_{i'j'}^{1t}\bar{t}_{i'j'} \tag{3.1}$$

In summary, the constraints for this model can be categorized into four parts. The first part is selecting the start node, for equations (3.2) – (3.4) are used. The second part is the movement of the UAV and GV, equations (3.5) – (3.11) comprise this part. The third part is battery power consumption, as shown in equations (3.12) – (3.25). The last part is selecting the end node, as shown in equations (3.26) – (3.29). The details of each constraint are explained as follows:

The constraints shown in Equation (3.2) define how the GV will select its start node. The GV will select only one node i' among the candidates of potential start nodes in the ground space.

$$\sum_{(s',i')\in A_{start}} y_{s'i'}^{10} = 1 \tag{3.2}$$

The constraints shown in Equation (3.3) define how the UAV will select its start node in the ground space.

$$\sum_{(s',i')\in A_{start}} x_{s'i'}^{10} = 1 \tag{3.3}$$

In our model, we require the UAV to start together with the GV. This condition is defined by the constraints shown in Equation (3.4).

$$x_{s'i'}^{10} - y_{s'i'}^{10} = 0 \qquad \forall (s',i') \in A_{start} \tag{3.4}$$

The constraints shown in equations (3.5) – (3.6) define the flow balance for the GV. Equation (3.5) defines the flow balance at a node i' when $t = 0$. At this moment, the dummy starts arcs are included in the incoming arcs, but the dummy end arcs are not included in the outgoing arcs since we know clearly that the task will not be completed at $t = 0$. As for Equation (3.6), we include the dummy end arcs as outgoing arcs since we do not know exactly when the task will be completed.

$$\begin{aligned}&\sum_{(i',j')\in A_{ground}} y_{i'j'}^{1t} + \sum_{(i',i')\in A_{loop}^{1}} y_{i'i'}^{1t} - \sum_{(s',i')\in A_{start}} y_{s'i'}^{10} - \sum_{(h',i')\in A_{ground}} y_{h'i'}^{1(t-\bar{t}_{h'i'})} \\ &- \sum_{(i',i')\in A_{loop}^{1}} y_{i'i'}^{1(t-\bar{t}_{i'i'})} = 0 \qquad \forall i' \in V_{ground}, t = 0\end{aligned} \tag{3.5}$$

$$\begin{aligned}&\sum_{(i',j')\in A_{ground}} y_{i'j'}^{1t} + \sum_{(i',i')\in A_{loop}^{1}} y_{i'i'}^{1t} + \sum_{(i,s'')\in A_{end}} y_{i's''}^{1t} - \sum_{(h',i')\in A_{ground}} y_{h'i'}^{1(t-\bar{t}_{h'i'})} \\ &- \sum_{(i',i')\in A_{loop}^{1}} y_{i'i'}^{1(t-\bar{t}_{i'i'})} = 0 \qquad \forall i' \in V_{ground}, t \neq 0\end{aligned} \tag{3.6}$$

doi:10.6844/NCKU201901976

The constraints shown in equations (3.7) – (3.8) define the flow balance for the UAV in the ground space. The condition is the same as that for the flow balance for the GV. The difference is that for both the incoming and outgoing arcs, the connecting arcs are included. The air to ground arcs are in the incoming arcs and the ground to air arcs are in the outgoing arcs.

$$\sum_{(i',j')\in A_{ground}} x_{i'j'}^{1t} + \sum_{(i',i')\in A_{loop}^{1}} x_{i'i'}^{1t} + \sum_{(i',j'')\in A_{connect}^{1}} x_{i'j''}^{1t} - \sum_{(s',i)\in A_{start}} x_{s'i'}^{10} - \sum_{(h',i')\in A_{ground}} x_{h'i'}^{1(t-\bar{t}_{h'i'})} - \sum_{(i',i')\in A_{loop}^{1}} x_{i'i'}^{1(t-\bar{t}_{i'i'})} - \sum_{(h'',i')\in A_{connect}^{2}} x_{h''i'}^{1(t-\bar{t}_{h''i'})} = 0 \qquad \forall i' \in V_{ground}, t = 0 \tag{3.7}$$

$$\sum_{(i',j')\in A_{ground}} x_{i'j'}^{1t} + \sum_{(i',i')\in A_{loop}^{1}} x_{i'i'}^{1t} + \sum_{(i',j'')\in A_{connect}^{1}} x_{i'j''}^{1t} + \sum_{(i,s'')\in A_{end}} x_{i's''}^{10} - \sum_{(h',i')\in A_{ground}} x_{h'i'}^{1(t-\bar{t}_{h'i'})} - \sum_{(i',i')\in A_{loop}^{1}} x_{i'i'}^{1(t-\bar{t}_{i'i'})} - \sum_{(h'',i')\in A_{connect}^{2}} x_{h''i'}^{1(t-\bar{t}_{h''i'})} = 0 \qquad \forall i' \in V_{ground}, t \neq 0 \tag{3.8}$$

The constraints shown in Equation (3.9) define the flow balance for the UAV in the airspace.

$$\sum_{(i'',j'')\in A_{air}} x_{i''j''}^{1t} + \sum_{(i'',i'')\in A_{loop}^{2}} x_{i''i''}^{1t} + \sum_{(i'',j')\in A_{connect}^{2}} x_{i''j'}^{1t} - \sum_{(h'',i'')\in A_{air}} x_{h''i''}^{1(t-\bar{t}_{h''i''})} - \sum_{(i'',i'')\in A_{loop}^{2}} x_{i''i''}^{1(t-\bar{t}_{i''i''})} - \sum_{(h',i'')\in A_{connect}^{1}} x_{h'i''}^{1(t-\bar{t}_{h'i''})} = 0 \qquad \forall i'' \in V_{air}, t \in T \tag{3.9}$$

The constraints shown in Equation (3.10) indicate that each cell in the airspace will be served exactly once by the UAV.

$$\sum_{t\in T} \sum_{(i'',i'')\in A_{loop}^{2}} x_{i''i''}^{1t} = 1 \qquad \forall i'' \in V_{air} \tag{3.10}$$

The UAV has some restrictions when moving in the ground space. The restriction is when the UAV wants to use the ground to ground arcs, it must be carried by the GV. Thus, Equation (3.11) defines this requirement.

$$x_{i'j'}^{1t} \leq y_{i'j'}^{1t} \qquad \forall t \in T, (i', j') \in A_{ground} \tag{3.11}$$

The constraints shown in equations (3.12) – (3.13) define the condition when the UAV and the GV at time period t meet in node i'.

$$\alpha_{i't}^{11} \leq \frac{\sum_{(i',j')\in A_{ground}} x_{i'j'}^{1t} + \sum_{(i',i')\in A_{loop}^{1}} x_{i'i'}^{1t} + \sum_{(i',j'')\in A_{connect}^{1}} x_{i'j''}^{1t} + \sum_{(i',j')\in A_{ground}} y_{i'j'}^{1t} + \sum_{(i',i')\in A_{loop}^{1}} y_{i'i'}^{1t}}{2} \qquad \forall i' \in V_{ground}, t \neq 0 \tag{3.12}$$

doi:10.6844/NCKU201901976

$$\alpha_{i't}^{11} \geq \sum_{(i',j')\in A_{ground}} x_{i'j'}^{1t} + \sum_{(i',i')\in A_{loop}^{1}} x_{i'i'}^{1t} + \sum_{(i',j'')\in A_{connect}^{1}} x_{i'j''}^{1t} + \sum_{(i',j')\in A_{ground}} y_{i'j'}^{1t} + \sum_{(i',i')\in A_{loop}^{1}} y_{i'i'}^{1t} - 1 \quad \forall i' \in V_{ground}, t \neq 0 \tag{3.13}$$

When the UAV meet the GV at a node i', then the UAV assumed will recharge or swap its battery. Equations (3.14) – (3.15) define whether or not a battery recharge is performed.

$$\lambda_{i'j}^{1t} \leq (x_{i'j}^{1t} + \alpha_{i't}^{11})/2 \qquad \forall (i', j) \in A_{ground} \cup A_{loop}^{1} \cup A_{connect}^{1}, t \neq 0 \tag{3.14}$$

$$\lambda_{i'j}^{1t} \geq x_{i'j}^{1t} + \alpha_{i't}^{11} - 1 \qquad \forall (i', j) \in A_{ground} \cup A_{loop}^{1} \cup A_{connect}^{1}, t \neq 0 \tag{3.15}$$

The constraints shown in equations (3.16) – (3.17) define the condition when the UAV carried by the GV in the ground space.

$$\omega_{i'j'}^{1t} \leq (x_{i'j'}^{1t} + y_{i'j'}^{1t})/2 \qquad \forall t \in T, (i', j') \in A_{ground} \cup A_{loop}^{1} \tag{3.16}$$

$$\omega_{i'j'}^{1t} \geq x_{i'j'}^{1t} + y_{i'j'}^{1t} - 1 \qquad \forall t \in T, (i', j') \in A_{ground} \cup A_{loop}^{1} \tag{3.17}$$

The constraints shown in equations (3.18) – (3.19) define the upper limit for outgoing and incoming battery levels at each arc. When the arc is selected, the maximum battery level is B.

$$\hat{\beta}_{ij}^{1t} \leq B x_{ij}^{1t} \qquad \forall t \in T, (i, j) \in A \tag{3.18}$$

$$\breve{\beta}_{ij}^{1t} \leq B x_{ij}^{1t} \qquad \forall t \in T, (i, j) \in A \tag{3.19}$$

Equation (3.20) indicates that the initial battery at node i' is the maximum battery level.

$$\hat{\beta}_{ij}^{10} - B x_{ij}^{10} = 0 \qquad \forall (i, j) \in A_{ground} \cup A_{loop}^{1} \cup A_{connect}^{1} \tag{3.20}$$

