# Optimization Models for the Inspection of Electric Power System Assets Using Drones

by

Hyeoncheol Baik

A dissertation submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
August 3, 2019

Keywords: Drone, Unmanned aircraft system, Transmission tower inspection routing, Wind turbine inspection routing, Optimization

Copyright 2019 by Hyeoncheol Baik

Approved by

Jorge Valenzuela, Chair, Professor of Industrial and Systems Engineering
Saad Biaz, Professor of Computer Science and Software Engineering
Alexsandr Vinel, Assistant Professor of Industrial and Systems Engineering
Kang Bok Lee, Assistant Professor of Systems and Technology

## Abstract

To inspect electric power system assets, electric utilities have relied on inspection personnel and costly equipment. For example, the utilities inspect a power transmission tower by dispatching line crews and helicopters. Wind turbines are inspected by a technician using rope access. These practices are ineffective due to high operation costs, safety concerns, and low inspection rate. Recently, there has been an increasing interest in using a drone, or also known as unmanned aircraft system (UAS). We explore the drone routing optimization models for inspecting these electric power system assets. In the first chapter, we formulate an optimization model to find an efficient drone flight path for inspecting a transmission tower. The objective of the model is to maximize a function involving three performance ratios, namely, flight time, image quality, and tower coverage. This optimization problem is solved by using two metaheuristic algorithms which are a particle swarm optimization (PSO) based-algorithm and a simulated annealing (SA) based-algorithm. In the second chapter, we develop a routing optimization model to minimize the total operation time for inspecting a wind farm by using one drone and one ground vehicle for carrying the drone. This optimization problem is decomposed into two steps. The first step clusters the wind turbines and optimizes the drone routing in each cluster by solving the classical traveling salesman problem (TSP) using an integer linear programming model. The second step optimizes the ground vehicle routing by solving the equality generalized traveling salesman problem (E-GTSP) using an integer linear programming model. We test our approach using three case studies created by using actual wind farm locations. Finally, in the third chapter, we extend the optimization model

presented in the previous chapter by considering the visual line-of-sight regulation and a multi-drone system. The multi-drone routes are optimized by solving the multiple traveling salesman problem (mTSP) using a mixed integer linear programming model. Also, the proposed model is tested on three case studies.

**Acknowledgments**

First and foremost, I would like to thank Dr. Jorge Valenzuela, my academic and research advisor at Auburn University, for all support, guidance, and patience throughout my Ph.D. study. Completing a Ph.D. was a challenging and longstanding journey. However, his knowledge and insightful comments were fundamental to overcome my challenges, and his enthusiastic and frank discussion shaped me as a researcher.

I would like to thank my committee members. Developing the solution approaches were possible due to the constructive feedbacks from Dr. Saad Biaz and the techniques I have learnt from Dr. Alexsandr Vinel's classes. Dr. Kang Bok Lee is always supportive and taught me the joy of learning. Also, I would like to thank Dr. Sumin Han for reviewing my dissertation.

My special thanks go to Eddie and Ruthie Ferrell for all their help, prayers, and fun times. In addition, I would like to thank Dr. Kyongsun Kim who was always willing to discuss my issues and provide inspiration and advices.

Of course, I would like to thank my parents and parents in law from the bottom of my heart for all their endless love and support. I am here because of you.

Finally, none of this would have been possible without the continuous support and faith of my wife, Woo Kyung Shim and my daughter Laeun.

## Table of Contents

Abstract ..... ii

Acknowledgments ..... iv

List of Figures ..... viii

List of Tables ..... xi

List of Abbreviations ..... xii

Introduction ..... 1

Chapter 1 Drone Path Planning for Inspecting Electric Transmission Towers ..... 2

1.1 Abstract ..... 2

1.2 Introduction ..... 3

1.3 The proposed model ..... 5

1.3.1 Tower model ..... 6

1.3.2 Flight time ratio ..... 7

1.3.3 Image quality ratio ..... 8

1.3.4 Tower coverage ratio ..... 10

1.3.5 Optimization model ..... 13

1.4 Solution algorithms ..... 15

1.4.1 PSO-based solution algorithm ..... 15

1.4.2 SA-based solution algorithm ..... 17
1.5 Numerical experiments ..... 20
1.5.1 Inspection strategies ..... 21
1.5.2 Parameter tuning and stopping criterion ..... 22
1.5.3 Experimental results ..... 24
1.6 Conclusion ..... 30
References ..... 32
Chapter 2 Drone Routing for Inspecting Wind Farms ..... 37
2.1 Abstract ..... 37
2.2 Introduction ..... 38
2.3 Proposed optimization model and solution approach ..... 42
2.3.1 Optimizing the drone route ..... 43
2.3.2 Optimizing the ground vehicle route ..... 45
2.4 Experimental results ..... 48
2.4.1 System parameters and case studies ..... 49
2.4.2 Routing solutions ..... 53
2.4.3 Effect of clustering and ground vehicle routing ..... 55
2.4.4 Results of the total operation time ..... 56
2.4.5 Effect of the flight speed ..... 57
2.4.6 Effect of the flight endurance ..... 58
2.5 Conclusion ..... 60
References ..... 62

Chapter 3 Inspecting Wind Farms with A Multi-Drone System ..... 67

3.1 Abstract ..... 67

3.2 Introduction ..... 68

3.3 Proposed optimization model and solution approach ..... 71

3.3.1 Optimizing the multi-drone routes ..... 71

3.3.2 Optimizing the ground vehicle route ..... 76

3.4 Experimental results ..... 77

3.4.1 System parameters and case studies ..... 78

3.4.2 Routing solutions ..... 78

3.4.3 Results of the total operation time ..... 80

3.4.4 Results of swapping with new battery ..... 81

3.4.5 Effect of the number of stop locations ..... 83

3.4.6 Effect of the flight endurance ..... 84

3.5 Conclusion ..... 85

References ..... 86

Conclusion ..... 89

Appendix A: Nomenclature for Chapter 1 ..... 91

Appendix B: Nomenclature for Chapter 2 ..... 93

Appendix C: Nomenclature for Chapter 3 ..... 94

## List of Figures

1.1 132kV transmission tower dimension ..... 7

1.2 Transmission tower model ..... 7

1.3 Maximum flight distance for $N^{max} = 3$ ..... 8

1.4 Ground Sampling Distance (GSD) ..... 9

1.5 GSD in inspecting a transmission tower ..... 9

1.6 Camera, image, and real coordinate system ..... 10

1.7 Transforming real coordinates to camera coordinates ..... 11

1.8 Transforming camera to image coordinates ..... 12

1.9 Front and back surface elements ..... 13

1.10 Pseudocode for the PSO ..... 16

1.11 Flowchart for the SA ..... 18

1.12 Pseudocode for the SA ..... 20

1.13 Strategy S1 ($N = 12$) ..... 22

1.14 Strategy S2 ($N = 12$) ..... 22

1.15 Strategy S3 ($N = 12$) ..... 22

1.16 Average total performance ratios for different numbers of waypoints ..... 26

1.17 Flight path for the best solution ..... 27

1.18 Average total performance ratios for different weights of the fitness function by changing $\omega_1$ ..... 28

1.19 Average total performance ratios for different weights of the fitness function by changing $\omega_2$ ..... 29

1.20 Average total performance ratios for different weights of the fitness function by changing $\omega_3$ ..... 29

2.1 15-wind turbine cluster assignment ..... 45

2.2 Flow chart of the solution approach ..... 48

2.3 Histogram of the number of wind turbines per wind project in Texas ..... 50

2.4 Google satellite maps of the wind turbines for the three case studies ..... 51

2.5 Small-sized wind farm routing solution for the drone and ground vehicle ..... 54

2.6 Medium-sized wind farm routing solution of the drone and ground vehicle ..... 54

2.7 Large-sized wind farm routing solution of the drone and ground vehicle ..... 55

2.8 Effect of changing the flight speed in the small-sized wind farm ..... 57

2.9 Effect of changing the flight speed in the medium-sized wind farm ..... 58

2.10 Effect of changing the flight speed in the large-sized wind farm ..... 58

2.11 Effect of changing the flight endurance in the small-sized wind farm ..... 59

2.12 Effect of changing the flight endurance in the medium-sized wind farm ..... 60

2.13 Effect of changing the flight endurance in the large-sized wind farm ..... 60

3.1 Pseudocode of the solution approach ..... 77

3.2 Small-sized wind farm routing solution for two drones and one ground vehicle ..... 79

3.3 Medium-sized wind farm routing solution for two drones and one ground vehicle ..... 79

3.4 Large-sized wind farm routing solution for two drones and one ground vehicle ..... 80

3.5 Results of the total operation time ..... 81

3.6 Effect of the number of stops in the small-sized wind farm ..... 83

3.7 Effect of changing the flight endurance for two drones and one ground vehicle ..... 84

## List of Tables

1.1 System parameters ..... 21

1.2 PSO parameter tuning results ..... 23

1.3 SA parameter tuning results ..... 23

1.4 Performance results over 3 strategies with different algorithms ($N = 12$) ..... 24

1.5 Performance results for different numbers of waypoints ..... 25

1.6 Performance results for different swarm sizes (S1-PSO with $N = 16$) ..... 26

2.1 System parameters ..... 49

2.2 Geographic locations of the wind turbines ..... 51

2.3 Comparative results of the total operation time ..... 56

3.1 System parameters ..... 78

3.2 Results of swapping with new battery ..... 82

## List of Abbreviations

| | |
|---|---|
| CV | Coefficient of Variation |
| E-GTSP | Equality Generalized Traveling Salesman Problem |
| EPRI | Electric Power Research Institute |
| FAA | Federal Aviation Administration |
| GRASP | Greedy Randomized Adaptive Search Procedure |
| GSD | Ground Sampling Distance |
| LKH | Lin-Kernighan-Helsgaun |
| MILP | Mixed Integer Linear Programming |
| MIP | Mixed Integer Programming |
| PDSTSP | Parallel Drone Scheduling Traveling Salesman Problem |
| PSO | Particle Swarm Optimization |
| SA | Simulated Annealing |
| T&D | Transmission and Distribution |
| TSP | Traveling Salesman Problem |
| UAS | Unmanned Aircraft System |
| VLOS | Visual Line-Of-Sight |

# Introduction

Over the decades, electric utilities have not much changed the ways of inspection their electric power system assets. The conventional ways of inspection have issues such as high operation and maintenance costs, safety concerns, and low inspection rate. Thus, utilities are seeking innovative technologies that can replace the existing practices. A promising alternative is the use of a drone, or also known as unmanned aircraft system (UAS). In recent years, there has been growing interest in identifying innovative ways to apply this technology in the commercial sector. For example, Amazon has developed a drone-based delivery service since the plan was released on 2013. In the electric power industry, a drone can be utilized to perform numerous tasks. It can be used to inspect electric power system assets, assess weather-related disaster damage and monitor vegetation on power transmission and distribution (T&D) systems. With such applications in mind, we have developed optimization models for the inspection of transmission towers and wind turbines using drones.

The remainder of this dissertation is organized as follows. Chapter 1 presents an optimization model to find an efficient drone flight path for inspecting a transmission tower. Chapter 2 proposes a drone routing optimization model to minimize the total operation time for inspecting a wind farm with a ground vehicle that is used to carry the drone. Finally, Chapter 3 describes an optimization model to minimize the total operation time for inspecting a wind farm with a multi-drone system.

# Chapter 1 Drone Path Planning for Inspecting Electric Transmission Towers

## 1.1 Abstract

To detect faults on the power transmission and distribution systems, current electric utilities perform a visual inspection by dispatching line crews and helicopters. This practice has disadvantages such as high operation costs and safety concerns. To resolve these issues, power utilities are considering the use of a drone. In this chapter, we formulate an optimization model to find an efficient flight path for a drone for visually inspecting a transmission tower. The objective of the model is to maximize a function involving three performance ratios, namely, flight time, image quality, and tower coverage. The optimization model is non-linear, non-differentiable, and multimodal. We solve the problem by using a particle swarm optimization (PSO) based-algorithm and a simulated annealing (SA) based-algorithm and compare their results. We test the model under three inspection strategies. The experimental results show that the PSO-based algorithm outperforms the SA-based algorithm. They also show that the proposed model can provide a flight path that comprises a good balance over the three performance ratios.

**Keywords:** Drone, Unmanned aircraft system, Camera calibration, Particle swarm optimization, Simulated annealing, Transmission tower inspection

## 1.2 Introduction

Over the decades, electric utilities have relied on foot patrol and helicopters to perform a visual inspection on power transmission and distribution (T&D) systems. These practices are time-consuming, expensive and highly dangerous. Operating helicopters can cost up to $1,500 an hour [1] and more than $2 per kilometer [2]. According to a report from the U.S. Bureau of Labor Statistics, electrical power-line installers and repairers ranked in the top 10 lists of civilian jobs with high fatal work injury rates in 2017 [3]. In addition, a visual inspection is the main mean of identifying defects on T&D systems [4]. For these reasons, utilities are seeking innovative technologies that can replace line crews and helicopters in power T&D inspection. One promising alternative is the use of a drone, or also known as unmanned aircraft system (UAS) [5]. The federal aviation administration (FAA) defines drones as aircrafts maneuvered by an operator on the ground instead of a human pilot onboard [6].

Drone applications in power T&D systems have shown several benefits since early drone researchers proposed a schematic concept and proved it with laboratory tests [7, 8]. First, a drone can be equipped with diverse types of sensors for an inspection. The electric power research institute (EPRI) elaborates possible inspection scenarios using the various sensors with a comprehensive summary of up-to-date sensing technology [9, 10]. Supporting their proposal, EPRI demonstrates that a drone can be used effectively to assess storm damage [11]. Second, a drone can be used as a mobile platform to collect and relay data [9, 10, 12]. Third, a drone can execute two different levels of inspections (macro/micro) depending on the wing type of a drone: fixed-wing or rotary-wing. Since a fixed-wing drone cannot hover itself, it provides a macro-level inspection which, for example, is surveying power lines in a large area [13]. On the other hand, a rotary-wing drone can perform a micro-level inspection such as checking for mechanical failures

on transmission tower components [14]. Sometimes, it is possible to use cooperative operation of both types [15]. Fourth, a drone can improve routine inspections such as vegetation monitoring and unscheduled inspections such as weather-related damages. Lastly, a drone allows us to implement an automatic inspection system with the aid of image processing [16, 17].