The constraints shown in equations (3.21) – (3.22) define the relationships among the outgoing and incoming battery levels at each arc. Equation (3.22) expresses that at the airspace and the connecting arcs, the incoming battery level is equal to the outgoing battery level reduced by the required energy to pass the arc. When in the ground space, the UAV will consume some battery power only if the UAV moves alone. This condition is stated in Equation (3.21).

$$\hat{\beta}_{ij}^{1t} - \bar{b}_{ij}(x_{ij}^{1t} - \omega_{ij}^{1t}) = \breve{\beta}_{ij}^{1t} \qquad \forall t \in T, (i, j) \in A_{ground} \cup A_{loop}^{1} \tag{3.21}$$

$$\hat{\beta}_{ij}^{1t} - \bar{b}_{ij} x_{ij}^{1t} = \breve{\beta}_{ij}^{1t} \qquad \forall t \in T, (i, j) \in A_{air} \cup A_{loop}^{2} \cup A_{connect} \tag{3.22}$$

The constraints shown in equations (3.23) – (3.24) define the battery flow balance when the UAV is located in the ground space. If the UAV at node i' does not meet the GV, then the outgoing battery level at the selected outgoing arcs in time *t* is equal to the previous incoming battery level from the incoming arcs. If the UAV meets the GV, then the outgoing

doi:10.6844/NCKU201901976

battery level at the selected outgoing arcs is equal to the maximum battery level since a recharge or battery swap is performed.

$$\sum_{(h',i')\in A_{ground}} \breve{\beta}_{h'i'}^{1(t-\bar{t}_{h'i'})} + \sum_{(i',i')\in A_{loop}^{1}} \breve{\beta}_{i'i'}^{1(t-\bar{t}_{i'i'})} + \sum_{(h'',i')\in A_{connect}^{2}} \breve{\beta}_{h''i'}^{1(t-\bar{t}_{h''i'})} + B\sum_{(i',j')\in A_{ground}} \lambda_{i'j'}^{1t} + B\sum_{(i',i')\in A_{loop}^{1}} \lambda_{i'i'}^{1t}$$
$$+B\sum_{(i',j'')\in A_{connect}^{1}} \lambda_{i'j''}^{1t} \geq \sum_{(i',j')\in A_{ground}} \widehat{\beta}_{i'j'}^{1t} + \sum_{(i',i')\in A_{loop}^{1}} \widehat{\beta}_{i'i'}^{1t} + \sum_{(i',j'')\in A_{connect}^{1}} \widehat{\beta}_{i'j''}^{1t} \qquad \forall t\neq 0, i'\in V_{ground} \tag{3.23}$$

$$\widehat{\beta}_{ij}^{1t} \geq B\lambda_{ij}^{1t} \qquad \forall t\neq 0,(i,j)\in A_{ground}\cup A_{loop}^{1}\cup A_{connect}^{1} \tag{3.24}$$

The constraints shown in Equation (3.25) define the battery flow balance when the UAV is located in the airspace. When the UAV is going to or already in the airspace, it will always consume some energy to move, so the outgoing battery level for the selected arcs at time t is equal to the previous incoming battery level from the incoming arcs.

$$\sum_{(i'',j'')\in A_{air}} \widehat{\beta}_{i''j''}^{1t} + \sum_{(i'',i'')\in A_{loop}^{2}} \widehat{\beta}_{i''i''}^{1t} + \sum_{(i'',j')\in A_{connect}^{2}} \widehat{\beta}_{i''j'}^{1t} - \sum_{(h'',i'')\in A_{air}} \breve{\beta}_{h''i''}^{1(t-\bar{t}_{h''i''})} - \sum_{(i'',i'')\in A_{loop}^{2}} \breve{\beta}_{i''i''}^{1(t-\bar{t}_{i''i''})}$$
$$-\sum_{(h',i'')\in A_{connect}^{1}} \breve{\beta}_{h'i''}^{1(t-\bar{t}_{h'i''})} = 0 \qquad \forall t\neq 0, i''\in V_{air} \tag{3.25}$$

The constraints shown in Equation (3.26) indicate that the GV will select the dummy sink node to end the job.

$$\sum_{t\in T, t\neq 0}\sum_{(i',s'')\in A_{end}} y_{i's''}^{1t} = 1 \tag{3.26}$$

The constraints shown in Equation (3.27) indicate that the UAV will select the dummy sink node to end the job.

$$\sum_{t\in T, t\neq 0}\sum_{(i',s'')\in A_{end}} x_{i's''}^{1t} = 1 \tag{3.27}$$

To end the task, the UAV must land on the GV. Therefore, the UAV will choose the same node as the GV. Equation (3.28) shows this condition.

$$x_{i's''}^{1t} - y_{i's''}^{1t} = 0 \qquad \forall t\neq 0,(i',s'')\in A_{end} \tag{3.28}$$

The UAV will end its coverage/monitoring task only when all cells in the airspace are being served. This requirement is described in Equation (3.29).

$$m\sum_{(i',s'')\in A_{end}} x_{i's''}^{1t} \leq \sum_{\theta=1}^{t}\sum_{(h'',h'')\in A_{loop}^{2}} x_{h''h''}^{1\theta} \qquad \forall t\in T, i'\in V_{ground} \tag{3.29}$$

doi:10.6844/NCKU201901976

3.3.3. The one-GV-multiple-UAV model M_{1-V}^{k-D}

This model is an extension of the previous model. Instead of carrying only one UAV, now the GV is carrying multiple UAVs ($k\epsilon\{1,\dots,K\}$). This model is developed since in the real world, deploying multiple UAVs instead one is more reasonable for monitoring tasks over a wider area or when the task must be done faster. The mathematical model for M_{1-V}^{k-D} can be described as follows:

The objective functions are the same as those in the previous model, that is, to minimize the total time required to complete the task. However, since the number of UAVs now is increased, then the objective function becomes a mini-max function, with the goal to minimize the maximum time of k- UAV's operating time. The objective function in (3.1) changes into Equation (3.30).

$$Minimize: \underset{k\in K}{Max}\left(\sum_{i\in V}\sum_{j\in V}\sum_{t\in T}x_{ij}^{kt}\bar{t}_{ij}\right)+\varepsilon\sum_{i'\in V_{ground}}\sum_{j'\in V_{ground}}\sum_{t\in T}y_{i'j'}^{lt}\bar{t}_{i'j'} \tag{3.30}$$

Since Equation (3.30) is in minimax form, it must be transformed into a linear programming form. The complete mathematical model for M_{1-V}^{k-D} is:

$$Minimize: w+\varepsilon\sum_{i'\in V_{ground}}\sum_{j'\in V_{ground}}\sum_{t\in T}y_{i'j'}^{lt}\bar{t}_{i'j'} \tag{3.31}$$

All assumptions in M_{1-V}^{1-D} also apply to M_{1-V}^{k-D}. As for the constraints, basically it almost identical with the M_{1-V}^{1-D} model, which for M_{1-V}^{k-D} model we have *k* numbers of UAVs instead only one. There is an additional constraint shown in Equation (3.32) and an additional assumption, that each UAVs fleet has the same type. By this assumption, the battery power consumption rate and traversal time for all UAVs are the same. The following is a list of constraints for this model:

Equation (3.32) is related to the transformation of the objective function from minimax to a linear programming form.

$$\sum_{i\in V}\sum_{j\in V}\sum_{t\in T}x_{ij}^{kt}\bar{t}_{ij}\le w \qquad \forall k\in K \tag{3.32}$$

$$\sum_{(s',i')\in A_{start}}y_{s'i'}^{10}=1 \tag{3.33}$$

$$\sum_{(s',i')\in A_{start}}x_{s'i'}^{k0}=1 \qquad \forall k \tag{3.34}$$

$$x_{s'i'}^{k0}-y_{s'i'}^{10}=0 \qquad \forall k,(s',i')\in A_{start} \tag{3.35}$$

doi:10.6844/NCKU201901976

$$\sum_{(i',j')\in A_{ground}} y_{i'j'}^{1t} + \sum_{(i',i')\in A_{loop}^1} y_{i'i'}^{1t} - \sum_{(s',i')\in A_{start}} y_{s'i'}^{10} - \sum_{(h',i')\in A_{ground}} y_{h'i'}^{1(t-\bar{t}_{h'i'})} - \sum_{(i',i')\in A_{loop}^1} y_{i'i'}^{1(t-\bar{t}_{i'i'})} = 0 \qquad \forall i' \in V_{ground}, t = 0 \tag{3.36}$$

$$\sum_{(i',j')\in A_{ground}} y_{i'j'}^{1t} + \sum_{(i',i')\in A_{loop}^1} y_{i'i'}^{1t} + \sum_{(i,s'')\in A_{end}} y_{i's''}^{1t} - \sum_{(h',i')\in A_{ground}} y_{h'i'}^{1(t-\bar{t}_{h'i'})} - \sum_{(i',i')\in A_{loop}^1} y_{i'i'}^{1(t-\bar{t}_{i'i'})} = 0 \qquad \forall i' \in V_{ground}, t \neq 0 \tag{3.37}$$

$$\sum_{(i',j')\in A_{ground}} x_{i'j'}^{kt} + \sum_{(i',i')\in A_{loop}^1} x_{i'i'}^{kt} + \sum_{(i',j'')\in A_{connect}^1} x_{i'j''}^{kt} - \sum_{(s',i)\in A_{start}} x_{s'i'}^{k0} - \sum_{(h',i')\in A_{ground}} x_{h'i'}^{k(t-\bar{t}_{h'i'})} - \sum_{(i',i')\in A_{loop}^1} x_{i'i'}^{k(t-\bar{t}_{i'i'})} - \sum_{(h'',i')\in A_{connect}^2} x_{h''i'}^{k(t-\bar{t}_{h''i'})} = 0 \qquad \forall k, i' \in V_{ground}, t = 0 \tag{3.38}$$