However, using a drone is challenging because there are still major barriers blocking its full implementation. Technically, it has a limited battery capacity and may not be as capable operating in a large area when compared to a manned helicopter [7]. In addition, a stable flight control mechanism should be embedded to keep a drone away from a strong electromagnetic field and any obstacles [18, 19]. In [20], the authors point out that little research has been done on the flying robot because it deals with more complex technical problems than the climbing robot. Regulatory issues are another factor constraining the operation scope. For example, in the United States, a drone must fly within the visual line-of-sight (VLOS) of the remote pilot or observer during daylight hours according to a rule enacted by the FAA [21]. Other issues include privacy intrusions, liability of an accident, and hacking. We emphasize that the limited battery capacity and VLOS requirement reinforce the need to determine the flight path of a drone for operational efficiency.

To the best of our knowledge, no research has addressed the flight path planning optimization problem of a drone for inspecting transmission towers, although there are some research works for inspecting other objects such as structure and terrain [22 – 24]. Thus, our research question is how to determine an optimal flight path considering three areas of performance: (i) flight time, (ii) image quality, and (iii) tower coverage that is the total surface area of the transmission tower captured by the camera of a drone. Note that our model will help to build an autonomous system which is the goal of most literature research [5, 16, 17, 19].

The remainder of this chapter is organized as follows. In Section 1.3, we describe how to mathematically model this flight path planning problem. Section 1.4 shows how to apply the particle swarm optimization (PSO) and simulated annealing (SA) algorithm to find the optimal flight path. Section 1.5 discusses the numerical experiments and results analysis. Finally, Section 1.6 states our conclusion.

## 1.3 The proposed model

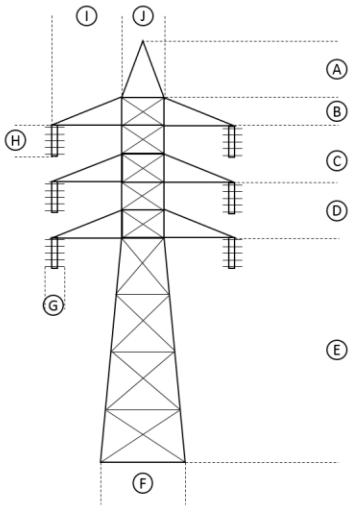
To perform a micro-level inspection of a transmission tower by hovering, we use a rotary-wing drone that navigates autonomously through waypoints input by a remote pilot. A waypoint is a position that the drone must visit on a flight path. The optimization problem is to find the optimal flight path considering flight time, image resolution, and tower coverage performance ratios. The flight time ratio is computed with respect to the maximum flight time $FT^{max}$ obtained from the hovering time at $N^{max}$ waypoints plus the corresponding cruising time. It is assumed that the drone flies at a constant speed, *FS* m/s, hovers at a waypoint for a constant time of *H* seconds and is not affected by wind. The battery charge should allow the drone to fly for $FT^{max}$ seconds. To compute the image quality ratio, we assume that one standard digital camera is mounted on the drone and cannot be tilted. By using a camera gimbal, the drone keeps the camera upright with respect to the horizon regardless of its motion. We use a prime lens on the camera to take images at a constant resolution. Thus, the distance from the lens to the image sensor is *FL* millimeters. For the image size, we choose the 4:3 aspect ratio. The vertical number of pixels is given by $N^p$. To comply with FAA rules, the drone must weigh less than 25 kilograms including the camera [21]. The tower coverage ratio is calculated by representing the tower surface by a discrete set of $N^s$ triangular and rectangular elements. These surface elements are denoted by the set $\Phi_S$. Thus, the

tower coverage is obtained by totaling the different surface elements identified in each of the images taken by the drone during its flight.

The drone is restricted to flying within the area given by the set $\Phi$ which is a cuboid delimited by **D.** This area should meet the VLOS requirement and consider the no tree zone within a transmission line right-of-way. We designate the position $(X^{tc}, Y^{tc}, 0)$ to be the center-base of the tower and the position $(0, 0, 0)$ to be the home-base of the drone. To consider the electromagnetic field around the tower and the tower itself, we define the no-fly zone $\Phi_N$. It is assumed that the drone takes off from the home-base and returns to the same location. At each waypoint, the drone hovers and takes an image. We assume that the drone flies along a straight line connected by two waypoints. If the drone encounters the no-fly zone, it detours around the boundaries of the no-fly zone. We also assume that there is no other obstacle within the flying area.

### 1.3.1 Tower model

The tower surface is represented by a discrete set of $N^s$ triangular and rectangular elements. As an example, we choose the 132kV transmission tower that is used to transmit extra-high voltage over a long distance. Figure 1.1 shows the tower dimensions obtained from [25] and Figure 1.2 depicts the tower model and its surface elements.




| Symbol | Length (meter) |
|---|---|
| A | 2.03 |
| B | 1.39 |
| C | 4.27 |
| D | 4.27 |
| E | 22.05 |
| F | 4.61 |
| G | 0.25 |
| H | 2.24 |
| I | 3.03 |
| J | 1.56 |

**Figure 1.1** 132kV transmission tower dimension

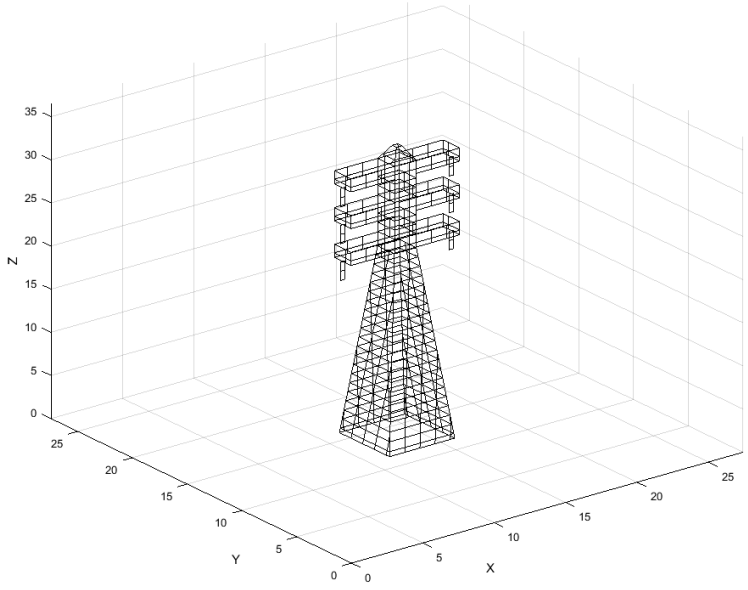



**Figure 1.2** Transmission tower model

### 1.3.2 Flight time ratio

We define the flight time ratio of path *i* as the difference between the maximum flight time ($FT^{max}$) and the flight time of path *i* relative to $FT^{max}$ (see Eq. (1.8)). The flight time consists of hovering and cruising times. The hovering time is calculated by multiplying *H* by *N*. The cruising

time is obtained by dividing the Euclidean distance between waypoints by the flight speed (*FS*). Figure 1.3 illustrates the process of determining $FT^{mx}$ when the maximum number of waypoints $N^{max}$ is 3. The process starts by dividing the flying area into $N^{max}$ cuboids. The first waypoint is assigned to be the vertex opposite from the drone home-base. The second waypoint is the vertex that has the longest distance from the first waypoint. This process continues until the remaining waypoints are determined. The maximum flight distance ($FD^{max}$) is the sum of the distances between two consecutive waypoints. The cruising time is obtained by dividing $FD^{max}$ by the flight speed *FS*. The hovering time is computed by multiplying $H$ by $N^{max}$. The $FT^{max}$ is the sum of the cruising and hovering times.

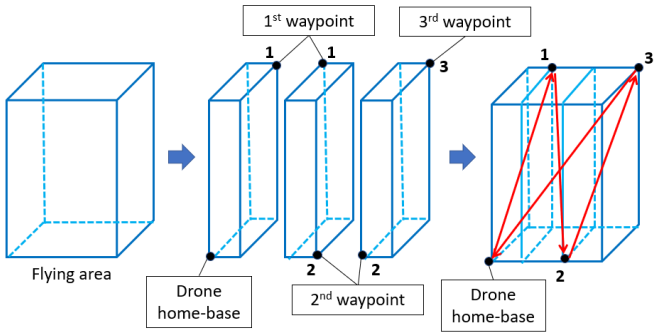



**Figure 1.3** Maximum flight distance for $N^{max} = 3$

### 1.3.3 Image quality ratio

The most common way to measure image quality is to use pixel resolution, which is a camera dependent specification (e.g. 12 megapixels). However, pixel resolution does not provide a good way to measure the image quality of a feature in the image. Instead, we use spatial resolution that refers to the size of the smallest possible object discernible in the image. To measure the spatial resolution, we use the ground sampling distance (GSD) that is the distance between two pixels' centers measured on the ground as shown in Figure 1.4 [26]. The unit of measure is

millimeter per pixel. A lower GSD provides a higher image quality. In our situation, the ground corresponds to the vertical plane that is parallel to the drone position and contains the tower center-base (see Figure 1.5). As the drone moves closer to the tower, the GSD decreases and thus, the quality of the image will increase. The maximum GSD ($G^{max}$) is the GSD of an image taken at the farthermost waypoint from the tower center-base constrained to the flying area. We define the image quality ratio of path *i* as the difference between $G^{max}$ and the average image GSD of path *i* relative to $G^{max}$ (see Eq. (1.9)).

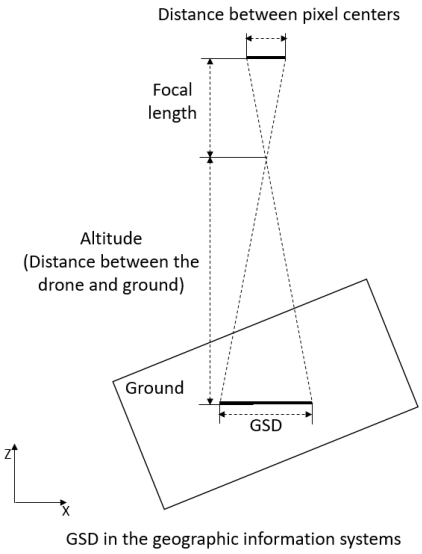



**Figure 1.4** Ground Sampling Distance (GSD)

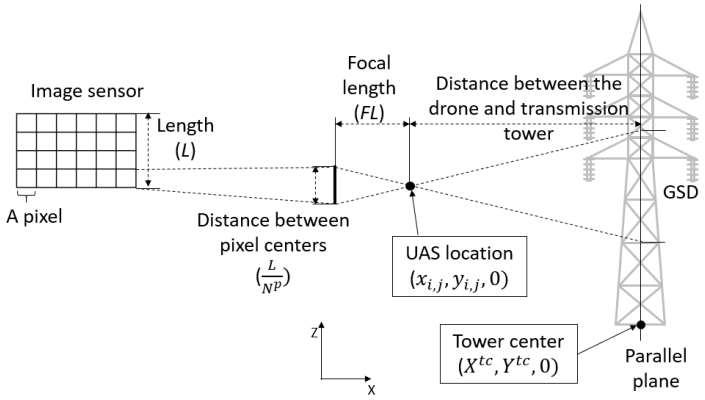



**Figure 1.5** GSD in inspecting a transmission tower

### 1.3.4 Tower coverage ratio

We define the tower coverage ratio of path *i* as the number of surface elements ($n_i^s$) covered by the images taken during path $i$ relative to the total number of surface elements of the tower ($N^s$). To compute $n_i^s$, we use three different coordinate systems: (i) real coordinates, (ii) camera coordinates, and (iii) image coordinates. The origin of the real coordinates is the home-base of the drone. The origin of the camera coordinates is the center of the camera lens. The origin of the image coordinate is the center of the image sensor (see Figure 1.6).

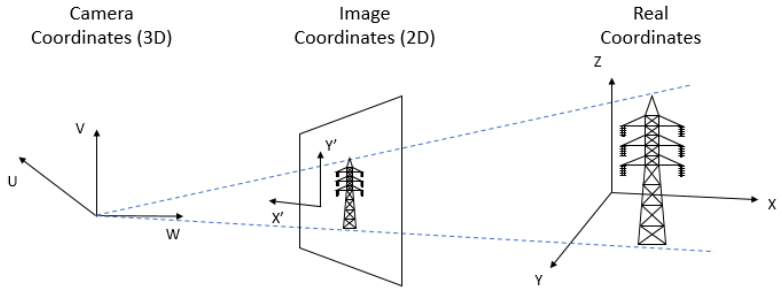



**Figure 1.6** Camera, image, and real coordinate system

#### 1.3.4.1 Transforming real coordinates to camera coordinates

Suppose that $S^{rc}$ is the point in the real coordinate system and that $S^{cc}$ is the corresponding point in the camera coordinate system (see Figure 1.7). Thus, $S^{cc}$ is given by Eq. (1.1) where **TR** is the rigid transformation matrix obtained by multiplying the rotation matrix (**R**) and the translation matrix (**T**) [27].

$$S^{cc} = \mathbf{TR} \cdot S^{rc} = \mathbf{R} \cdot \mathbf{T} \cdot S^{rc} \tag{1.1}$$

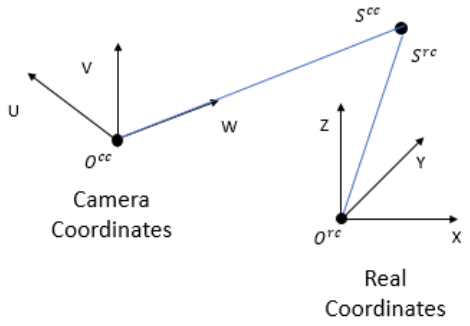



**Figure 1.7** Transforming real coordinates to camera coordinates

Assuming that the center of the camera lens is at the same location as the waypoint $(x_{i,j}, y_{i,j}, z_{i,j})$, the coordinate transformation is given by Eq. (1.2).

$$\begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -x_{i,j} \\ 1 & 0 & 0 & -y_{i,j} \\ 1 & 0 & 0 & -z_{i,j} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{1.2}$$

To compute **R**, we adopt the same technique described in [28]. **R** is an orthogonal matrix that has the following properties: (i) the squares of the elements in any row or column sum to one, and (ii) the rows are the projection of a unit vector along the camera coordinate axes onto the real coordinates.

### 1.3.4.2 Transforming camera coordinates to image coordinates

To map surface elements from camera coordinates to 2D image coordinates, we use the perspective projection method (also known as the pinhole camera model) [28]. In Figure 1.8, the point $S^{cc}$ in camera coordinates is projected by Eqs. (1.3) and (1.4) into the point $S^{ic}$ in image coordinates. Notice that only surface elements in image coordinates located within the image size are retained.