$$\sum_{(i',j')\in A_{ground}} x_{i'j'}^{kt} + \sum_{(i',j')\in A_{ground}} x_{i'j'}^{kt} + \sum_{(i',i')\in A_{loop}^1} x_{i'i'}^{kt} + \sum_{(i',j'')\in A_{connect}^1} x_{i'j''}^{kt} + \sum_{(i,s'')\in A_{end}} x_{i's''}^{k0} - \sum_{(h',i')\in A_{ground}} x_{h'i'}^{k(t-\bar{t}_{h'i'})} - \sum_{(i',i')\in A_{loop}^1} x_{i'i'}^{k(t-\bar{t}_{i'i'})} - \sum_{(h'',i')\in A_{connect}^2} x_{h''i'}^{k(t-\bar{t}_{h''i'})} = 0 \qquad \forall k, i' \in V_{ground}, t \neq 0 \tag{3.39}$$

$$\sum_{(i'',j'')\in A_{air}} x_{i''j''}^{kt} + \sum_{(i'',i'')\in A_{loop}^2} x_{i''i''}^{kt} + \sum_{(i'',j')\in A_{connect}^2} x_{i''j'}^{kt} - \sum_{(h'',i'')\in A_{air}} x_{h''i''}^{k(t-\bar{t}_{h''i''})} - \sum_{(i'',i'')\in A_{loop}^2} x_{i''i''}^{k(t-\bar{t}_{i''i''})} - \sum_{(h',i'')\in A_{connect}^1} x_{h'i''}^{k(t-\bar{t}_{h'i''})} = 0 \qquad \forall k, i'' \in V_{air}, t \in T \tag{3.40}$$

$$\sum_{k\in K}\sum_{t\in T}\sum_{(i'',i'')\in A_{loop}^2} x_{i''i''}^{1t} = 1 \qquad \forall i'' \in V_{air} \tag{3.41}$$

$$x_{i'j'}^{kt} \leq y_{i'j'}^{1t} \qquad \forall k, t \in T, (i', j') \in A_{ground} \tag{3.42}$$

$$\alpha_{i't}^{k1} \leq \frac{\sum_{(i',j')\in A_{ground}} x_{i'j'}^{kt} + \sum_{(i',i')\in A_{loop}^1} x_{i'i'}^{kt} + \sum_{(i',j'')\in A_{connect}^1} x_{i'j''}^{kt} + \sum_{(i',j')\in A_{ground}} y_{i'j'}^{1t} + \sum_{(i',i')\in A_{loop}^1} y_{i'i'}^{1t}}{2} \qquad \forall k, i' \in V_{ground}, t \neq 0 \tag{3.43}$$

$$\alpha_{i't}^{k1} \geq \sum_{(i',j')\in A_{ground}} x_{i'j'}^{kt} + \sum_{(i',i')\in A_{loop}^1} x_{i'i'}^{kt} + \sum_{(i',j'')\in A_{connect}^1} x_{i'j''}^{kt} + \sum_{(i',j')\in A_{ground}} y_{i'j'}^{1t} + \sum_{(i',i')\in A_{loop}^1} y_{i'i'}^{1t} - 1 \quad \forall k, i' \in V_{ground}, t \neq 0$$

$$\lambda_{i'j}^{kt} \leq (x_{i'j}^{kt} + \alpha_{i't}^{k1})/2 \qquad \forall k, (i', j) \in A_{ground} \cup A_{loop}^1 \cup A_{connect}^1, t \neq 0 \tag{3.44}$$

$$\lambda_{i'j}^{kt} \geq x_{i'j}^{kt} + \alpha_{i't}^{k1} - 1 \qquad \forall k, (i', j) \in A_{ground} \cup A_{loop}^1 \cup A_{connect}^1, t \neq 0 \tag{3.45}$$

$$\omega_{i'j'}^{kt} \leq (x_{i'j'}^{kt} + y_{i'j'}^{1t})/2 \qquad \forall k, t \in T, (i', j') \in A_{ground} \cup A_{loop}^1 \tag{3.46}$$

$$\omega_{i'j'}^{kt} \geq x_{i'j'}^{kt} + y_{i'j'}^{1t} - 1 \qquad \forall k, t \in T, (i', j') \in A_{ground} \cup A_{loop}^1 \tag{3.47}$$

$$\widehat{\beta}_{ij}^{kt} \leq B x_{ij}^{kt} \qquad \forall k, t \in T, (i, j) \in A \tag{3.48}$$

doi:10.6844/NCKU201901976

$$\breve{\beta}_{ij}^{kt} \le Bx_{ij}^{kt} \qquad \forall k,t \in T,(i,j) \in A \tag{3.49}$$

$$\hat{\beta}_{ij}^{k0} - Bx_{ij}^{k0} = 0 \qquad \forall k,(i,j) \in A_{ground} \cup A_{loop}^{1} \cup A_{connect}^{1} \tag{3.50}$$

$$\hat{\beta}_{ij}^{kt} - \bar{b}_{ij}(x_{ij}^{kt} - \omega_{ij}^{kt}) = \breve{\beta}_{ij}^{kt} \qquad \forall k,t \in T,(i,j) \in A_{ground} \cup A_{loop}^{1} \tag{3.51}$$

$$\hat{\beta}_{ij}^{kt} - \bar{b}_{ij}x_{ij}^{kt} = \breve{\beta}_{ij}^{kt} \qquad \forall k,t \in T,(i,j) \in A_{air} \cup A_{loop}^{2} \cup A_{connect} \tag{3.52}$$

$$\begin{aligned} &\sum_{(h',i') \in A_{ground}} \breve{\beta}_{h'i'}^{k(t-\bar{t}_{h'i'})} + \sum_{(i',i') \in A_{loop}^{1}} \breve{\beta}_{i'i'}^{k(t-\bar{t}_{i'i'})} + \sum_{(h'',i') \in A_{connect}^{2}} \breve{\beta}_{h''i'}^{k(t-\bar{t}_{h''i'})} + B\sum_{(i',j') \in A_{ground}} \lambda_{i'j'}^{kt} + B\sum_{(i',i') \in A_{loop}^{1}} \lambda_{i'i'}^{kt} \\ &+B\sum_{(i',j'') \in A_{connect}^{1}} \lambda_{i'j''}^{kt} \ge \sum_{(i',j') \in A_{ground}} \hat{\beta}_{i'j'}^{kt} + \sum_{(i',i') \in A_{loop}^{1}} \hat{\beta}_{i'i'}^{kt} + \sum_{(i',j'') \in A_{connect}^{1}} \hat{\beta}_{i'j''}^{kt} \qquad \forall k,t \ne 0,i' \in V_{ground} \end{aligned} \tag{3.53}$$

$$\hat{\beta}_{ij}^{kt} \ge B\lambda_{ij}^{kt} \qquad \forall k,t \ne 0,(i,j) \in A_{ground} \cup A_{loop}^{1} \cup A_{connect}^{1} \tag{3.54}$$

$$\begin{aligned} &\sum_{(i'',j'') \in A_{air}} \hat{\beta}_{i''j''}^{kt} + \sum_{(i'',i'') \in A_{loop}^{2}} \hat{\beta}_{i''i''}^{kt} + \sum_{(i'',j') \in A_{connect}^{2}} \hat{\beta}_{i''j'}^{kt} - \sum_{(h'',i'') \in A_{air}} \breve{\beta}_{h''i''}^{k(t-\bar{t}_{h''i''})} - \sum_{(i'',i'') \in A_{loop}^{2}} \breve{\beta}_{i''i''}^{k(t-\bar{t}_{i''i''})} \\ &- \sum_{(h',i'') \in A_{connect}^{1}} \breve{\beta}_{h'i''}^{k(t-\bar{t}_{h'i''})} = 0 \qquad \forall k,t \ne 0,i'' \in V_{air} \end{aligned} \tag{3.55}$$

$$\sum_{t \in T,t \ne 0} \sum_{(i',s'') \in A_{end}} y_{i's''}^{1t} = 1 \tag{3.56}$$

$$\sum_{t \in T,t \ne 0} \sum_{(i',s'') \in A_{end}} x_{i's''}^{kt} = 1 \qquad \forall k \tag{3.57}$$

$$x_{i's''}^{kt} - y_{i's''}^{1t} = 0 \qquad \forall k,t \ne 0,(i',s'') \in A_{end} \tag{3.58}$$

$$m\sum_{(i',s'') \in A_{end}} x_{i's''}^{kt} \le \sum_{\theta=1}^{t} \sum_{(h'',h'') \in A_{loop}^{2}} x_{h''h''}^{k\theta} \qquad \forall k,t \in T,i' \in V_{ground} \tag{3.59}$$

Equations (3.33) – (3.35) related to selecting the start node. These equations have the same function as equations (3.2) – (3.4). Equations (3.36) – (3.42) are similar to equation (3.5) – (3.11), to define the movement of the UAV and GV. Equations (3.43) – (3.55) have the same purpose as equation (3.12) – (3.25) to define the battery power consumption for the UAVs. Last parts are equations (3.56) – (3.59) that related to selecting the end node, which have the same function as equations (3.26) – (3.29).

doi:10.6844/NCKU201901976

Chapter 4 Heuristic Algorithms for Planning the Joint UAV and GV Paths

Mathematical models typically can be used to obtain an optimal solution. However, they require substantial computational time, especially when the problem size is large. Our mathematical model also experiences this problem. In this chapter, we explain our proposed algorithm for the one-GV-one-UAV model.

4.1. Greedy Algorithm for the one-GV-one-UAV case

Our algorithm contains two parts: (1) finding the UAV path, and (2) finding the GV path. The idea of this algorithm is based on the route first split-second concept proposed by Luo et al. (2017). However, Luo et al.'s (2017) and our algorithm are different. The algorithm proposed by Luo et al. (2017) uses two steps to find the UAV route. At the "UAV routing" step, it finds the initial route starting from the depot to the destination by ignoring the UAV limitation. Then, based on the initial route, the "splitting step" is performed. In our algorithm, the process of finding the possible "feasible" route already takes the UAV limitation into consideration. Thus, the finding and splitting process will be executed simultaneously. An illustration of the process to find the path for the UAV and the GV is shown in Figure 4.1.

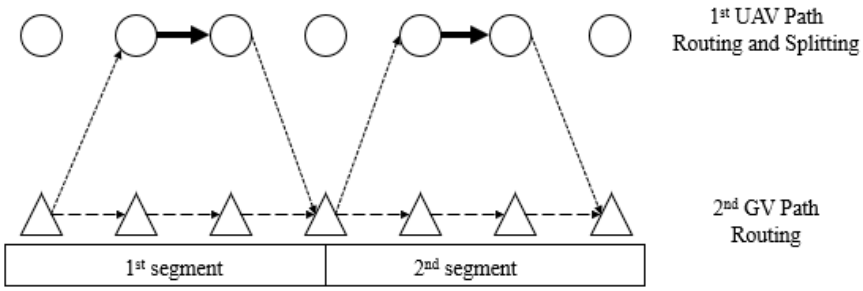

Figure 4.1. Simplified illustration of the finding path process

We adopt the so-called Depth First Search (DFS) method to find the "feasible" path for both the UAV and the GV. First, we perform Procedure 1 to seek the UAV path by ignoring the GV. At this stage, we assume that the GV is always able to catch up with the UAV and perform a battery recharge or swap. As shown in Figure 4.1, while one feasible path for the UAV is found, it may have several segments since UAVs have a battery limitation. Thus,

doi:10.6844/NCKU201901976

we perform Procedure 2 to check the GV path at each segment. If the GV path has been found for each segment, then the algorithm will return the UAV path along with its battery condition and the GV path as a result, and the algorithm will be terminated. However, if in one segment, the GV path is not found, then the current UAV path is not considered valid, so the algorithm continues to seek other possible path given the determined T and B values. If all feasible moves in the complete time-space networks have been visited, but there is no feasible path found, then the algorithm is terminated. The summary of our algorithm is drawn into the flowchart shown in Figure 4.2.

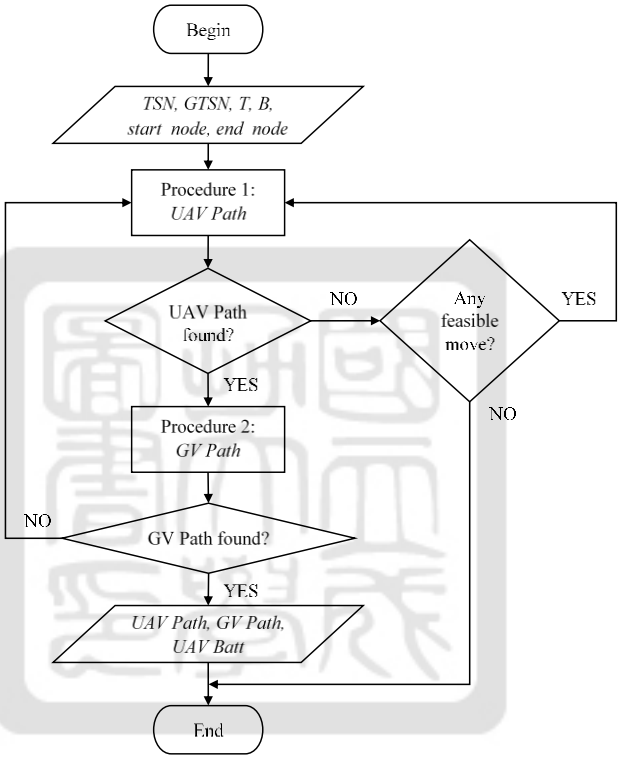

Figure 4.2. Flowchart of the proposed algorithm

As shown in Figure 4.2, our algorithm requires some information. First, it requires the complete time-space network that contains both the airspace and the ground space network, as defined by *TSN*. The *GTSN* is defined for the time-space network for the ground space network only. *T* is defined as the maximum operation time. *B* is defined as the maximum UAV battery level. The *start_node* refers to the location for both the UAV and GV to start their task, and the *end_node* refers to the location at which both the UAV and GV end their task.

doi:10.6844/NCKU201901976

4.1.1. UAV path planning heuristic

The first procedure in this algorithm is used to find a feasible path for the UAV. We call this *Procedure 1: UAV Path*, where *TSN, T, B, start_node, end_node,* and *m* as the number of virtual cells comprise the information used in this procedure.

For initialization, we set the $Node_{current} = (t, start\_node)$ where initial $t = 0$, $Batt_{current} = B$ and $scan = \varnothing$ where *scan* is the list to record node $i'' \in V_{air}$ if they are being served. If all virtual cells are being served, then the total element in *scan* must be equal to *m*.

Table 4.1. Procedure 1: UAV path pseudocode

Algorithm for Procedure 1: UAV path
Data: *TSN, T, B, start_node, end_node, m*
Initialization: $Node_{current} = (t, start\_node)$, $Batt_{current} = B$, $scan = \varnothing$, and $segment = \varnothing$
1 : **While** $scan \neq m$ **do:**
2 : Search path to finish task, update $Node_{current}$, $Batt_{current}$, $scan$ when select next node
3 : **End While**
4 : **While** $scan = m$ do:
5 : retrieve *segment*
6 : **for** ([*t, gv_start*], [*t', gv_end*]) in *segment*:
7 : Call Procedure 2: GV path
8 : **If** *GV Path* is valid:
9 : return current *UAV Path, UAV Batt, GV Path*
10 : break
11 : **Else**:
12 : delete current *UAV Path, UAV Batt, scan*
13 : **if** *UAV Path* is valid:
14 : break
15 : **End While**
Result: *UAV Path, UAV Batt, GV Path*.

As shown in Table 4.1, Procedure 1: UAV path has two conditions. The first condition is the total element in $scan \neq m$. When this condition occurs, the algorithm will seek a feasible path where all of the virtual cells will be served and make $scan = m$. At this point,

doi:10.6844/NCKU201901976

there are two possibilities that may occur. The $Node_{current} = (t, start\_node)$ with *start_node* in the ground space and *start_node* in the airspace. Since we use the time-space network, the algorithm may do many unnecessary checks for each node at each time *t*. To reduce these unnecessary checks, we define several conditions.

When the $Node_{current} = (t, start\_node)$ with *start_node* located in the airspace, we select the adjacent node only if the remaining battery at that node is not negative. Second, we do not select the adjacent node located in the airspace if that node is already being served at any time *t*. Third, we set a specific battery limit so that the UAV will go to an adjacent node located in the ground space only if this battery's limit is reached. For example, the UAV selects the adjacent node only if the battery condition at the $Node_{current}$ or the remaining battery at the adjacent node is less than or equal to 50% of the *B* value.

When the $Node_{current} = (t, start\_node)$ with *start_node* located in the ground space, we select the adjacent node located in the airspace that is not being served at any time *t*, or we select the adjacent node located in the ground space if all virtual cells within the $Node_{current}$ range threshold are already being served.

These conditions may cause the result of the algorithm to be inferior to the proposed model since these conditions limit the UAV movement options. However, these conditions may reduce the time required for the algorithm process. $Node_{current}$, $Batt_{current}$, and $scan$ will be updated from the current node to the adjacent node when selecting the adjacent node as the next node.

The second condition is when the total element in $scan = m$, which we will call the Procedure 2: GV path to validate the current path when the $Node_{current} = (t, sink)$. If the $Node_{current} \neq (t, sink)$, then the algorithm will seek a feasible path to reach the *sink*. When the $Node_{current} = (t, sink)$, then basically we already have one feasible *UAV Path*. Since the *UAV Path* may contain several segments, then we can retrieve *segment*. The segment is a list of nodes that exist in the UAV path, which are only located in the ground space. We use *segment* as input for the Procedure 2: GV path. The *GV Path* is valid when for each segment there exists a path for the GV, meaning that all paths in each segment are connected. When the *GV Path* is valid, it will return *UAV Path*, *UAV Batt*, and *GV Path* as the results and make the *UAV Path* valid. Since the *UAV Path* is valid, the algorithm will be terminated.

doi:10.6844/NCKU201901976

4.1.2. GV path planning heuristic

When one feasible UAV Path is found, then the Procedure 2: GV path will be called. The purpose of this procedure is to validate the *UAV Path* by checking the *GV Path* at each segment, where, given the specific (*t, start_node*) and (*t', end_node*), the GV is able to find the path to reach the *end_node* at time *t'* in each segment. This procedure is necessary to ensure that the assumption used in Procedure 1: UAV path is valid. *GTSN*, (*t, gv_start*) and (*t', gv_end*) comprise the information used in this procedure.

For initialization, we set the $Node_{current} = (t, gv\_start)$ and $Node_{end} = (t', gv\_end)$. If the first segment is completed, then the procedure will continue for the next segment until all segments have been checked. If there are any GVs that cannot reach the $Node_{end} = (t', gv\_end)$ at one segment, then the entire procedure has to be stopped in order to search for another feasible *UAV Path*, or the algorithm has to be re-run from the very beginning with a larger *T* if the algorithm cannot find any path at the current *T*. Table 4.2 shows the procedure to perform the algorithm used to find the GV route.