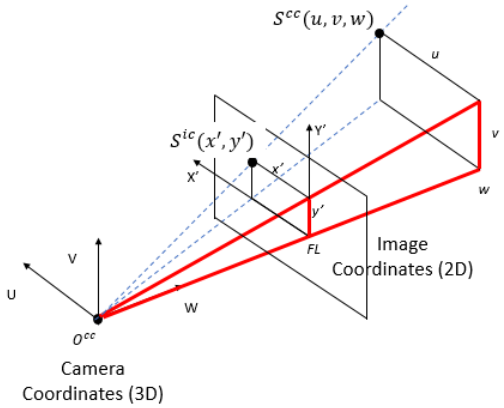



**Figure 1.8** Transforming camera to image coordinates

$$x' = FL\frac{u}{w} \tag{1.3}$$

$$y' = FL\frac{v}{w} \tag{1.4}$$

### 1.3.4.3 Hidden surface elimination problem

The hidden surface elimination problem deals with finding the surface elements that are visible in the image [29]. Surface elements in image coordinates within the image size may not be visible because they may be behind other surface elements (see Figure 1.9). To solve this problem, we use the z-buffer method [30]. When multiple surface elements have the same image coordinates, the surface element with the shortest distance, measured in camera coordinates, to the center of the camera lens is retained. After gathering all visible surface elements in the images from all waypoints, duplicated surface elements are removed. The resulting number of visible unique surface elements for path *i* is denoted by $n_i^s$.

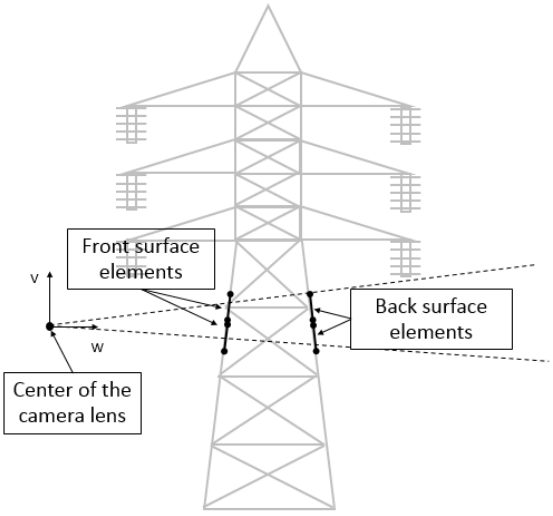



**Figure 1.9** Front and back surface elements

### 1.3.5 Optimization model

The objective of the optimization model is to maximize a total performance ratio ($pr_T$) which is the weighted average of flight time ratio, image quality ratio and tower coverage ratio (Eq. (1.7)). Eq. (1.8) to Eq. (1.18) calculate the ratios. Eq. (1.19) ensures that the drone flies within the flying area. The decision variable is a flight path *i* having *N* waypoints and expressed by Eqs. (1.5) and (1.6).

$$\mathbf{fp}_i = \left(\mathbf{wp}_{i,1}, \mathbf{wp}_{i,2}, \dots, \mathbf{wp}_{i,N}\right) \tag{1.5}$$

where

$$\mathbf{wp}_{i,j} = \left(x_{i,j}, y_{i,j}, z_{i,j}\right) \tag{1.6}$$

The objective function and constraints are given by Eqs. (1.7) – (1.19).

$$\text{Max} \quad pr_T = \omega_1 pr_1 + \omega_2 pr_2 + \omega_3 pr_3 \tag{1.7}$$

Subject to

$$pr_1 = 1 - \frac{ft_i}{FT^{max}} \tag{1.8}$$

$$pr_2 = 1 - \left(\frac{1}{N}\sum_{j=1}^{N} g_{i,j}\right)\left(\frac{1}{G^{max}}\right) \tag{1.9}$$

$$pr_3 = \frac{n_i^s}{N^s} \tag{1.10}$$

$$FT^{max} = \frac{FD^{max}}{FS} + H \times N^{max} \tag{1.11}$$

$$ft_i = \frac{fd_i}{FS} + H \times N \tag{1.12}$$

$$fd_i = wd_i + \sqrt{x_{i,N}^2 + y_{i,N}^2 + z_{i,N}^2} \tag{1.13}$$

$$wd_i = \sum_{j=0}^{N-1} \sqrt{\Delta x_{i,j}^2 + \Delta y_{i,j}^2 + \Delta z_{i,j}^2} \tag{1.14}$$

$$\Delta x_{i,j} = x_{i,j+1} - x_{i,j} \tag{1.15}$$

$$\Delta y_{i,j} = y_{i,j+1} - y_{i,j} \tag{1.16}$$

$$\Delta z_{i,j} = z_{i,j+1} - z_{i,j} \tag{1.17}$$

$$g_{i,j} = \frac{L \times \sqrt{\left(x_{i,j} - X^{tc}\right)^2 + \left(y_{i,j} - Y^{tc}\right)^2}}{N^p \times FL} \quad \forall j \tag{1.18}$$

$$\mathbf{fp}_i \in \Phi_C \tag{1.19}$$

Notice that the objective function of this optimization problem is non-linear, non-differentiable and multimodal. The variables $pr_1$ and $pr_2$ in the objective function are calculated using Eqs. (1.13) and (1.18) which are non-linear equations. The variable $pr_3$ is calculated using Eq. (1.10) in which the variable $n_i^s$ is an integer. The problem is multimodal because multiple local optima coexist.

## 1.4 Solution algorithms

The optimization problem cannot be solved by using a gradient-based optimization method. Thus, we use two common gradient-free algorithms, namely, PSO based-algorithm and SA based-algorithm.

### 1.4.1 PSO-based solution algorithm

The PSO is a global and evolutionary programming algorithm developed from simulating social behaviors such as bird flocking and fish schooling [31]. Specially, this algorithm is easy to implement by adjusting very few parameters and can obtain the optimal or near-optimal solutions within relatively short time [32].

In the standard PSO, a candidate solution, a set of candidate solutions, and the objective function are called particle, swarm, and fitness function, respectively. The algorithm starts by randomly initializing the position and velocity of the particles in the search space. Assuming a *K*-dimensional search space and *M* particles, the position and velocity are described by Eqs. (1.20) and (1.21).

$$\mathbf{P}_i = \left(p_{i,1}, p_{i,2}, \cdots, p_{i,K}\right), \quad \forall i \tag{1.20}$$

$$\mathbf{V}_i = \left(v_{i,1}, v_{i,2}, \cdots, v_{i,K}\right), \quad \forall i \tag{1.21}$$

Particles move to the next position by adjusting their velocity. The velocity is determined by considering three factors: current position, the best previous position of the particle ($\mathbf{PB}_i^t$) at the current time step, and the best previous position of the swarm ($\mathbf{GB}^t$) at the current time step. Eqs. (1.22) and (1.23) show updating $\mathbf{P}_i$ and $\mathbf{V}_i$ at each time step *t*.

$$\mathbf{P}_i^{t+1} = \mathbf{P}_i^t + \mathbf{V}_i^{t+1} \tag{1.22}$$

$$\mathbf{V}_i^{t+1} = W\mathbf{V}_i^t + C_1 r_1(\mathbf{PB}_i^t - \mathbf{P}_i^t) + C_2 r_2(\mathbf{GB}^t - \mathbf{P}_i^t) \tag{1.23}$$

where $W$ is the inertial weight, $C_1$ and $C_2$ the learning factors, and $r_1$ and $r_2$ random numbers between 0 and 1. This updating continues until the stopping criterion is satisfied. The stopping criterion is either the number of iterations or no further improvement in the fitness value between consecutive iterations. The pseudocode of the PSO algorithm is shown in Figure 1.10.

```
For i = 1 to M do
    Initialize P_i^1 and V_i^1
    Compute the fitness value of particle i
    Determine PB_i^1
End for
Determine GB^1
Increase the time step t by 1
While (stop criterion is not met) do
    Update P_i^t and V_i^t using Eqs. (1.22) and (1.23)
    For i = 1 to M do
        Compute the fitness value of particle i
        Update PB_i^t
    End for
    Update GB^t
    Increase the time step t by 1
End while
```

**Figure 1.10** Pseudocode for the PSO

In terms of the communication of the swarm, two models have been proposed: local topology and global topology. While the former allows a particle to exchange information with only proximate neighbors, in the latter all particles share information together. It is not possible to say which model has superior performance. A global topology shows a faster convergence rate than a local topology that escapes from a local optimum. However, a local topology finds a better fitness value than a global topology after some number of iterations for particularly multimodal problems [33].

In our problem, a particle is a candidate flight path $i$. For the communication of the swarm, we use a global topology to avoid obtaining a local optimum. It is possible that particles fly beyond

the flying area $\Phi_C$. To fix this issue, we limit the velocity to $v^{max}$ and use a boundary condition that places the particle back into the area $\Phi_C$. We use the absorbing approach described in [34] to relocate the particle at the boundary of the $\Phi_C$ with zero velocity.

### 1.4.2 SA-based solution algorithm

The SA is a global programming algorithm derived from an analogy with annealing metals and firstly introduced to solve combinatorial optimization problem [35]. In thermodynamics, molecules are free to move at high temperatures, while they are restricted to move at low temperatures. If the temperature is lowered slowly in the cooling process, the molecules form a rigid structure that has the lowest energy state. Similarly, in the minimization problem, cooling gradually enables to find the solution having the lowest objective function value.

Our benchmark is the SA algorithm presented in [36] that is applicable for multimodal functions of continuous variables. As shown in Figure 1.11, the algorithm starts by initializing the temperature ($T$), step vector (**s**), and decision variable vector (**x**). Then, four consecutive loops are executed until the stopping criterion is met.

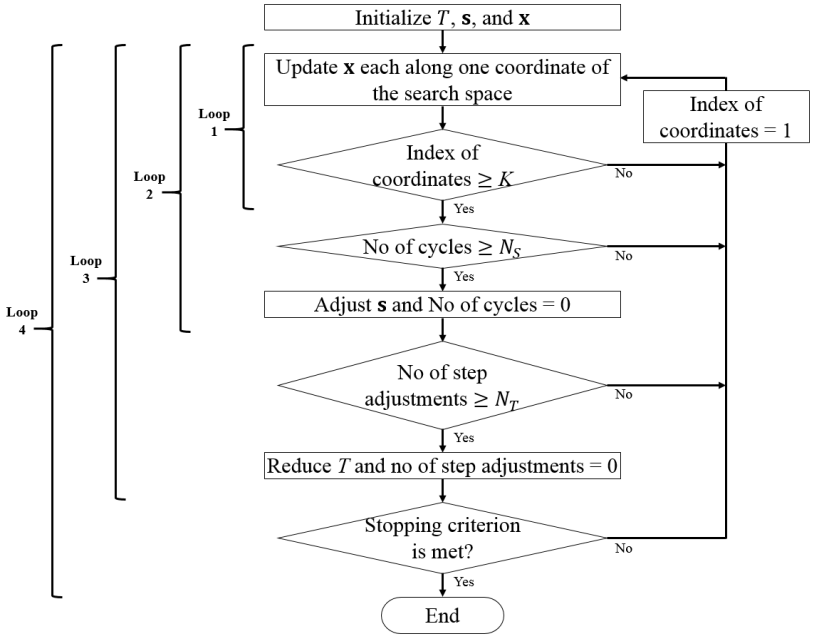



**Figure 1.11** Flowchart for the SA

Loop 1: Random move

Assuming a search space has *K* coordinates, **x** is updated each along one coordinate according to Eq. (1.24).

$$\mathbf{x}' = \mathbf{x} + r \cdot \mathbf{s} \cdot \mathbf{e} \tag{1.24}$$

where $r$ is the random number between 0 and 1, and $\mathbf{e}$ is the unit vector of each coordinate. When $\mathbf{x}'$ violates the boundary constraints, another $\mathbf{x}'$ will be generated until it satisfies the constraints. If the new objective function value ($f(\mathbf{x}')$) is less than current one ($f(\mathbf{x}^*)$), we accept the $\mathbf{x}'$ and record it as the optimum ($\mathbf{x}^*$). If not, we only accept or reject $\mathbf{x}'$ with acceptance probability ($p$) computed by Eq. (1.25).

$$p = \exp\left(-\frac{f(\mathbf{x}') - f(\mathbf{x})}{T}\right) \tag{1.25}$$

This loop is repeated *K* times which is the number of coordinates.

Loop 2: Cycle

This loop includes *K* random moves and one step vector adjustment which is done each along one coordinate according to Eq. (1.26) or (1.27).

$$\mathbf{s} = \times \left( 1 + c \times \frac{m_k / N_S - 0.6}{0.4} \right) \quad \text{For } m_k > 0.6 \times N_S \tag{1.26}$$

$$\mathbf{s} = \frac{\mathbf{s}}{(1 + c \times \frac{0.4 - m_k / N_S}{0.4})} \quad \text{For } m_k < 0.4 \times N_S \tag{1.27}$$

where $m_k$ is the number of accepting $\mathbf{x}'$ in each coordinate and $N_S$ is the number of cycles. The pseudocode of the random move and cycle is illustrated in Figure 1.12.

Loop 3: Step vector adjustment

This loop consists of *K* random moves and $N_S$ cycles. It is repeated $N_T$ times which is the number of step vector adjustments.

Loop 4: Temperature reduction

This loop consists of *K* random moves, $N_S$ cycles, $N_T$ step vector adjustments, and one temperature reduction which is done by Eq. (1.28).

$$T = r_T \times T \tag{1.28}$$

where $r_T$ is the reduction coefficient.

```
For n_cycles = 0 to N_S do
  If (n_cycles < N_S) then
    For k = 1 to K do
      Update x using Eq. (1.24)
      While (x' violates the boundary constraints) do
        Update x using Eq. (1.24)
      End while
      If (f(x') < f(x)) then
        Accept x', x* ← x', m_k ← m_k + 1
      Else
        Compute p using Eq. (1.25)
        If (rand() ≤ p) then
          Accept x', m_k ← m_k + 1
        End if
      End if
    End for
  Else
    For k = 1 to K do
      If (m_k > 0.6 × N_S) then
        Adjust s using Eq. (1.26)
      Else if (m_k < 0.4 × N_S) then
        Adjust s using Eq. (1.27)
      End if
    End for
  End if
End for
```

**Figure 1.12** Pseudocode for the SA

The algorithm is terminated when either two conditions are satisfied: (i) all changes of the last accepted $\mathbf{x}'$ over $N_\varepsilon$ successive temperature reductions are less than ε, and the difference between the last accepted $f(\mathbf{x}')$ and current $f(\mathbf{x}^*)$ is less than ε, or (ii) the number of objective function evaluations is greater than $N_{eval}$.