Table 4.2. Procedure 2: GV path

Algorithm for Procedure 2: GV path
Data: *GTSN, T,* (*t, gv_start*), (*t', end_node*)
Initialization: $Node_{current} = (t, gv\_start)$, $Node_{end} = (t', gv\_end)$
1 : **While** $Node_{current} \neq Node_{end}$ **do**:
2 : **If** not ($Node_{next}$):
3 : Update $Node_{current}$
4 : **If** $Node_{current} = Node_{end}$:
5 : return *GV Path*
6 : break
7 : **End While**
Result: *GV Path*

4.2. Estimation on *T* and cases of multiple UAVs

The greedy algorithm proposed in Section 4.1 requires an input on the upper bound of the

doi:10.6844/NCKU201901976

operation deadline *T*, which is supposed to be calculated in the process as well. An effective technique is to use a bisection search mechanism to converge the best *T*. In particular, we may first roughly estimate a *T* by approximately solving a Traveling Salesman Problem that visits all required air nodes without consideration on the battery renewing. Then, using that estimated *T* as a start to conduct our algorithm. Since our algorithm has considered battery consumption, it will report whether a feasible joint path planning is possible. If yes, then we reduce *T*, otherwise we increase *T*, in a bisection search fashion, until finally we obtain an optimal *T* and all the GV and UAV paths.

To deal with multiple UAVs by heuristics, we suggest using some mechanism to divide the entire regions to be several ones, each region is assigned to a UAV. We may need an encoding mechanism to record the region-UAV assignment. However, we expect such a mechanism may encounter slow convergence because of too many possible assignments, and each of which requires multiple application of the one-UAV algorithm proposed in Section 4.1. In addition, how to connect the route of the GV in different region is also a challenge. Because of so many challenges in dealing with multiple UAVs, we leave this part for the future research.

doi:10.6844/NCKU201901976

Chapter 5 Computational Experiments

In this chapter, we provide the results of the computational experiments for the model proposed in Chapter 3 and the algorithm proposed in Chapter 4. These computational experiments were conducted to test the validity of the proposed models and the proposed algorithm. We also provide a performance comparison of the proposed models and the proposed algorithm.

5.1. Settings in the Computational Environment

The computational experiments were conducted on a personal computer (PC). The specifications for the PC and the software used to perform the computational experiments are shown in Table 5.1.

Table 5.1. Technical specifications

Hardware Settings	
Operating System	Windows 7 Service Pack 1
Processor	Intel® Core™ i7-2600 CPU @ 3.40Ghz
RAM	8 GB
Software Settings	
Gurobi Solver	Version 8.1.1
Anaconda	Version 4.4.0 (64 bit)
Python	Version 3.6.8

5.2. Settings of Experimental Cases

We created eleven novel cases to test our model. For each case, we set only one candidate for the start and end node, where the start and end node were not the same nodes. For example, if node 1 was selected as start node, then node 1 was not considered as a candidate for the end node.

Each case was named in an X-Y format, where X refers to the total number of virtual cells in the airspace that need to be covered, and Y refers to the total number of ground space nodes. We assumed that all virtual cells were connected to the ground node within its range threshold, where a virtual cell was considered to be connected to at least one of the ground nodes if it was located in more than one range threshold.

doi:10.6844/NCKU201901976

Table 5.2 shows the information for each experimental case setting used in the next section, where *M* is the total number of virtual cells in the airspace that need to be covered; *N* is the total number of ground space nodes; *A* is the total number of arcs in the network; *s_node* is the starting point, and *e_node* is the ending point.

Table 5.2. Case Information

***Case* (X – Y)**	***M***	***N***	***A***	***s_node***	***e_node***
4 – 2	4	2	28	1	2
6 – 2	6	2	48	1	2
8 – 3	8	3	67	1	3
8 – 4	8	4	60	1	4
9 – 4	9	4	67	1	4
10 – 4	10	4	74	2	3
10 – 5	10	5	81	5	4
11 – 4	11	4	72	4	3
11 – 5	11	5	92	5	1
12 – 4	12	4	88	3	1
12 – 5	12	5	133	1	5

5.3. Experimental Results

In this section, the experimental results are provided using the scenarios explained in Section 5.2. This section is divided into three subsections. Subsection 5.3.1 shows the experimental results of the computations for the proposed mathematical models, and Subsection 5.3.2 shows the experimental results of the computations for the proposed algorithm. Subsection 5.3.3 discusses the performance comparison and the related analysis.

5.3.1. Mathematical Models Result

To evaluate the performance of the mathematical models mentioned in Chapter 3, each case was run on each model with the same value of T and B. Each experiment was considered alone in an individual run at the first step. All the cases were solved in a 3 hr time limit, irrespective of whether a feasible or optimal solution was found. A 3 hr time limit was used because increasing this time limit would not lead to any significant improvement or because it would take a long time to obtain any significant improvement. The results of the experiment are shown in Table 5.3. All indicator values in Table 5.3 were obtained using the Gurobi Solver.

doi:10.6844/NCKU201901976

T represents the maximum time period for the model; FS represents the number of feasible solutions obtained within the 3hr time limit; *CPU* represents the total computational time necessary to obtain the solution, which becomes 10800 (s) if an optimal solution is not obtained within the 3 hr time limit. *Gap* represents the difference between the best-known solution and the best-bound value. We set the B = 100 for all cases.

Table 5.3. Performance summary of the mathematical models

Case (X – Y)	T	M_{1-V}^{1-D}			M_{1-V}^{k-D}					
					$k=2$			$k=3$		
		***Gap* (%)**	***CPU* (s)**	***FS***	***Gap* (%)**	***CPU* (s)**	***FS***	***Gap* (%)**	***CPU* (s)**	***FS***
4 – 2	30	**0.00**	6.08	6	**0.00**	62.73	4	**0.00**	**4.68**	2
6 – 2	55	**0.00**	162.17	5	**0.00**	555.03	2	**0.00**	**113.34**	1
8 – 3	65	**0.00**	**2031.12**	4	23.53	10800.00	6	4.55	10800.00	6
8 – 4	70	**0.00**	**2285.13**	4	14.71	10800.00	10	**0.00**	10785.30	3
9 – 4	75	**0.00**	**2531.50**	4	8.82	10800.00	10	**0.00**	9220.82	10
10 – 4	75	**0.00**	**9448.36**	5	8.33	10800.00	7	16.00	10800.00	3
10 – 5	100	-	**10800.00**	-	**25.00**	**10800.00**	3	-	**10800.00**	-
11 – 4	75	**0.00**	**3588.92**	4	12.20	10800.00	8	-	10800.00	-
11 – 5	75	-	**10800.00**	-	29.79	**10800.00**	4	-	**10800.00**	-
12 – 4	75	**0.00**	**6684.16**	1	27.66	10800.00	4	70.60	10800.00	4
12 – 5	75	**0.00**	**5259.29**	1	17.07	10800.00	6	53.85	10800.00	4

Note: - means that Gap and FS are not available since the model cannot find any feasible solution within the 10800 (s) time limit

According to the results presented in Table 5.3, given the same value of T and B for the small case, the first model (M_{1-V}^{1-D}) has better performance than the second model (M_{1-V}^{k-D}) in terms of absolute Gap and CPU time. However, for the second model, using k = 3 results in better performance compared to when using k = 2 when the scale of the case is small.

When T is large, and the size of the case is greater, more time is required for the mathematical model to obtain a solution. However, since it is too time-consuming, we may not be able to tell exactly if the given T can arrive at a solution if the T value is large. As an example, given T = 100 for the Case 10 – 5, we not able to find any solution within the 3hr time limit. However, we cannot tell precisely if T = 100 is large enough to obtain a feasible solution because the same case with k = 3 is not able to find any solution while k = 2 is. This means that for the case with k = 3, more time is required to find a feasible solution.

doi:10.6844/NCKU201901976

According to Table 5.3, our mathematical models are time-consuming, including the time necessary to determine feasible solution candidates as an optimal solution. However, most of optimal solutions actually are already being found earlier before the model concludes that as an optimal solution in the end. In order to provide support for this claim, we compare the time between the first time an optimal solution was found and the end time for the model execution, as shown in Table 5.3. Table 5.4 shows the comparison for the first model (M_{1-V}^{1-D}), Table 5.5 shown the comparison for the second model (M_{1-V}^{k-D}) with k = 2, and Table 5.6 shown the comparison for the second model (M_{1-V}^{k-D}) with k = 3.