In our problem, the decision variable vector is a flight path ($\mathbf{fp}$) determined in the search space having $N$ coordinates. To apply the SA algorithm, we transform our problem to a minimization problem by changing the objective function (i.e. Min $1 - pr_T$).

## 1.5 Numerical experiments

All models for the numerical experiments are coded in MATLAB. Each experiment is run ten times on a computer with an Intel® Core™ i3-3110M (3M Cache, 2.40 GHz) processor with

8 GB of RAM memory. Table 1.1 shows the parameters of the drone, camera, transmission tower, and flying area.

**Table 1.1** System parameters

| | |
|---|---|
| Drone specifications | |
| Flying speed ($FS$) | 5 m/s |
| Hovering time to take an image ($H$) | 3 sec |
| Maximum flight time ($FT^{max}$) | 491 sec |
| Maximum flight distance ($FD^{max}$) | 1,856 m |
| Camera specifications | |
| Height of the image sensor ($L$) | 4.6 mm |
| Focal length ($FL$) | 3.6 mm |
| Vertical number of pixels in the image ($N^p$) | 3,000 |
| Transmission tower | |
| Total number of surface elements ($N^s$) | 464 |
| Tower center-base coordinates ($X^{tc}, Y^{tc}, 0$) | (13.72,13.72,0) |
| Flying area | |
| Maximum *x*, *y*, and *z* of the flying area (**D**) | (27.43, 27.43, 36.58) |
| Maximum number of waypoints ($N^{max}$) | 40 |
| Maximum ground sampling distance ($G^{max}$) | 8.2 mm |

### 1.5.1 Inspection strategies

We develop three inspection strategies (S1, S2, S3) for scanning the transmission tower shown in Figure 1.2.

Strategy S1

The flying area is divided into horizontal cuboids. The drone starts to scan from the bottom cuboid and flies through waypoints. Then, the drone continues to scan in the cuboid above until it reaches the top cuboid (see Figure 1.13).

Strategy S2

The flying area is divided into vertical cuboids. The drone starts to scan from the home-base cuboid and continues to scan in the neighboring cuboid until it reaches the first cuboid (see Figure 1.14).

Strategy S3

The flying area is not divided (see Figure 1.15).

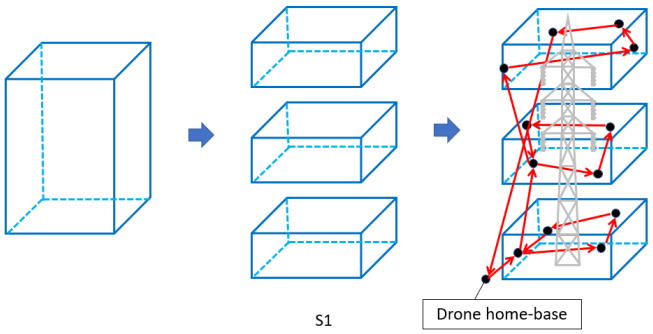



**Figure 1.13** Strategy S1 ($N = 12$)

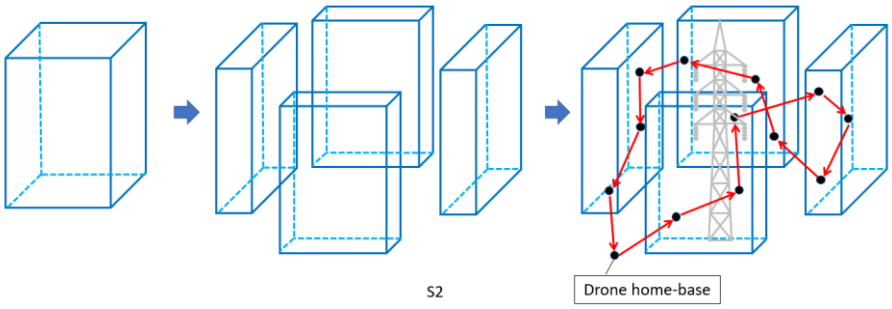



**Figure 1.14** Strategy S2 ($N = 12$)

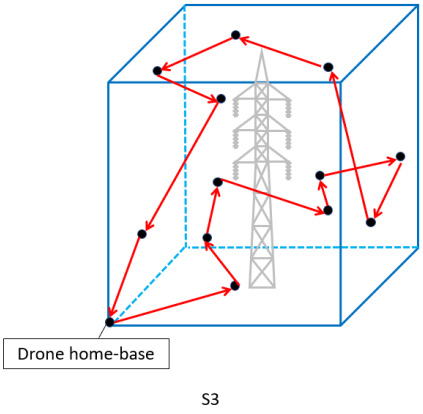



**Figure 1.15** Strategy S3 ($N = 12$)

### 1.5.2 Parameter tuning and stopping criterion

In order to maximize the performance of the algorithms, it is necessary to tune the parameters. For the PSO-based algorithm, we run the inspection-strategy S1 with 16 waypoints

for 20 different combinations of the parameters. The results are given in Table 1.2. The best performance (highest $pr_T$) is obtained with $W$ = 0.6, $C_1$ = 2, and $C_2$ = 1. For the SA-based algorithm, we run the inspection-strategy S1 with 16 waypoints for 9 different combinations of the initial temperature $T_0$ and reduction coefficient $r_T$ parameters, keeping other parameters ($N_S = 20$, $N_T = 20$, $N_\varepsilon = 4$, and $c = 2$) constant. The results are given in Table 1.3. The best performance (highest $pr_T$) is obtained with $T_0$ = 0.03 and $r_T$ = 0.5. In terms of the stopping criterion, we use 50 number of iterations for the PSO-based algorithm, and $\varepsilon = 0.001$ and $N_{eval} = 10{,}000$ for the SA-based algorithm.

**Table 1.2** PSO parameter tuning results

| $W$ | $C_1$ | $C_2$ | $pr_T$ | $pr_1$ | $pr_2$ | $pr_3$ |
|---|---|---|---|---|---|---|
| 0.3 | 1 | 1 | 0.83 | 0.81 | 0.69 | 0.98 |
| | 1 | 2 | 0.83 | 0.82 | 0.70 | 0.98 |
| | 2 | 1 | 0.83 | 0.82 | 0.70 | 0.99 |
| | 2 | 2 | 0.84 | 0.82 | 0.72 | 0.98 |
| 0.6 | 1 | 1 | 0.84 | 0.81 | 0.71 | 0.99 |
| | 1 | 2 | 0.84 | 0.82 | 0.72 | 0.98 |
| | 2 | 1 | 0.84 | 0.82 | 0.73 | 0.99 |
| | 2 | 2 | 0.83 | 0.82 | 0.70 | 0.98 |
| 0.9 | 1 | 1 | 0.83 | 0.81 | 0.70 | 0.98 |
| | 1 | 2 | 0.81 | 0.81 | 0.65 | 0.98 |
| | 2 | 1 | 0.82 | 0.81 | 0.67 | 0.98 |
| | 2 | 2 | 0.80 | 0.80 | 0.62 | 0.97 |
| 1.2 | 1 | 1 | 0.80 | 0.80 | 0.61 | 0.97 |
| | 1 | 2 | 0.79 | 0.80 | 0.59 | 0.98 |
| | 2 | 1 | 0.80 | 0.80 | 0.63 | 0.97 |
| | 2 | 2 | 0.79 | 0.80 | 0.59 | 0.97 |
| 1.5 | 1 | 1 | 0.79 | 0.80 | 0.60 | 0.96 |
| | 1 | 2 | 0.78 | 0.79 | 0.57 | 0.97 |
| | 2 | 1 | 0.79 | 0.80 | 0.60 | 0.97 |
| | 2 | 2 | 0.78 | 0.79 | 0.58 | 0.98 |

**Table 1.3** SA parameter tuning results

| $T_0$ | $r_T$ | $pr_T$ | $pr_1$ | $pr_2$ | $pr_3$ |
|---|---|---|---|---|---|
| 0.01 | 0.50 | 0.74 | 0.79 | 0.48 | 0.94 |
| | 0.70 | 0.73 | 0.79 | 0.47 | 0.93 |
| | 0.85 | 0.73 | 0.79 | 0.48 | 0.93 |
| 0.02 | 0.50 | 0.73 | 0.77 | 0.47 | 0.95 |
| | 0.70 | 0.74 | 0.78 | 0.47 | 0.96 |
| | 0.85 | 0.73 | 0.78 | 0.46 | 0.94 |
| 0.03 | 0.50 | 0.74 | 0.78 | 0.48 | 0.97 |
| | 0.70 | 0.73 | 0.78 | 0.46 | 0.95 |
| | 0.85 | 0.72 | 0.77 | 0.44 | 0.95 |

### 1.5.3 Experimental results

We compare the performance ratios under the previously defined three inspection strategies for the PSO-based and SA-based algorithms. For all experiments, we obtain the best/average/worst performance ratios, coefficient of variation (CV), and the average computational time over ten replicates. We measure the computational time by using the "tic" and "toc" MATLAB functions.

#### 1.5.3.1 Results under equal weight values

This section provides the results when we use equal weight values of the objective function ($\omega_1 = \omega_2 = \omega_3 = 1/3$). Using the same number of waypoints ($N$ = 12), Table 1.4 shows that the strategies S1, S2, and S3 with the PSO-based algorithm surpass those with the SA-based algorithm in terms of the average total performance ratio. The image quality ratio is the main factor that causes this difference on the average total performance ratios. We find that the S1-PSO is the most efficient flight path because it produces the highest average total performance ratio and shortest computational time.

**Table 1.4** Performance results over 3 strategies with different algorithms ($N$ = 12)

| | $pr_T$ | | | | $pr_1$ | | $pr_2$ | | $pr_3$ | | Time (min) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Best | Avg. | Worst | CV(%) | Avg. | CV(%) | Avg. | CV(%) | Avg. | CV(%) | |
| S1-PSO | 0.85 | 0.84 | 0.83 | 0.66% | 0.85 | 0.47% | 0.70 | 2.67% | 0.98 | 0.63% | 4.61 |
| S2-PSO | 0.77 | 0.77 | 0.75 | 0.68% | 0.83 | 1.06% | 0.48 | 2.30% | 0.98 | 0.49% | 15.87 |
| S3-PSO | 0.82 | 0.81 | 0.81 | 0.44% | 0.83 | 1.39% | 0.62 | 1.75% | 0.98 | 0.50% | 48.33 |
| S1-SA | 0.76 | 0.73 | 0.70 | 2.44% | 0.84 | 0.85% | 0.43 | 5.64% | 0.92 | 4.96% | 21.49 |
| S2-SA | 0.73 | 0.69 | 0.62 | 4.64% | 0.84 | 1.03% | 0.44 | 9.59% | 0.79 | 14.19% | 55.26 |
| S3-SA | 0.79 | 0.78 | 0.77 | 0.97% | 0.83 | 0.82% | 0.54 | 2.35% | 0.96 | 1.39% | 268.82 |

We test the strategies S1 and S2 by changing the number of waypoints $N$. Table 1.5 gives detailed results and Figure 1.16 shows the average total performance ratio for different numbers of waypoints. As shown in this plot, the average total performance ratio improves as $N$ increases, but decreases when $N$ > 16 for the S1-PSO, S2-PSO and S1-SA, and $N$ > 28 for the S2-SA. The

reason is that a larger number of waypoints requires a longer hovering time, which in turn, deteriorates the flight time ratio. We also observe that a larger number of waypoints increases the image quality ratio when we use the PSO-based algorithm.

**Table 1.5** Performance results for different numbers of waypoints

| | $N$ | $pr_T$ | | | | $pr_1$ | | $pr_2$ | | $pr_3$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Best | Avg. | Worst | CV(%) | Avg. | CV(%) | Avg. | CV(%) | Avg. | CV(%) |
| S1-PSO | 8 | 0.83 | 0.81 | 0.80 | 0.70% | 0.88 | 0.65% | 0.59 | 2.15% | 0.97 | 1.44% |
| | 12 | 0.85 | 0.84 | 0.83 | 0.66% | 0.85 | 0.47% | 0.70 | 2.67% | 0.98 | 0.63% |
| | 16 | 0.85 | 0.84 | 0.84 | 0.46% | 0.82 | 0.63% | 0.73 | 1.38% | 0.99 | 0.44% |
| | 20 | 0.84 | 0.84 | 0.83 | 0.35% | 0.79 | 0.53% | 0.74 | 0.93% | 0.98 | 0.55% |
| | 24 | 0.84 | 0.84 | 0.83 | 0.50% | 0.76 | 0.88% | 0.76 | 1.18% | 0.99 | 0.27% |
| | 28 | 0.83 | 0.83 | 0.82 | 0.18% | 0.72 | 0.61% | 0.77 | 0.63% | 0.98 | 0.41% |
| | 32 | 0.82 | 0.82 | 0.81 | 0.48% | 0.69 | 0.52% | 0.79 | 1.20% | 0.98 | 0.52% |
| S2-PSO | 8 | 0.76 | 0.73 | 0.72 | 1.70% | 0.88 | 1.11% | 0.43 | 8.86% | 0.90 | 3.78% |
| | 12 | 0.77 | 0.77 | 0.75 | 0.68% | 0.83 | 1.06% | 0.48 | 2.30% | 0.98 | 0.49% |
| | 16 | 0.78 | 0.77 | 0.75 | 1.05% | 0.79 | 0.85% | 0.53 | 4.30% | 0.99 | 0.31% |
| | 20 | 0.76 | 0.75 | 0.75 | 0.35% | 0.74 | 0.96% | 0.54 | 1.83% | 0.99 | 0.19% |
| | 24 | 0.76 | 0.75 | 0.73 | 1.06% | 0.69 | 2.06% | 0.56 | 3.07% | 0.99 | 0.18% |
| | 28 | 0.75 | 0.74 | 0.72 | 1.03% | 0.66 | 2.94% | 0.56 | 2.11% | 0.99 | 0.19% |
| | 32 | 0.73 | 0.73 | 0.72 | 0.74% | 0.62 | 2.02% | 0.58 | 1.82% | 0.99 | 0.26% |
| S1-SA | 8 | 0.74 | 0.72 | 0.68 | 2.29% | 0.84 | 0.85% | 0.43 | 5.64% | 0.92 | 4.96% |
| | 12 | 0.76 | 0.73 | 0.70 | 2.44% | 0.84 | 0.85% | 0.43 | 5.64% | 0.92 | 4.96% |
| | 16 | 0.76 | 0.74 | 0.71 | 1.90% | 0.78 | 0.95% | 0.48 | 7.47% | 0.97 | 1.30% |
| | 20 | 0.74 | 0.73 | 0.72 | 1.04% | 0.73 | 1.16% | 0.49 | 6.96% | 0.96 | 2.04% |
| | 24 | 0.72 | 0.71 | 0.68 | 1.78% | 0.68 | 2.31% | 0.47 | 8.61% | 0.96 | 1.93% |
| | 28 | 0.71 | 0.70 | 0.69 | 1.32% | 0.64 | 2.74% | 0.48 | 5.00% | 0.97 | 3.07% |
| | 32 | 0.70 | 0.69 | 0.67 | 1.25% | 0.58 | 2.16% | 0.50 | 4.94% | 0.98 | 1.26% |
| S2-SA | 8 | 0.68 | 0.66 | 0.63 | 2.18% | 0.90 | 0.49% | 0.44 | 13.27% | 0.64 | 15.36% |
| | 12 | 0.73 | 0.69 | 0.62 | 4.64% | 0.84 | 1.03% | 0.44 | 9.59% | 0.79 | 14.19% |
| | 16 | 0.72 | 0.69 | 0.66 | 2.87% | 0.78 | 1.38% | 0.87 | 8.15% | 0.43 | 7.00% |
| | 20 | 0.73 | 0.70 | 0.68 | 2.64% | 0.74 | 1.81% | 0.46 | 8.45% | 0.90 | 7.21% |
| | 24 | 0.72 | 0.68 | 0.65 | 3.42% | 0.69 | 2.41% | 0.47 | 7.41% | 0.90 | 7.00% |
| | 28 | 0.70 | 0.67 | 0.65 | 2.81% | 0.63 | 2.43% | 0.45 | 4.84% | 0.93 | 6.68% |
| | 32 | 0.67 | 0.66 | 0.64 | 1.48% | 0.58 | 3.84% | 0.46 | 6.40% | 0.93 | 5.46% |