Table 5.4. Performance of M_{1-V}^{1-D}

Case (X – Y)	*T*	*FS*	Opt. First time		Opt. Final time	
			CPU (s)	*Gap* (%)	*CPU* (s)	*Gap* (%)
4 – 2	30	22	5	8.80	6.08	0.00
6 – 2	55	49	155	3.30	162.17	0.00
8 – 3	65	59	2031	0.00	2031.12	0.00
8 – 4	70	59	2285	0.00	2285.13	0.00
9 – 4	75	59	1627	3.70	2531.50	0.00
10 – 4	75	67	9448	1.49	9448.36	0.00
10 – 5	100	-	-	-	10800.00	-
11 – 4	75	69	2606	4.20	3588.92	0.00
11 – 5	75	-	-	-	10800.00	-
12 – 4	75	69	2349	4.09	6684.16	0.00
12 – 5	75	75	4467	0.77	5259.29	0.00

Note: - means the model cannot find any feasible solution within the 10800 (s) time limit

Table 5.5. Performance of M_{1-V}^{k-D} with k = 2

Case (X – Y)	*T*	*FS*	Opt. First time		Opt. Final time	
			CPU (s)	*Gap* (%)	*CPU* (s)	*Gap* (%)
4 – 2	30	13	12	15.40	62.73	0.00
6 – 2	55	25	411	5.00	555.03	0.00
8 – 3	65	34	544	26.00	10800.00	23.53
8 – 4	70	34	4103	14.71	10800.00	14.71
9 – 4	75	34	8776	8.82	10800.00	8.82
10 – 4	75	36	10192	8.33	10800.00	8.33
10 – 5	100	52	3312	26.90	10800.00	25.00
11 – 4	75	41	10613	12.20	10800.00	12.20
11 – 5	75	47	5804	40.40	10800.00	29.79
12 – 4	75	47	9579	29.80	10800.00	27.66
12 – 5	75	41	8055	17.07	10800.00	17.07

Note: - means the model cannot find any feasible solution within the 10800 (s) time limit

doi:10.6844/NCKU201901976

Table 5.6. Performance of M_{1-V}^{k-D} with $k = 3$

Case (X – Y)	*T*	*FS*	Opt. First time		Opt. Final time	
			CPU (s)	*Gap* (%)	*CPU* (s)	*Gap* (%)
4 – 2	30	9	4	0	4.68	0.00
6 – 2	55	19	21	11.90	113.34	0.00
8 – 3	65	22	3673	13.60	10800.00	4.55
8 – 4	70	23	2543	13.00	10785.30	0.00
9 – 4	75	23	7915	17.4	9220.82	0.00
10 – 4	75	25	4491	20.00	10800.00	16.00
10 – 5	100	-	-	-	10800.00	-
11 – 4	75	-	-	-	10800.00	-
11 – 5	75	-	-	-	10800.00	-
12 – 4	75	68	9733	70.60	10800.00	70.60
12 – 5	75	52	4994	53.85	10800.00	53.85

Note: - means the model cannot find any feasible solution within the 10800 (s) time limit

As shown in Tables 5.4 to 5.6, most of the best solutions were found earlier than the final time limit, especially in the case of the second model. For the first model, the difference in the *Gap* value between the first time and the final time for the best solution was found to be not too far (less than 10%). This was also the case for the second model, either using $k = 2$ or $k = 3$. However, the time required to obtain the Gap value at the final time, starting from the first time the best solution was found for the second model, takes more time compared to the first model.

5.3.2. Results for the Proposed Algorithm

To evaluate the performance of the algorithm discussed in Chapter 4, each case was run on each model with the same value of *T* and *B*, as discussed in Subsection 5.3.1. Each experiment was considered alone in an individual run in the first step. All the cases were given a 10-minute time limit, irrespective of whether a feasible or optimal solution was found or not. This time limit was set because the scale of the cases was relatively small, so we posited that the algorithm would be able to solve them in less than 10 minutes.

The results of the experiments are shown in Table 5.7, where *T* represents the maximum time period for the algorithm; *B* represents the maximum battery power level, which we set B=100 to all cases; *Opt*. represents a feasible solutions obtained within the 10 minute time limit; *CPU* represents the total computational time necessary to obtain the solution, which becomes 600 (s) if the solution is not obtained within the 10 minute time limit.

doi:10.6844/NCKU201901976

Table 5.7. Performance summary of the proposed algorithm

Case (X – Y)	***T***	***FS***	***CPU* (s)**
4 – 2	30	29	0.001
6 – 2	55	53	0.002
8 – 3	65	-	600.000
8 – 4	70	69	0.730
9 – 4	75	63	0.003
10 – 4	75	71	0.003
10 – 5	100	100	0.038
11 – 4	75	74	0.024
11 – 5	75	-	600.000
12 – 4	75	-	600.000
12 – 5	75	-	600.000

Note: - means the *FS.* is not available since the algorithm cannot find any feasible solution within the 600 (s) time limit

As shown in Table 5.7, given the values of *T* and *B*, the proposed algorithm can obtain a solution within seconds. Some cases were not able to find the solution. This may be attributed to three reasons: First, the given *T* contained a feasible solution but it was time-consuming for the algorithm to discover it, so it was not able to find the solution within the given time limit. Second, the conditions that were imposed on the algorithm prevented it from obtaining a feasible solution given *T*, meaning that it may be necessary to set a larger *T* value. Last, the given *T* did not contain any feasible solution.

Our algorithm is designed to obtain one feasible solution instead of all feasible solutions. We designed our algorithm in this way since in fact. it still time-consuming to check every possible move, even with several conditions described in Chapter 4 already being imposed. This means that in order to obtain the best solution, we need to run the algorithm with a *T* value that is almost or equal to the best solution itself. Table 5.8 shows a comparison between the initial solution using the data in Table 5.7 and the best solution that the proposed algorithm was able to obtain within the 10-minute time limit.

As shown in Table 5.8, obtaining the best solution takes more time compared to the initial solution, but most cases could still be solved within one minute. We expect this was caused by the condition that we imposed on the algorithm, especially when the UAV was able to reach ground space for the airspace when it had still not completed its task. When the best solution requires the UAV to stay longer in airspace at one or more segments, then it may take more time to find this type of path compared to finding a path where the battery

doi:10.6844/NCKU201901976

remaining at each segment is closer to the limit condition. For example, if we set the battery remaining of the UAV is at least 50%, then the path with each segment has a battery remaining closer to 50% may be found faster than the path with one or some segment has a with battery remaining closer to 0%.

For the cases that do not have any initial solution, sometimes it may be necessary to increase the *T* value to obtain the solution, which means it is the best one we can find. However, this best solution may not be an optimal solution since we impose several conditions, as explained in Chapter 4.

Table 5.8. Comparison between the initial and the best solution

Case (X – Y)	Initial solution			Best solution		
	T	***FS***	***CPU* (s)**	***T***	***FS***	***CPU* (s)**
4 – 2	30	29	0.001	24	24	6.185
6 – 2	55	53	0.002	51	51	43.247
8 – 3	65	-	600.000	68	68	16.384
8 – 4	70	69	0.730	60	60	22.111
9 – 4	75	63	0.003	59	59	0.547
10 – 4	75	71	0.003	71	71	0.003
10 – 5	100	100	0.038	96	96	3.297
11 – 4	75	74	0.024	74	74	0.018
11 – 5	75	-	600.000	76	76	0.215
12 – 4	75	-	600.000	76	76	11.365
12 – 5	75	-	600.000	80	80	0.005

Note: - means the *FS.* is not available since the algorithm cannot find any feasible solution within the 600 (s) time limit

5.3.3. Performance Comparison

In this section, we compare the performance of the proposed mathematical model M_{1-V}^{1-D} and the proposed algorithm since the proposed algorithm works only for the one-GV-one-UAV. We compare the first time a feasible solution is obtained and the best solution found by the mathematical model with the initial solution and the best solution found by the proposed algorithm. The mathematical model experimental results are based on the experimental results discussed in Section 5.3.1, and the proposed algorithm results are based on the experimental results discussed in Section 5.3.2.

Table 5.9 shows the comparison between the mathematical model (M_{1-V}^{1-D}) and the initial algorithm solution, and Table 5.10 shows the comparison between the mathematical model

doi:10.6844/NCKU201901976

(M_{1-V}^{1-D}) and the best algorithm solution, where T represents the maximum of the time period; T^{1st} represents the first feasible solution found by the mathematical model, and T^{1st}_CPU is the time when the T^{1st} was found for the first time; *Opt* represent the best solutions found by the mathematical model or a solution found by the proposed algorithm (initial and best solution), and *CPU* is the time when the *Opt* was found for the first time.

Table 5.9. Comparison between M_{1-V}^{1-D} and the initial solution of the algorithm

Case (X – Y)	T	Mathematical Model (M_{1-V}^{1-D})				Algorithm	
		T^{1st}	T^{1st}_CPU (s)	*Opt*	*CPU* (s)	*Opt*	*CPU* (s)
4 – 2	30	29	1	22	5	29	0.001
6 – 2	55	55	5	49	155	53	0.002
8 – 3	65	65	1048	59	2031	-	600.000
8 – 4	70	63	452	59	2285	69	0.730
9 – 4	75	67	1246	59	1627	63	0.003
10 – 4	75	74	1914	67	9448	71	0.003
10 – 5	100	-	-	-	-	100	0.038
11 – 4	75	74	2170	69	2606	74	0.024
11 – 5	75	-	-	-	-	-	600.000
12 – 4	75	69	2349	69	2349	-	600.000
12 – 5	75	75	4467	75	4467	-	600.000

Note: - means the model or the algorithm cannot find any feasible solution

Table 5.10. Comparison between M_{1-V}^{1-D} and the best solution of the algorithm

Case (X – Y)	Mathematical Model (M_{1-V}^{1-D})					Algorithm		
	T	T^{1st}	T^{1st}_CPU (s)	*Opt*	*CPU* (s)	T	*Opt*	*CPU* (s)
4 – 2	30	29	1	22	5	24	24	6.185
6 – 2	55	55	5	49	155	51	51	43.247
8 – 3	65	65	1048	59	2031	68	68	16.384
8 – 4	70	63	452	59	2285	60	60	22.111
9 – 4	75	67	1246	59	1627	59	59	0.547
10 – 4	75	74	1914	67	9448	71	71	0.003
10 – 5	100	-	-	-	-	96	96	3.297
11 – 4	75	74	2170	69	2606	74	74	0.018
11 – 5	75	-	-	-	-	76	76	0.215
12 – 4	75	69	2349	69	2349	76	76	11.365
12 – 5	75	75	4467	75	4467	80	80	0.005

Note: - means the model or the algorithm cannot find any feasible solution

doi:10.6844/NCKU201901976

In this section, we show both the first feasible solution and best solution found using the mathematical model since the first time the best solution is found by the mathematical model indicates that it is time-consuming, and the initial solutions found using the proposed algorithm mostly have values that are closer to the first feasible solution instead of the best solution found by the mathematical model.

As shown in Table 5.9, the initial solution found by the proposed algorithm is faster even if we compare it with the first feasible solution found by the mathematical model. The best solution found by the proposed algorithm is faster than an optimalsolution found by the mathematical model, as shown in Table 5.10. However, the best solution found by the proposed algorithm is generally not optimal. This fact hence confirms our statement in the Chapter 4, that is these conditions imposed on the algorithm may cause the result of the algorithm to be inferior to the proposed model since these conditions limit the UAV movement options.

When comparing the best solution of the proposed algorithm compared to the first feasible solution of the mathematical model, it is clear that most of the cases show that the algorithm is still faster than the mathematical model. However, there are some first feasible solutions given by the mathematical model that have better values than the best solution found by the algorithm.

5.4. Additional Experiment

In this section, we report our experimental results using a scenario created based on an actual map. We only conducted this experiment using the proposed algorithm since the case size was large enough for the mathematical model to solve this problem. In this case, we divided the area into 30 virtual cells and for the ground network, and it had 15 nodes. We called this case as Case 30-15. Figure 5.1 shows the actual map used for Case 30-15, including the grid and the ground space network. Figure 5.2 shows the details for the ground space network for Case 30-15, and Figure 5.3 shows the details for the airspace network for Case 30-15.

doi:10.6844/NCKU201901976

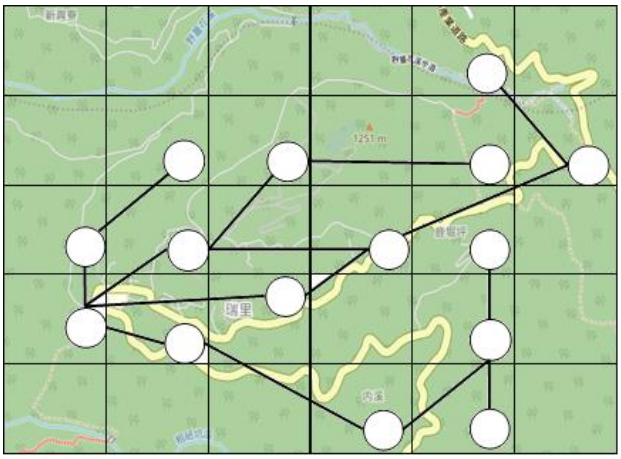

Figure 5.1. Case 30-15

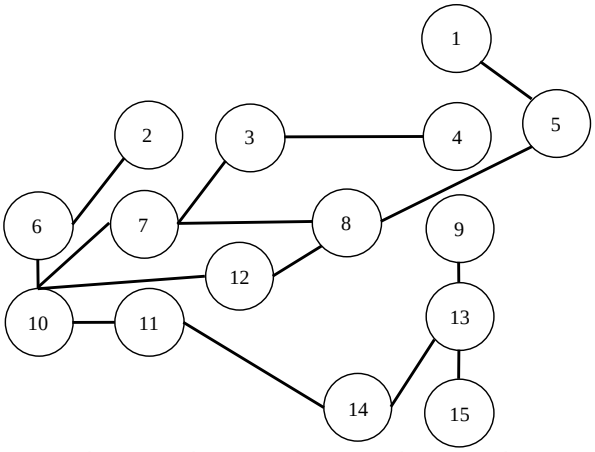

Figure 5.2. Ground space network for Case 30-15 (ground node with indices 1-15)

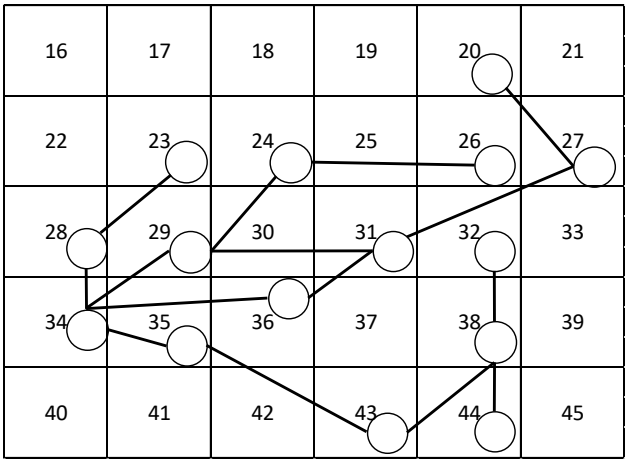

Figure 5.3. Airspace network for Case 30-15 (air nodes with indices 16-40)

doi:10.6844/NCKU201901976

In this experiment, we set B = 100 and T = 300 for the initial experiment, and then reduced the T value until we obtained the best solution. We selected nodes 1, 2, 10, and 15 as candidates for the starting and ending points due to their geographical position advantages, such as being located at the exit of the areas, in the middle of the area, or being connected to many other nodes. Table 5.11 shows the results of the experiment, where *S_node* is the starting point; *E_node* is the ending point; *Opt* is the solution value found by the algorithm, and *CPU* is the first time at which the algorithm found the solution. Each experiment was executed with a 10-minute time limit, where if the T-1 value was used, and it was unable to obtain any result within the 10-minute time limit, then we decided that this T value was the best solution.

Table 5.11. Performance summary for Case 30-15

S_node	***E_node***	**Initial solution**		**Best solution**		
		Opt	***CPU* (s)**	***T***	***Opt***	***CPU* (s)**
1	1	281	63.521	281	281	3.474
	5	280	65.546	280	280	3.015
	10	275	66.644	261	**261**	16.314
	15	**272**	71.185	263	263	0.402
5	1	273	0.075	273	273	0.060
	5	272	0.070	272	272	0.590
	10	267	0.065	263	263	58.645
	15	**264**	0.067	258	**258**	7.092
10	1	273	0.071	273	273	0.064
	5	272	0.068	266	266	50.730
	10	267	0.070	267	267	0.058
	15	**264**	0.071	261	**261**	0.081
15	1	-	-	-	-	-
	5	-	-	-	-	-
	10	-	-	-	-	-
	15	-	-	-	-	-

Based on experimental results shown in Table 5.11, given the candidates for the starting and ending points, the minimum time that the UAV was able to achieve in order to cover the entire area was T = 258. This T could be achieved if the GV and the UAV started their tasks at node 5 and finished their tasks at node 15.

It should be noted that when we set the starting node at 15, the proposed algorithm could not solve this problem within the 10-minute time limit, even when for the rest of the nodes, we could obtain the solution in less than 2 minutes. We attributed this situation to the

doi:10.6844/NCKU201901976

fact that the data sequence was stored to the algorithm. Even if we had already imposed several rules to enhance the searching path process speed, we still were not able to interfere with the decision as to which node the algorithm would check first. Since we adopted the depth-first search algorithm, this meant that selecting the wrong node as a starting node would lead to an unnecessary search. For example, if we selected node 15 as a starting point, it would go to cell 37 and then either scan it or would go to cell 30 (since we stored the data based from small to large numbers), and so on. Even when we had already imposed rules to reduce unnecessary searches, since it chose the wrong cell, it still ultimately performed an unnecessary search.

5.5. Summary

We performed computational experiments to evaluate the performance of the proposed mathematical models and the proposed algorithm. According to the experimental results, the mathematical models are too time-consuming, both to in terms of obtaining the first feasible solution and in terms of determining the candidate for a feasible solution as an optimal solution. The experiments also showed that the mathematical models take a long time to arrive at a solution if the given T contains a feasible solution or an infeasible value when the T value is set to be same as an optimal or nearly optimal solution. For example, if an optimal solution is $T = 75$, then when we set $T = 75$ or $T = 74$, it is possible that the proposed mathematical model will take a lot of time to decide whether the given T contains a feasible solution or an infeasible one.

The proposed algorithm can obtain a feasible solution faster than the proposed mathematical model. However, the proposed algorithm, in general, is only able to obtain a feasible rather than an optimal solution. This is because we imposed some rules. These rules were set to avoid unnecessary searches by the algorithm. Because our algorithm was developed based on a depth-first search algorithm, then the performance of the algorithm also depends on the sequence of the data stored into the algorithm. Even though we had already reduced the number of unnecessary searches by imposing several rules, if the algorithm searches a wrong node due to the sequence of data stored into it, then the algorithm still conducts an unnecessary search that leads to it taking too long or being unable to find a feasible solution within the specified time limit.

doi:10.6844/NCKU201901976

Chapter 6 Conclusions and Suggested Future Research

6.1. Conclusions

UAVs have many advantages when conducting area coverage compared to the satellite or other aerial systems, but their operation is limited due to battery constraints. This limitation can be overcome by combining a UAV with a ground vehicle. Thus, this study investigates a joint UAV and GV path planning problem for area coverage taking energy into consideration. The objective of this research is to calculate optimal routes for both GV and UAVs so that all the target areas can be scanned by UAVs within a minimum allocated total time. Planning the path for UAVs and GVs must be done simultaneously. Considering the energy consumption of UAVs for path planning makes our model more practical. The conclusions derived from our study are as follows:

1. There are two integer programming models proposed, (1) one GV with one UAV case (M_{1-V}^{1-D}), and (2) one GV with multiple UAVs case (M_{1-V}^{k-D}). The joint UAV and GV path planning problem for area coverage is quite new, and thus there is little literature that can be used as a basis for comparison. To the best of our knowledge, most related literature only considered the one GV with one UAV cases. We only found one study by Hu et al. (2018) where one GV with multiple UAVs case was studied, but they only proposed an algorithm to deal with the problem. Thus, we might have arguably proposed the first mathematical programming model that can deal with the one GV with multiple UAVs case.
2. Studies that have considered the UAV battery power limitation typically assumed a constant battery consumption rate (or in other words, a constant flying speed). Thus, they can replace the battery limit with a constant time-bound flight. However, in practice, the battery consumption rate may be different for different flight modes, as is the case in the models proposed in the current work.
3. Our proposed models and proposed algorithm can calculate the detailed movement of both UAVs and GVs, including when they arrive at and depart from a specific air or ground node to conduct a search mission, recharge, wait, or move.
4. The computational testing results for our mathematical programming models show that the one GV with one UAV case performs better compared to the one GV with multiple UAVs case in term of absolute Gap and CPU Time given the same parameters of *T* and

doi:10.6844/NCKU201901976

B for each test scenario. However, neither model was able to solve a large case since it was quite time-consuming. It was also quite time-consuming to determine a feasible solution as the best or optimal solution.