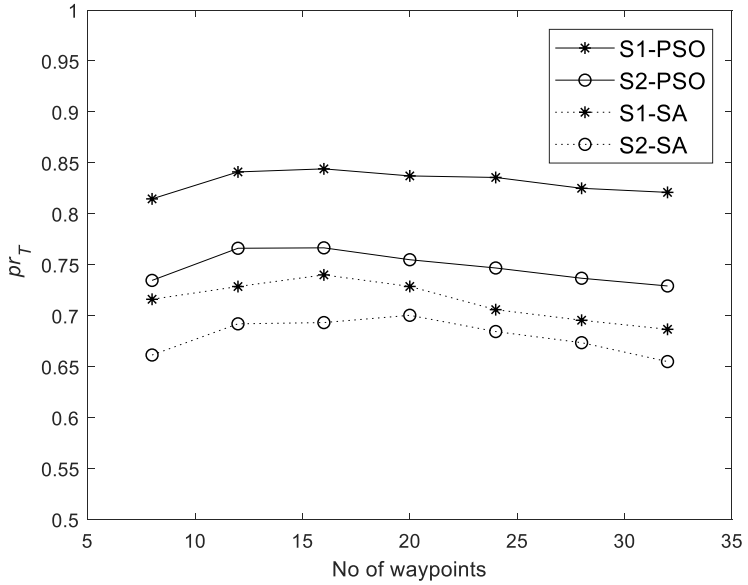



**Figure 1.16** Average total performance ratios for different numbers of waypoints

We study the effect of the swarm size on the performance ratios of the S1-PSO with 16 waypoints. In Table 1.6, we show the results for different swarm sizes. The results indicate that increasing the number of particles (particularly greater than 20 particles) does not affect significantly the performance ratios. However, having more particles requires a longer computational time. This observation supports the argument given in [33] that states that a swarm size between 20 to 100 particles yields substantially no superior results.

**Table 1.6** Performance results for different swarm sizes (S1-PSO with $N$ = 16)

| Swarm size | $pr_T$ | | | | $pr_1$ | | $pr_2$ | | $pr_3$ | | Time (min) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Best | Avg. | Worst | CV(%) | Avg. | CV(%) | Avg. | CV(%) | Avg. | CV(%) | |
| 10 | 0.842 | 0.836 | 0.827 | 0.59% | 0.817 | 0.33% | 0.704 | 1.55% | 0.986 | 0.41% | 1.81 |
| 20 | 0.851 | 0.844 | 0.840 | 0.46% | 0.819 | 0.63% | 0.726 | 1.38% | 0.987 | 0.44% | 3.43 |
| 30 | 0.850 | 0.845 | 0.838 | 0.52% | 0.819 | 0.71% | 0.734 | 1.59% | 0.981 | 0.82% | 5.35 |
| 40 | 0.854 | 0.845 | 0.836 | 0.66% | 0.820 | 0.40% | 0.732 | 2.36% | 0.983 | 0.93% | 7.01 |
| 50 | 0.851 | 0.848 | 0.843 | 0.33% | 0.820 | 0.46% | 0.737 | 0.78% | 0.986 | 0.45% | 8.54 |

From Tables 5 and 6 we create Figure 1.17 to illustrate the waypoints of the flight path for the best solution from S1-PSO with 16 waypoints obtained using a swarm size of 50. Notice that

the best solution of the PSO-based algorithm (S1-PSO) improves the average total performance ratio of the best solution of the SA-based algorithm (S3-SA) by 9 percent.

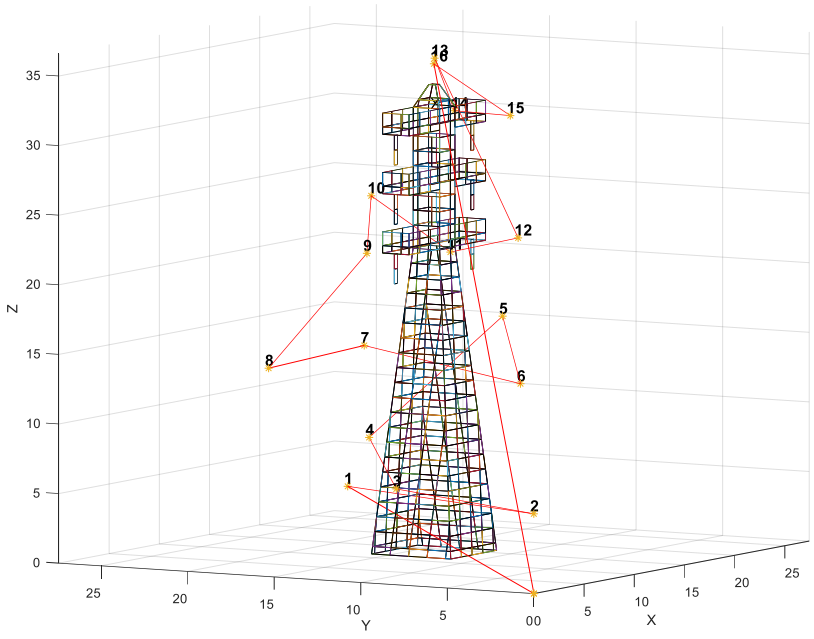



**Figure 1.17** Flight path for the best solution

### 1.5.3.2 Results under different weight values

We examine the average total performance ratios under different weights of the objective function by changing one weight while keeping the other two weights at the same value. For example, when we change the first weight, the weight vector **Ω** will be [0 1/2 1/2], [1/4 3/8 3/8], [1/2 1/4 1/4], [3/4 1/8 1/8], and [1 0 0]. We solve the model for the S1-PSO, S2-PSO, S1-SA, and S2-SA with 12 waypoints. Figures 1.18, 1.19, and 1.20 illustrate the average total performance ratios as the $\omega_1$, $\omega_2$, and $\omega_3$ are changed respectively. Overall, the PSO-based algorithm produces higher average total performance ratios than the SA-based algorithm except when **Ω** = [1 0 0].

Also, the strategy 1 produces higher average total performance ratios than the strategy 2 except when $\boldsymbol{\Omega}$ = [0 1 0] and $\boldsymbol{\Omega}$ = [0 1 1] and we use the SA-based algorithm. The average total performance ratios are generally increased as we increase $\omega_1$ and $\omega_3$, but the ratios are decreased as we increase $\omega_2$. When only the tower coverage ratio is maximized (i.e. $\boldsymbol{\Omega}$ = [0 0 1]), we achieve almost 100 percent coverage of the tower. However, when only the flight time ratio or image quality ratio are maximized (i.e. $\boldsymbol{\Omega}$ = [1 0 0] or $\boldsymbol{\Omega}$ = [0 1 0]), we do not achieve 100 percent on these ratios. The 100 percent flight time ratio corresponds to a zero flight-time. Similarly, a 100 percent image quality ratio corresponds to all waypoints located at the surface of the transmission tower creating a path collision risk.

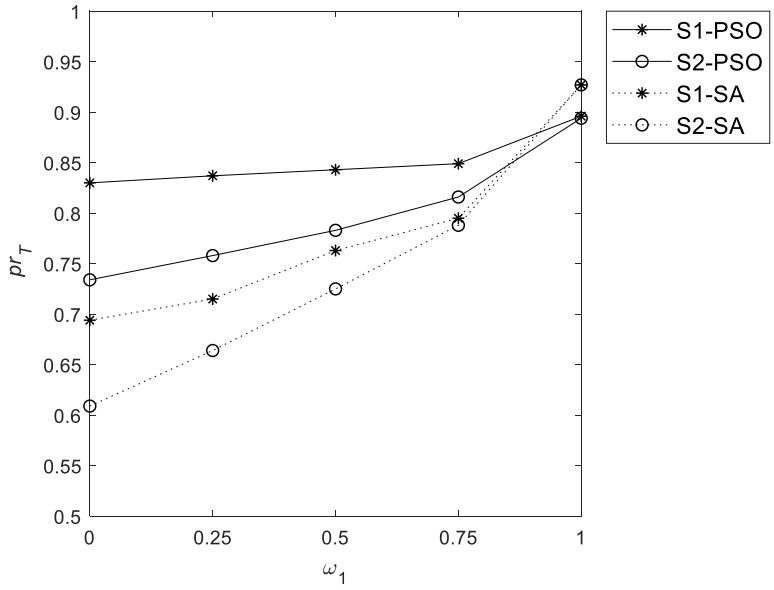



**Figure 1.18** Average total performance ratios for different weights of the fitness function by changing $\omega_1$

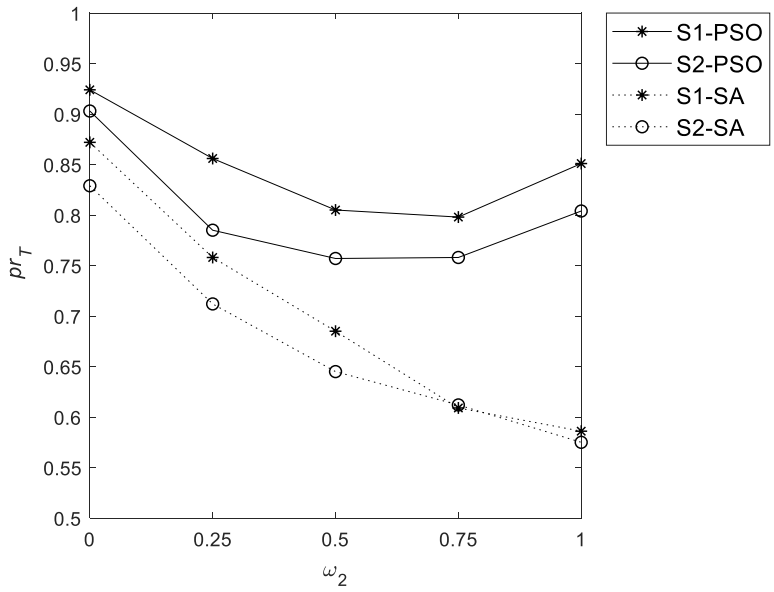



**Figure 1.19** Average total performance ratios for different weights of the fitness function by changing $\omega_2$

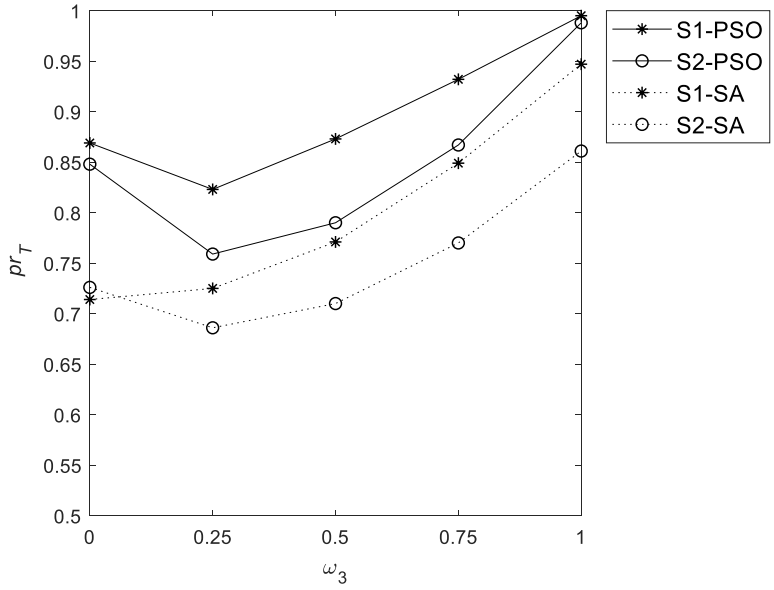



**Figure 1.20** Average total performance ratios for different weights of the fitness function by changing $\omega_3$

## 1.6 Conclusion

We proposed a PSO-based algorithm to find an efficient drone flight path for inspecting electric transmission towers. Our experimental results showed that the PSO-based algorithm outperformed the SA-based algorithm in terms of the average total performance ratio and computational time. When the flying area is divided horizontally and the waypoints are determined using the PSO-based algorithm, the average total performance ratio and computational time were superior. One advantage of the PSO-based algorithm is that it can provide multiple best solutions. Under equal weight values of the objective function and a swarm size of 20, we obtained two best solutions with the same total performance ratio ($pr_T$ = 0.84), a solution A with 12 waypoints and performance ratios $pr_1$ = 0.85, $pr_2$ = 0.70, $pr_3$ = 0.98 and a solution B with 16 waypoints and performance ratios $pr_1$ = 0.82, $pr_2$ = 0.73, $pr_3$ = 0.99. The solution A would be preferable if fewer waypoints were required to reduce complexity of the navigation and if reducing flight time was more critical than obtaining images with higher quality. However, the solution B would be preferable if higher tower coverage and higher image quality were more critical than navigation time. We also noticed that the swarm size did not affect significantly the quality of the solutions.

When we maximized the flight time ratio only, we obtained a flight path that provided 93% of flight time ratio. But, the image quality and tower coverage ratios were low, 20% and 0% respectively. Similarly, by maximizing the image quality ratio, we obtained a flight path that provided 85% of image quality ratio. But, the flight time and tower coverage ratios were 87% and 10%, respectively. By maximizing the tower coverage, we obtained a flight path that covered the tower 99%. But, the flight time and image quality ratios were 81% and 31%, respectively. By maximizing the three performance ratios simultaneously, we obtained a flight path that provided flight time, image quality, and tower coverage ratios of 85%, 70%, and 98%, respectively. These

results show that the use of the proposed model can provide a flight path that comprises a good balance over the three performance ratios.

## References


[1] D. Common, “Drones go commercial, take on tasks from industry to farming,” *CBC News*, May 30, 2014. [Online]. Available: https://www.cbc.ca/news/technology/drones-go-commercial-take-on-tasks-from-industry-to-arming-1.2657036?cmp=rss. [Accessed Dec. 10, 2017].

[2] M. Marshall, “Aerial infrared line inspection,” in *Proceedings of the Rural Electric Power Conference*, 1999, pp. A3/1–A3/2.

[3] U.S. Department of Labor, Bureau of Labor Statistics, “National census of fatal occupation injuries in 2017,” 2018. [Online]. Available: https://www.bls.gov/news.release/pdf/cfoi.pdf. [Accessed May 24, 2019].

[4] L. L. Grigsby, *Electric power generation, transmission, and distribution*, 3rd ed. Boca Raton, FL: CRC Press, 2012, pp. 12.10-12.14.

[5] S. Montambault and N. Pouliot, “About the future of power line robotics,” in *Proceedings of the International Conference on Applied Robotics for the Power Industry*, 2010.

[6] U.S. Federal Aviation Administration, *Unmanned Aircraft Systems*, 2017. [Online]. Available: https://www.faa.gov/uas. [Accessed Dec. 10, 2017].

[7] D. Jones, “Power line inspection – a UAV concept,” in *Proceedings of the IEE Forum on Autonomous Systems*, 2005.

[8] M. Williams, D. I. Jones, G. K. Earp, “Obstacle avoidance during aerial inspection of power lines,” *Aircraft Engineering and Aerospace Technology*, 73(5), pp. 472-479, 2001.

[9] Electric Power Research Institute, "Future inspection of overhead transmission lines," Technical update report 1016921, 2008. [Online]. Available: http://mydocs.epri.com/docs/corporatedocuments/sectorpages/pdu/sensorsrobots/Tline%20Roadmap.pdf. [Accessed Dec. 10, 2017].

[10] Electric Power Research Institute, "Emerging and future inspection of overhead transmission lines," Technical update report 1021876, 2011.

[11] U. Wang, "A new weapon for storm responders: send in the drones," *Electric Power Research Institute Journal*, 2, pp 7-9 2012.

[12] N. Zhou, "Study and application of power transmission line patrol system based on microwave communication," in *Proceedings of the International Conference on Measuring Technology and Mechatronics Automation*, 2015, pp. 1036-1039.

[13] G. Dong, X. Chen, B. Wang, J. Zhang, L. Liu, Q. Wang, and C. Wei, "Inspecting transmission lines with an unmanned fixed-wings aircraft," in *Proceedings of the International Conference on Applied Robotics for the Power Industry*, 2012, pp. 173-174.

[14] L. F. Luque-Vega, B. Castillo-Toledo, A. Loukianov, and L. E. Gonzalez-Jimenez, "Power line inspection via an unmanned aerial system based on the quadrotor helicopter," in *Proceedings of the IEEE Mediterranean Electrotechnical Conference*, 2014, pp. 393-397.

[15] C. Deng, S. Wang, Z. Huang, Z. Tan, and J. Liu, "Unmanned aerial vehicles for power line inspection: a cooperative way in platforms and communications," *Journal of Communications*, 9(9), pp. 687-692, 2014.

[16] J. Zhang, L. Liu, B. Wang, X. Chen, Q. Wang, and T. Zheng, "High speed automatic power line detection and tracking for a UAV-based inspection," in *Proceedings of the International Conference on Industrial Control and Electronics Engineering*, 2012, pp. 266-269.

[17] J. I. Larrauri, G. Sorrosal, and M. Gonzalez, "Automatic system for overhead power line inspection using an unmanned aerial vehicle-RELIFO project," in *Proceedings of the International Conference on Unmanned Aircraft Systems*, 2013, pp. 244-252.

[18] S. Montambault, J. Beaudry, K. Toussaint, and N. Pouliot, "On the application of VTOL UAVs to the inspection of power utility assets," in *Proceedings of the International Conference on Applied Robotics for the Power Industry*, 2010.

[19] A. Pagnano, M. Hopf, and R. Teti, "A roadmap for automated power line inspection. Maintenance and repair," *Procedia CIRP*. 12, pp. 234-239, 2013.

[20] J. Katrasnik, F. Pernus, and B. Likar, "A survey of mobile robots for distribution power line inspection," *IEEE Transaction on Power Delivery*, 25(1), pp. 485-493 2010.

[21] U.S. Federal Aviation Administration, "Summary of small unmanned aircraft rule (part 107)," 2016. [Online]. Available: https://www.faa.gov/uas/media/Part_107_Summary.pdf. [Accessed Dec. 10, 2017].

[22] J. A. Guerrero and Y. Bestaoui, "UAV path planning for structure inspection in windy environments," *Journal of Intelligent & Robotic Systems*, 69(1-4), pp. 297-311, 2013.

[23] C. D. Franco and G. Buttazzo, "Coverage path planning for UAVs photogrammetry with energy and resolution constraints," *Journal of Intelligent & Robotic Systems*, 83(3-4), pp. 445-462, 2016.

[24] P. B. Sujit, B. P. Hudzietz, and S. Saripalli, "Route planning for angle constrained terrain mapping using an unmanned aerial vehicle," *Journal of Intelligent & Robotic Systems*, 69(1-4), pp. 273-283, 2013.

[25] C. A. Belhadj, M. M. Dawoud, N. Maalej, I. O. Habiballah, and T. K. Abdel-Galil, "Electric & magnetic field assessment for live-line workers next to A 132 KV transmission line

conductor," in *Proceedings of the IEEE/PES Transmission and Distribution Conference and Exposition: Latin America*, 2008.

[26] G. Duveiller and P. Defourny, "A conceptual framework to define the spatial resolution requirements for agricultural monitoring using remote sensing," *Remote Sensing of Environment*, 114, pp. 2637-2650, 2010.

[27] T. Zsedrovits, P. Bauer, A. Hiba, M. Nemeth, B. J. M. Pencz, A. Zarandy, B. Vanek, and J. Bokor, "Performance analysis of Camera rotation estimation algorithms in multi-sensor fusion for unmanned aircraft attitude estimation," *Journal of Intelligent & Robotic Systems*, 84(1-4), pp. 759-777, 2016.

[28] D. Wang, K. B. Lim, and W. L. Kee, "Geometrical approach for rectification of single-lens stereovision system with a triprism," *Machine Vision and Applications*, 24, pp. 821-833, 2013.

[29] E. E. Catmull, "A subdivision algorithm for computer display of curved surfaces," Ph.D. dissertation, Department of Computer Science, University of Utah, Salt Lake City, UT, 1974.

[30] J. F. Hughes, A. V. Dam, M. Mcguire, D. F. Sklar, J. D. Foley, S. K. Feiner, and K. Akeley, *Computer Graphics - Principles and Practice*, 3rd ed. Addison-Wesley, 2013.

[31] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, 1995, pp. 1942-1948.

[32] Y. Dong, J. Tang, B. Xu, and D. Wang, "An application of swarm optimization to nonlinear programming," *Computers and Mathematics with Applications*, 49, pp. 1655-1668, 2005.

[33] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization," in *Proceedings of the IEEE Swarm Intelligence System*, 2007, pp. 120-127.

[34] S. Xu and Y. Rahmat-Samii, "Boundary conditions in particle swarm optimization revisited," *IEEE Transactions on Antennas and Propagation*, 55(3), pp. 760-765, 2007.

[35] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, “Optimization by simulated annealing,” *Science*, 220, pp. 671-680, 1983.

[36] A. Corana, M. Marchesi, C. Martini, and S. Ridella, “Minimizing multimodal functions of continuous variables with the “simulated annealing” algorithm,” *ACM Transactions on Mathematical Software*, 13(3), pp. 262-280, 1987.

# Chapter 2 Drone Routing for Inspecting Wind Farms

## 2.1 Abstract

The use of wind turbines to generate electricity is growing worldwide. They are being installed in large numbers and built in remote locations on land and sea and far from cities. They comprise an extended area of hundreds of square miles, making the inspection process difficult and time consuming. Recently, there has been an increasing interest in using a drone, or also known as unmanned aircraft system (UAS), for inspecting wind turbines. In this chapter, we develop a routing optimization model to reduce the total operation time for inspecting a wind farm. We assume that one drone and one ground vehicle which carries the drone and extra batteries and charging equipment are available. The optimization model is solved in two steps. The first step clusters the wind turbines and optimizes the drone routing in each cluster by solving the classical traveling salesman problem (TSP) using an integer linear programming model. The second step optimizes the ground vehicle routing by solving the equality generalized traveling salesman problem (E-GTSP) using an integer linear programming model. We test our approach using three case studies created by using actual wind farm locations. The results show that the proposed model reduces the total operation time consistently compared to no clustering and using a greedy approach for determining the route of the ground vehicle.

**Keywords:** Drone, Unmanned aircraft system, Wind turbine inspection routing, Integer linear programming, Traveling salesman problem, Equality generalized traveling salesman problem

## 2.2 Introduction

According to the U.S. Energy Information Administration, on a percentage basis, renewables are the fastest growing energy source [1]. Wind farms are being built in large numbers on land and offshore, and usually comprise an extended area of hundreds of square miles. To identify potential problems to their proper operation, wind turbines are inspected on a regular basis. The most common inspection practice is a visual inspection by a technician using rope access. This technique is cost effective and offers flexible access to the structure for performing the maintenance and repair tasks. However, this practice has a very low inspection rate, in the range of 2 to 5 turbines per day [2]. Recently, there has been increased interest in using a drone for inspecting wind turbines. SkySpecs, a Michigan-based startup, can inspect a wind turbine in 15 minutes using a drone, covering 17 wind turbines per day [3].

Several recently published papers addressed the challenge of using a drone to inspect the structure of a wind turbine. In [4], the authors presented an algorithm that uses the Hough transform and Kalman filter for a camera-based autonomous navigation of a drone. In related research, [5] used a light detection sensor and a global positioning system to ensure an automated collision-free flight for an inspection drone. [6] equipped a drone with an infrared thermography camera to visually inspect a wind turbine. For inspecting offshore wind farms, [7] used an unmanned surface vessel to shuttle an inspection drone and recharge the drone's batteries.

In this chapter, our approach is to inspect several wind turbines using one drone and one ground vehicle which carries extra batteries and the battery-charging equipment. It is assumed that

the ground vehicle carrying the drone departs from a depot. The ground vehicle stops at pre-defined locations. At each stop, the drone flies to inspect several wind turbines then returns to the ground vehicle. In the ground vehicle, the operator swaps the batteries and charges used batteries. The ground vehicle moves to another location and the inspection process is repeated.

The combination of a drone and ground vehicle has been successfully used in parcel delivery. In this case, parcels are delivered directly to customers by either the drone or vehicle. The research problem has been in planning an optimal route for the drone and vehicle as well as assigning parcels to be delivered by either the drone or vehicle. [8] proposed two different optimization problems by considering two sets of assumptions. Each problem was formulated as a mixed integer linear programming (MILP) problem with the objective to reduce the total delivery time. Because these optimization problems are NP-hard, the authors developed two heuristic solution approaches and tested them assuming 10 and 20 customers. These optimization problems were extended by several researchers [9 – 13]. In [9], the optimization problem was solved using a SA metaheuristic. In [10], the authors used an objective function to minimize operational costs. The proposed a more effective heuristic called greedy randomized adaptive search procedure (GRASP) and tested it assuming 10, 50, and 100 customers. In [11], the optimization problem was solved using the randomized variable neighborhood descent heuristic. [12] considered a drone station defined as a facility that stores drone and charging equipment. The problem was formulated using a mixed integer programming (MIP) and solved by dividing it into a traveling salesman problem (TSP) and a parallel identical machine scheduling problem. [13] assumed that the drone can depart and return while the vehicle is moving. They solved the problem by using a heuristic algorithm which is a modified version of the GRASP method proposed in [10]. Also, [14] proposed a mixed-integer, second-order conic programming formulation and a ranking-based solution

algorithm to provide an exact solution for the routings of the general carrier (e.g. truck)-vehicle (e.g. drone) systems.

Multiple variations of the parcel delivery problem using drones exist in the literature. In [15], the authors assumed that the drone is constrained to use the same road network as the vehicle when delivering the parcels. The authors developed an integer programming formulation and several heuristic algorithms based on local search and dynamic programming. [16] extended this research by finding an exact solution to the dynamic programming formulation and developing two dynamic programming-based heuristics for large instances. Using the same assumptions on the take-off and landing stop locations described in [15], [17] derived six subproblems to optimize the schedule of drones based on a given truck route. They analyzed the computational complexity of each subproblem and developed efficient MIP models. [18] addressed the optimization problem on finding the locations of the drone charging stations to maximize the customer demand. They proposed a MIP formulation and a heuristic algorithm based on minimum spanning trees, greedy subtraction, spatial interchange, and simulated annealing (SA) algorithms.