5. The computational testing results for our proposed algorithm showed that our proposed algorithm was able to solve all of the cases faster than the mathematical model, even if we compared the first time at which a best solution was found by the mathematical model with the proposed algorithm. However, the proposed algorithm may not be able to give an optimal solution due to the conditions imposed on our algorithm.
6. Since our proposed algorithm was developed based on a depth-first search (DFS) algorithm, the algorithm had to search for a possible move by seeking a single path until it reached a dead-end and then backtracking to a point where it could choose another path if there were no more possible moves in a path. This meant that our algorithm was also affected by the input data storage sequence. This factor caused the possibility that the algorithm would take too much time or would not be able to solve the problem for some cases given the specific time limit.

6.2. Suggested Future Research

To the best of our knowledge, there are few former works related to the joint UAV and ground vehicle path planning problem for area coverage. Thus, there are some related issues worthy of further investigation. Here, we list a few as follows:

1. The communication constraint between the UAV and the GV

In our setting, we assume the UAV can always communicate with the GV at any time (e.g., through a satellite connection), even if the UAV is very far away from the GV. However, in practice, the connection between UAV and GV may require their distance to be within some given range at any time. In such cases, it would be more difficult to handle the routings for both the GV and the UAV.

2. Relaxing the assumption for the battery consumption rate and speed of a UAV

To simplify the problem, we assumed a UAV to have 3 moving modes (each with a corresponding battery consumption rate and speed): (1) a search mode to be performed on an air node, (2) a moving mode that moves between an air node and a ground node, and (3) a stay/waiting mode that stays at a ground node waiting to meet a GV.

doi:10.6844/NCKU201901976

In a real situation, one could further consider the detailed UAV movement with more moving modes, which would give a more accurate estimate of the battery consumption rate and speed, or, even more realistically, if one were to assume parallel arcs between some nodes to represent different consumption rates or speeds between two nodes.

3. Consideration of the use of several types of UAVs

In our model, we assumed that the fleet of UAVs to be of the same type. This assumption resulted in all of the UAVs having the same battery consumption rate and speed. In a real situation, one may deploy different types of UAVs for different missions.

4. The operational time limit

Take an example for area coverage application search and rescue where it is necessary to find a target in designated areas. The target will be definitely found if the UAV successfully covers all of the designated areas. If there is no time limitation, then our model may be applicable to this problem.

However, search and rescue operations that need to find a target (e.g. person) usually are time-limited operations since they are used in life threatening situations. These time-limited operations may make the completion of area coverage for all target areas impossible, so our model would not be applicable.

While there have been many studies considering a UAV only or a UAV with a stationary battery station, studies on joint UAV and ground vehicle operations are still limited in number. This is an interesting problem worthy of further investigation since the application is already available in real situations. Readers may refer to the Land Rover Project Hero for an example of a joint UAV and ground vehicle operation application used in search and rescue.

5. Grid size

In our problem, we assumed that the grid size that represent the virtual cell in the airspace are known beforehand. In fact, the grid size actually is affected by the camera field of view (FOV) of the UAV. For more detail of the UAV FOV, please read the research done by Nam et al. (2016),

6. Develop better algorithm and math programming models

Since our math programming model was quite time-consuming, as for the algorithm is still limited to the one GV with one UAV case and in the algorithm, implementation based

doi:10.6844/NCKU201901976

on a depth-first search algorithm also time-consuming, it is recommended the both models and the algorithm be improved. Developing an algorithm for the multiple UAV case is also a good opportunity for future research since until this study, there has been little literature discussing this problem.

doi:10.6844/NCKU201901976

Reference

Avellar, G., Pereira, G., Pimenta, L., & Iscold, P. (2015). Multi-UAV routing for area coverage and remote sensing with minimum time. *Sensors, 15*(11), 27783-27803.

Barrientos, A., Colorado, J., Cerro, J. d., Martinez, A., Rossi, C., Sanz, D., & Valente, J. (2011). Aerial remote sensing in agriculture: A practical approach to area coverage and path planning for fleets of mini aerial robots. *Journal of Field Robotics, 28*(5), 667-689.

Choi, Y., Choi, Y., Briceno, S., & Mavris, D. N. (2018). Coverage path planning for a UAS imagery mission using column generation with a turn penalty. *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*.

Cuda, R., Guastaroba, G., & Speranza, M. G. (2015). A survey on two-echelon routing problems. *Computers & Operations Research, 55*, 185-199.

Dalamagkidis, K. (2015). Definitions and terminology. In K. P. Valavanis & G. J. Vachtsevanos (Eds.), *Handbook of unmanned aerial vehicles* (pp. 43-55). Dordrecht: Springer Netherlands.

Fondation Suisse de Deminage (FSD). (2016). *Drones in humanitarian action*. Geneva. Retrieved from https://drones.fsd.ch/wp-content/uploads/2016/11/Drones-in-Humanitarian-Action.pdf

Galceran, E., & Carreras, M. (2013). A survey on coverage path planning for robotics. *Robotics and Autonomous Systems, 61*(12), 1258-1276.

Garone, E., Naldi, R., Casavola, A., & Frazzoli, E. (2010). Cooperative mission planning for a class of carrier-vehicle systems. In *49th IEEE Conference on Decision and Control (CDC)*. Atlanta, Georgia, USA: IEEE

Hu, M., Liu, W., Peng, K., Ma, X., Cheng, W., Liu, J., & Li, B. (2018). Joint routing and scheduling for vehicle-assisted multi-drone surveillance. *IEEE Internet of Things Journal*, 1-1.

Khan, A., Noreen, I., & Habib, Z. (2017). On complete coverage path planning algorithms for non-holonomic mobile robots: Survey and challenges. *Journal of Information Science and Engineering, 33*(1), 101-121.

Luo, Z., Liu, Z., & Shi, J. (2017). A two-echelon cooperated routing problem for a ground vehicle and its carried unmanned aerial vehicle. *Sensors, 17*(5), 1144.

doi:10.6844/NCKU201901976

Luo, Z., Liu, Z., Shi, J., Wang, Q., Zhou, T., & Liu, Y. (2018). The mathematical modeling of the two-echelon ground vehicle and its mounted unmanned aerial vehicle cooperated routing problem. *2018 IEEE Intelligent Vehicles Symposium (IV)*.

Manyam, S. G., Casbeer, D. W., & Sundar, K. (2016). Path planning for cooperative routing of air-ground vehicles. *2016 American Control Conference (ACC)*.

Mathew, N., Smith, S. L., & Waslander, S. L. (2013). A graph-based approach to multi-robot rendezvous for recharging in persistent tasks. *2013 IEEE International Conference on Robotics and Automation*.

Mathew, N., Smith, S. L., & Waslander, S. L. (2015a). Multirobot rendezvous planning for recharging in persistent tasks. *IEEE Transactions on Robotics, 31*(1), 128-142.

Mathew, N., Smith, S. L., & Waslander, S. L. (2015b). Planning paths for package delivery in heterogeneous multirobot teams. *IEEE Transactions on Automation Science and Engineering, 12*(4), 1298-1308.

Moravec, H., & Elfes, A. (1985). High resolution maps from wide angle sonar. *Proceedings*. *1985 IEEE International Conference on Robotics and Automation*.

Mourelo Ferrandez, S., Harbison, T., Weber, T., Sturges, R., & Rich, R. (2016). Optimization of a truck-drone in tandem delivery network using k-means and genetic algorithm. *Journal of Industrial Engineering and Management, 9*(2), 374.

Nam, L. H., Huang, L., Li, X. J., & Xu, J. F. (2016). An approach for coverage path planning for UAVs. *2016 IEEE 14th International Workshop on Advanced Motion Control (AMC)*.

Nedjati, A., Izbirak, G., Vizvari, B., & Arkat, J. (2016). Complete coverage path planning for a multi-UAV response system in post-earthquake assessment. *Robotics, 5*(4), 26.

Otto, A., Agatz, N., Campbell, J., Golden, B., & Pesch, E. (2018). Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: A survey. *Networks, 72*(4), 411-458.

Pham, T. H., Bestaoui, Y., & Mammar, S. (2017). Aerial robot coverage path planning approach with concave obstacles in precision agriculture. *2017 Workshop on Research, Education, and Development of Unmanned Aerial Systems (RED-UAS)*.

Tokekar, P., Hook, J. V., Mulla, D., & Isler, V. (2016). Sensor planning for a symbiotic UAV and UGV system for precision agriculture. *IEEE Transactions on Robotics, 32*(6), 1498-1511.

doi:10.6844/NCKU201901976

Vachtsevanos, G. J., & Valavanis, K. P. (2015). Military and civilian unmanned aircraft. In K. P. Valavanis & G. J. Vachtsevanos (Eds.), *Handbook of unmanned aerial vehicles* (pp. 93-103). Dordrecht: Springer Netherlands.

Vergouw, B., Nagel, H., Bondt, G., & Custers, B. (2016). Drone technology: Types, payloads, applications, frequency spectrum issues, and future developments. in b. custers (Ed.), *The future of drone use: Opportunities and threats from ethical and legal perspectives* (pp. 21-45). The Hague: T.M.C. Asser Press.

doi:10.6844/NCKU201901976