Moreover, [19 – 21] assumed that only the drone delivers the parcels while the driver just drives the vehicle. [19, 20] determined the stop locations of the ground vehicle by formulating the optimization problem as a TSP which is combined with the K-means clustering algorithm. To solve the TSP, a genetic algorithm (GA) was used in [19] and a nonlinear programming formulation with a commercial solver was used in [20]. In [21], the aim was to reduce the delivery time or fuel consumption. The optimization problem is solved by using the Lin-Kernighan-Helsgaun (LKH) heuristic after transforming the problem into a generalized TSP which in turn is transformed into a standard TSP. [22] did not consider a ground vehicle. They formulated an optimization model for routing multiple drones with the two objectives of minimizing the overall

delivery time of the parcels and total costs of drones. Their optimization model was a MILP formulation that was solved by using a SA algorithm. [23] found upper bounds for the time savings when using multiple drones and vehicles compared to using only one vehicle. In [24], they extended this research by considering the operation cost and limited battery capacity, and using other variants of the routing problem.

The combination of a drone and ground vehicle has also been explored in humanitarian logistics. [25] assumed that the drone is used to deliver emergency supplies when the truck roads are destroyed due to a natural disaster. They formulated a MIP for determining the locations of the distribution centers, their corresponding service areas and three ordering quantities of the emergency supplies. The objective was to minimize the total operation cost. The problem was solved using a continuous approximation technique. [26], that did not consider a ground vehicle, proposed a MILP formulation for routing a drone to minimize the total operation cost. They considered recharging stations and priority of outbreak locations.

Surveillance is another area where drones and vehicles are used together. The drone patrols certain targets and the vehicle carries personnel and the drone. [27] proposed an integer linear program and used two heuristic algorithms to solve the model based on the rules “drone first, ground vehicle second” and “ground vehicle first, drone second”, respectively. [28] assumed that the drone is operated to visit targets with multiple fixed depots for refueling. The objective function aims to minimize the total fuel consumption. These authors formulated a MILP and solved it using two heuristic algorithms based on the LKH and *k*-opt heuristics, respectively. In [29], they presented two MILP problems for either multiple ground vehicles or multiple drones and developed a branch-and-cut solution algorithm. [30] solved the same optimization problem using

a GA algorithm. For more applications on using the combination of a drone and vehicle, the interested reader is referred to the recent survey by [31].

The aim of this chapter is to optimize the routing of a drone and a ground vehicle for inspecting wind turbines while minimizing the total operation time. This optimization problem has three primary differences with the literature references described above. First, the drone can visit multiple locations, limited by the battery power, after departing from the ground vehicle. In most of the parcel delivery problems, the drone visited only one node and returned to the vehicle to retrieve another parcel. Second, the ground vehicle acts as a moving refueling depot unlike [12, 28] which use a fixed refueling depot. Third, the formulation is based on the rule "drone first, ground vehicle second" with clustering. Previously this rule has been used in the parcel delivery problem [19, 20] and the partial accessibility constrained vehicle routing problem [32]. The clustering has been used in the multiple depot vehicle routing problem [33, 34].

The remainder of this chapter is organized as follows. In Section 2.3, we describe the proposed optimization model and solution approach. In Section 2.4, we present the experimental results including three cases created by using real locations of wind farms. Finally, in Section 2.5, we state the conclusion.

## 2.3 Proposed optimization model and solution approach

The inspection system uses one drone and one ground vehicle. The drone performs the visual inspection of homogeneous wind turbines which requires the same inspection time of $s$ minutes per wind turbine. It is assumed that the drone flies at a constant speed of $FS$ mph and has a flight endurance of $FE$ minutes (maximum flight time). The ground vehicle moves from cluster to cluster at a constant speed of $GS$ mph and parks at only one of the designated locations near a

wind turbine of a cluster to deploy and collect the drone. At each stop, the drone pilot conducts pre and post-flight procedures for *p* minutes. The optimization model is based on the rule "drone first, ground vehicle second" and is solved in two steps. The first step optimizes the drone routing in each cluster. The second step optimizes the ground vehicle routing using the clustering outcome from the first step.

### 2.3.1 Optimizing the drone route

We represent the one depot and *n* wind turbines by the set $N = \{0, 1, 2, \dots, n\}$ where the value 0 refers to the depot. We first group the wind turbines into *k* clusters. The clustering approach is appropriate to our problem when wind farms have either a multiple/parallel string configuration or a cluster configuration [35]. We use the K-means clustering algorithm, also known as the Lloyd's algorithm [36]. Given the number of centroids *k* as a parameter, the algorithm iteratively finds the best cluster assignments so that the sum of the distances between the wind turbines and their centroid is minimized. After the wind turbines have been grouped into *k* clusters, we obtain *k* mutually exclusive and exhaustive subsets of the set $N$ excluding the depot. The subsets are denoted by $N^a$ which includes all wind turbines in cluster $a = 1, 2, \dots, k$. Then, we find the shortest flight path of the drone to visit all wind turbines in $N^a$. Note that this drone routing problem is the classical TSP [37]. For cluster with *m* wind turbines, we solve the drone routing problem using the integer linear programming model described in Eqs. (2.1) – (2.5).

$$\text{Min } T^D = \sum_{\substack{i \in N^a \\ i \neq j}} \sum_{\substack{j \in N^a \\ i \neq j}} f_{ij} z_{ij} + m \times s \tag{2.1}$$

Subject to

$$\sum_{i \in N^a} z_{ij} = 1 \quad \forall j \in \{N^a : i \neq j\} \tag{2.2}$$

$$\sum_{j \in N^a} z_{ij} = 1 \quad \forall i \in \{N^a : i \neq j\} \tag{2.3}$$

$$\sum_{i \in S} \sum_{j \in S} z_{ij} \leq |S| - 1 \quad \forall S \subseteq N^a \setminus \{1\}, S \neq \phi \tag{2.4}$$

$$z_{ij} \in \{0, 1\} \quad \forall i, j \in \{N^a : i \neq j\} \tag{2.5}$$

The objective function (2.1) minimizes the operation time by the drone, $T^D$, which is the sum of the total flight time and inspection time of the drone over all wind turbines in a cluster. Constraints (2.2) ensure that the drone enters to visit each wind turbine exactly once. Constraints (2.3) ensure that the drone leaves each wind turbine exactly once. Constraints (2.4) exclude subtours among wind turbines. Constraints (2.5) indicate that $z_{ij}$ is a binary variable.

We start by considering only one cluster ($k = 1$). The shortest flight path of the drone is determined by solving the optimization problem described in Eqs. (2.1) – (2.5) with an integer linear programming. We ignore the constant time due to taking-off and landing. The operation time by the drone ($T^D$) is compared to the flight endurance ($FE$), which is the maximum flight time due to the battery capacity. If the operation time by the drone is greater than the flight endurance, we consider two clusters. The clusters are determined by using the K-means clustering algorithm. If again the operation time by the drone in each cluster is greater than the flight endurance, we increase *k* by 1 and repeat the procedure. Otherwise, we stop the procedure and store two clustering outcomes: the cluster set *K* and cluster assignment in vector **C** (dimension $n + 1$). The value of the element in **C**, $C_i$ ($i = 0, 1, 2, \cdots, n$), refers to the depot (0) or the cluster

$(1, 2, \cdots, k)$ where the wind turbine $i$ has been assigned. Figure 2.1 illustrates an example where 15 wind turbines have been grouped into 4 clusters.

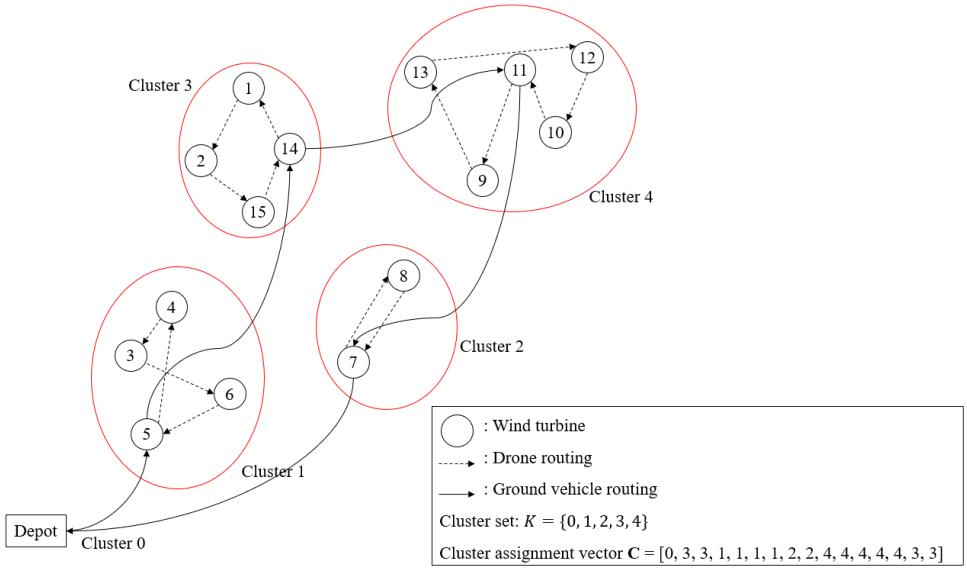



**Figure 2.1** 15-wind turbine cluster assignment

### 2.3.2 Optimizing the ground vehicle route

The problem of routing the ground vehicle consists of determining the shortest route to visit all *k* clusters, only once, and return to the depot location. In addition, the park location at each cluster needs to be determined. We assume that the possible park locations are the same as the locations of the wind turbines. Note that this ground vehicle routing problem is the equality generalized traveling salesman problem (E-GTSP) [38]. Feeding two clustering outcomes generated by the previous step as inputs, we solve the ground vehicle routing problem using the integer linear programming model described in Eqs. (2.6) – (2.18).

$$\text{Min } T^G = \sum_{\substack{i \in N \\ i \neq j}} \sum_{\substack{j \in N \\ i \neq j}} t_{ij} x_{ij} \tag{2.6}$$

Subject to

$$\sum_{\substack{i\in N\\ i\neq 0}} x_{0i} = 1 \tag{2.7}$$

$$\sum_{\substack{i\in N\\ i\neq 0}} x_{i0} = 1 \tag{2.8}$$

$$\sum_{\substack{i\in N\\ C_i=a}} \sum_{\substack{j\in N\\ C_j\neq a}} x_{ij} = 1 \quad \forall a \in \{K : a \neq 0\} \tag{2.9}$$

$$\sum_{\substack{i\in N\\ C_i\neq a}} \sum_{\substack{j\in N\\ C_j= a}} x_{ij} = 1 \quad \forall a \in \{K : a \neq 0\} \tag{2.10}$$

$$\sum_{\substack{i\in N\\ C_i= a}} \sum_{\substack{j\in N\\ C_j= b}} x_{ij} = y_{ab} \quad \forall a, b \in \{K : a \neq b\} \tag{2.11}$$

$$\sum_{\substack{i\in N\\ C_i\neq C_j}} x_{ij} = \sum_{\substack{l\in N\\ C_l\neq C_j}} x_{jl} \quad \forall j \in \{N : j \neq 0\} \tag{2.12}$$

$$u_0 = 0 \tag{2.13}$$

$$1 \leq u_a \leq k \quad \forall a \in \{K : a \neq 0\} \tag{2.14}$$

$$u_a - u_b + 1 \leq k(1 - y_{ab}) \quad \forall a, b \in \{K : a \neq b, a \neq 0, b \neq 0\} \tag{2.15}$$

$$x_{ij} \in \{0, 1\} \quad \forall i, j \in \{N : i \neq j\} \tag{2.16}$$

$$y_{ab} \in \{0, 1\} \quad \forall a, b \in \{K : a \neq b\} \tag{2.17}$$

$$u_a \in \{0, 1, 2, \dots, k\} \quad \forall a \in K \tag{2.18}$$

The objective function (2.6) minimizes the operation time by the ground vehicle, $T^G$, that is the total moving time of the ground vehicle, departing from and returning to the depot. Constraints (2.7) and (2.8) ensure that the ground vehicle departs from and returns to the depot

exactly once. Constraints (2.9) and (2.10) enforce that only one wind turbine in each cluster is selected as a stop location of the ground vehicle, ensuring that the ground vehicle visits each cluster exactly once. Constraints (2.11) force a relationship between the decision variables $x_{ij}$ and $y_{ab}$. Suppose that wind turbines *i* and *j* were assigned to clusters *a* and *b*, respectively. If the ground vehicle moves from wind turbine *i* to *j* ($x_{ij} = 1$), the ground vehicle should also move from cluster *a* to *b* ($y_{ab} = 1$). Constraints (2.12) ensure that the ground vehicle moving to the wind turbine *j* from wind turbine *i* must also depart from the same wind turbine *j* to another wind turbine, say *l*. Constraints (2.13) – (2.15) exclude subtours among clusters and the depot. We apply the Miller-Tucker-Zemlin (MTZ) formulation given in [39] where another decision variable u is introduced to represent the sequence number cluster visits. Constraints (2.16) and (2.17) indicate that $x_{ij}$ and $y_{ab}$ are binary variables. Constraints (2.18) indicate that *u* is an integer variable. The flow chart for optimizing the drone routing and the ground vehicle is shown in Figure 2.2. Finally, the total operation time (*T*) is computed by summing the operation time of the drone ($T^D$), the operation time of the ground vehicle ($T^G$), and the total time for conducting pre- and post-flight procedures over the *k* clusters as described in Eq. 2.19.

$$T = T^D + T^G + p \times k \tag{2.19}$$

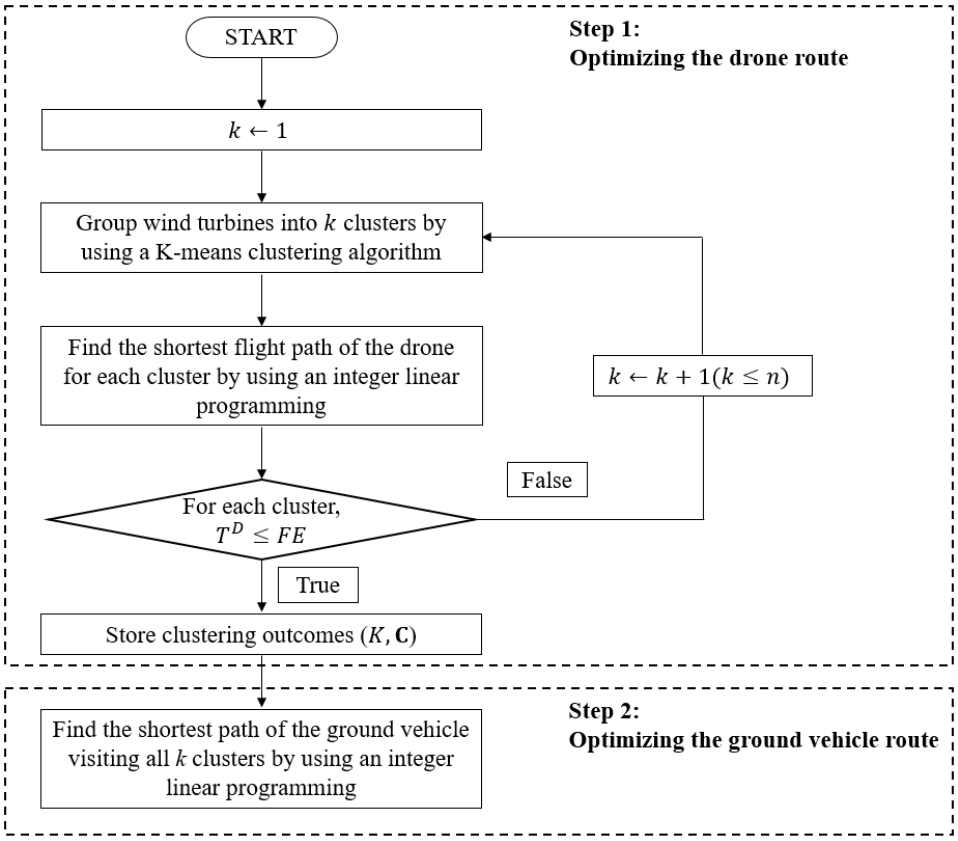



**Figure 2.2** Flow chart of the solution approach

## 2.4 Experimental results

For convenience, the drone routing is coded in MATLAB. We employ the K-means clustering function and solve the drone routing optimization problem using the "intlinprog" function of MATLAB. The ground vehicle routing is coded in AMPL and solved using the CPLEX 12.8 solver. Note that we cannot guarantee to solve our optimization problems in deterministic polynomial time because both TSP and E-GTSP are NP-hard problems. Thus, all the drone and ground vehicle routing optimization problems are solved with a 30-min time limit. All case studies

are run on a computer with an Intel® Core™ i5-8250U (1.60 GHz) processor with 8 GB of RAM memory.

### 2.4.1 System parameters and case studies

We use the drone and ground vehicle parameters shown in Table 2.1.

**Table 2.1** System parameters

| | |
|---|---|
| Drone | |
| Flying speed (*FS*) | 40 mph |
| Flight endurance (*FE*) | 50 min |
| Inspection time for a wind turbine (*s*) | 5 min |
| Ground vehicle | |
| Moving speed (*GS*) | 20 mph |
| Time for conducting pre- and post-flight procedures (*p*) | 5 min |

To test the proposed model and solution approach, we create three case studies by examining the number of wind turbines per wind project in the state of Texas. According to the American Wind Energy Association (AWEA), Texas has the largest wind energy capacity in the U.S. (more than 10,000 MW) [40]. As shown in Figure 2.3, we plot a histogram of the number of wind turbines on 172 wind projects in Texas.

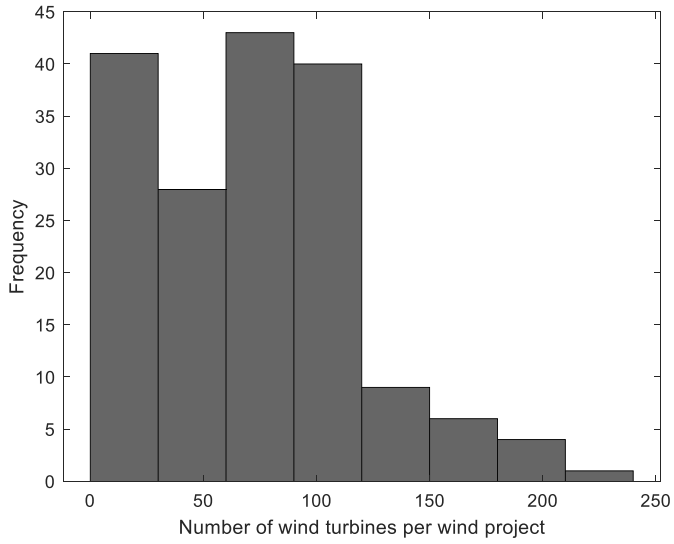
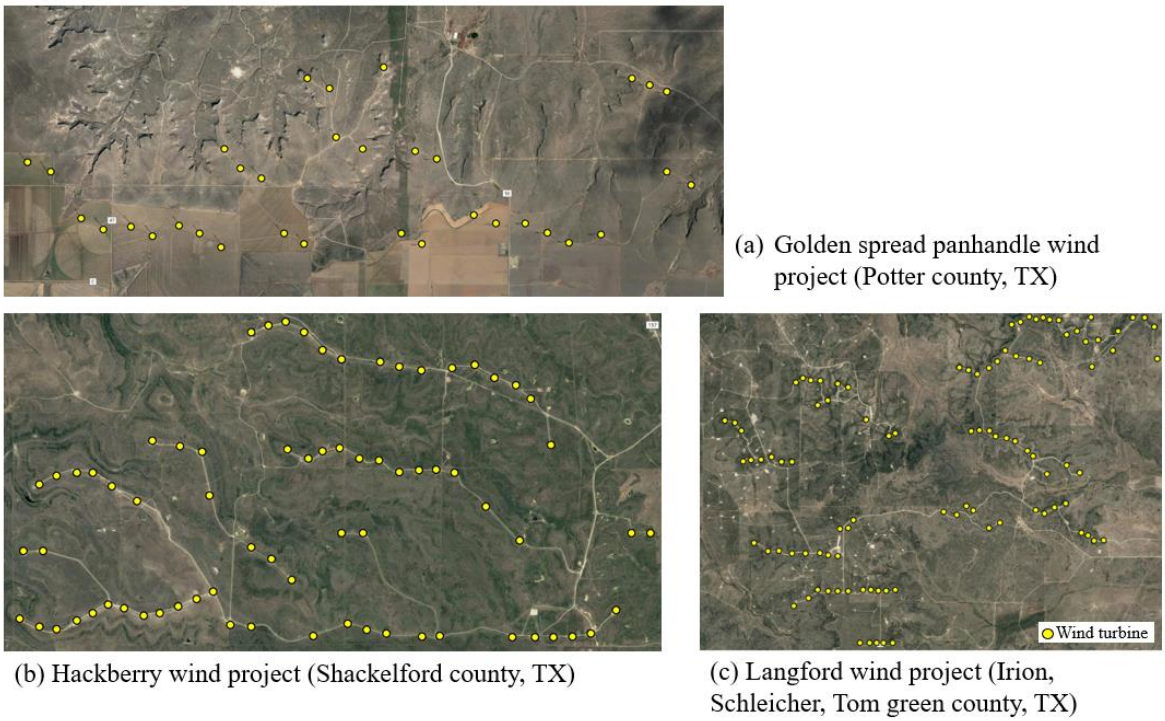



**Figure 2.3** Histogram of the number of wind turbines per wind project in Texas

We use the first, second, and third quartile of the size of the wind projects to determine the number of wind turbines for creating the three case studies. We select the Golden Spread Panhandle Wind Project near Wildorado, Texas to represent a small wind farm (first quartile). This case comprises 34 wind turbines across a 6-mile square region. To represent a medium-sized wind farm (second quartile), we select the Hackberry Wind Project in Shackelford County, Texas. This wind project has 72 wind turbines across a 15-mile square region. For the third case representing a large-sized wind farm (third quartile), we select the Langford Wind Project located around 300 miles southwest of Dallas, Texas. This wind project has 100 wind turbines across a 55-mile square region. Figure 2.4 maps the actual locations of the wind turbines for these three cases. Note that the depots are located at the mass center of the wind turbines.




(a) Golden spread panhandle wind project (Potter county, TX)

(b) Hackberry wind project (Shackelford county, TX)

(c) Langford wind project (Irion, Schleicher, Tom green county, TX)

**Figure 2.4** Google satellite maps of the wind turbines for the three case studies

In Table 2.2, we provide the geographic locations of the wind turbines for each of the three wind farms.

**Table 2.2** Geographic locations of the wind turbines

| No | Small-sized Wind Farm | | Medium-sized Wind Farm | | Large-sized Wind Farm | |
|---|---|---|---|---|---|---|
| | Latitude | Longitude | Latitude | Longitude | Latitude | Longitude |
| 1 | 35.2360 | -102.2494 | 32.7673 | -99.5048 | 31.0812 | -100.7139 |
| 2 | 35.2350 | -102.2463 | 32.7662 | -99.5021 | 31.0834 | -100.7090 |
| 3 | 35.2297 | -102.2419 | 32.7663 | -99.4990 | 31.0851 | -100.7048 |
| 4 | 35.2287 | -102.2391 | 32.7668 | -99.4961 | 31.0851 | -100.7015 |
| 5 | 35.2289 | -102.2350 | 32.7682 | -99.4936 | 31.0851 | -100.6981 |
| 6 | 35.2279 | -102.2321 | 32.7694 | -99.4910 | 31.0851 | -100.6950 |
| 7 | 35.2289 | -102.2282 | 32.7688 | -99.4882 | 31.0855 | -100.6889 |
| 8 | 35.2278 | -102.2253 | 32.7679 | -99.4853 | 31.0854 | -100.6864 |
| 9 | 35.2267 | -102.2224 | 32.7683 | -99.4824 | 31.0855 | -100.6838 |
| 10 | 35.2279 | -102.2138 | 32.7690 | -99.4796 | 31.0854 | -100.6814 |
| 11 | 35.2268 | -102.2110 | 32.7701 | -99.4770 | 31.0854 | -100.6789 |
| 12 | 35.2276 | -102.1973 | 32.7709 | -99.4741 | 31.0993 | -100.7264 |
| 13 | 35.2266 | -102.1944 | 32.7665 | -99.4711 | 31.0967 | -100.7218 |
| 14 | 35.2300 | -102.1869 | 32.7664 | -99.4683 | 31.0972 | -100.7181 |
| 15 | 35.2289 | -102.1840 | 32.7764 | -99.5046 | 31.0962 | -100.7140 |
| 16 | 35.2290 | -102.1799 | 32.7765 | -99.5015 | 31.0966 | -100.7093 |
| 17 | 35.2280 | -102.1771 | 32.7857 | -99.5020 | 31.0969 | -100.7045 |
| 18 | 35.2269 | -102.1743 | 32.7865 | -99.4993 | 31.0961 | -100.7007 |
| 19 | 35.2278 | -102.1697 | 32.7874 | -99.4964 | 31.0961 | -100.6980 |
| 20 | 35.2348 | -102.1606 | 32.7870 | -99.4935 | 31.1036 | -100.6983 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 21 | 35.2336 | -102.1574 | 32.7853 | -99.4901 | 31.1042 | -100.6956 |
| 22 | 35.2438 | -102.1605 | 32.7834 | -99.4864 | 31.1061 | -100.6930 |
| 23 | 35.2447 | -102.1629 | 32.7910 | -99.4838 | 31.0696 | -100.6904 |
| 24 | 35.2456 | -102.1652 | 32.7908 | -99.4796 | 31.0696 | -100.6880 |
| 25 | 35.2363 | -102.1924 | 32.7897 | -99.4763 | 31.0696 | -100.6851 |
| 26 | 35.2373 | -102.1952 | 32.7841 | -99.4746 | 31.0696 | -100.6820 |
| 27 | 35.2375 | -102.2030 | 32.7767 | -99.4681 | 31.0696 | -100.6795 |
| 28 | 35.2387 | -102.2063 | 32.7751 | -99.4650 | 31.1235 | -100.7310 |
| 29 | 35.2466 | -102.1996 | 32.7729 | -99.4616 | 31.1240 | -100.7283 |
| 30 | 35.2443 | -102.2074 | 32.7652 | -99.4579 | 31.1243 | -100.7252 |
| 31 | 35.2454 | -102.2103 | 32.7669 | -99.4525 | 31.1243 | -100.7211 |
| 32 | 35.2344 | -102.2171 | 32.7660 | -99.4495 | 31.1233 | -100.7177 |
| 33 | 35.2354 | -102.2200 | 32.7653 | -99.4466 | 31.1235 | -100.7141 |
| 34 | 35.2374 | -102.2224 | 32.7653 | -99.4410 | 31.1356 | -100.7369 |
| 35 | | | 32.7654 | -99.4379 | 31.1347 | -100.7340 |
| 36 | | | 32.7652 | -99.4266 | 31.1328 | -100.7315 |
| 37 | | | 32.7653 | -99.4235 | 31.1359 | -100.6885 |
| 38 | | | 32.7652 | -99.4197 | 31.1312 | -100.6810 |
| 39 | | | 32.7651 | -99.4167 | 31.1318 | -100.6786 |
| 40 | | | 32.7655 | -99.4137 | 31.1471 | -100.7127 |
| 41 | | | 32.7685 | -99.4102 | 31.1479 | -100.7097 |
| 42 | | | 32.7789 | -99.4537 | 31.1477 | -100.7073 |
| 43 | | | 32.7786 | -99.4501 | 31.1470 | -100.7042 |
| 44 | | | 32.7900 | -99.4625 | 31.1406 | -100.7047 |
| 45 | | | 32.7888 | -99.4592 | 31.1414 | -100.7023 |
| 46 | | | 32.7898 | -99.4567 | 31.1458 | -100.6982 |
| 47 | | | 32.7903 | -99.4538 | 31.1452 | -100.6949 |
| 48 | | | 32.7891 | -99.4507 | 31.1088 | -100.6612 |
| 49 | | | 32.7886 | -99.4476 | 31.1078 | -100.6579 |
| 50 | | | 32.7875 | -99.4444 | 31.1100 | -100.6541 |
| 51 | | | 32.7873 | -99.4414 | 31.1089 | -100.6513 |
| 52 | | | 32.7873 | -99.4385 | 31.1041 | -100.6460 |
| 53 | | | 32.7867 | -99.4355 | 31.1053 | -100.6433 |
| 54 | | | 32.7825 | -99.4307 | 31.1088 | -100.6301 |
| 55 | | | 32.7778 | -99.4255 | 31.1083 | -100.6264 |
| 56 | | | 32.7789 | -99.4073 | 31.1100 | -100.6236 |
| 57 | | | 32.7789 | -99.4043 | 31.1110 | -100.6198 |
| 58 | | | 32.8056 | -99.4684 | 31.1024 | -100.6143 |
| 59 | | | 32.8065 | -99.4656 | 31.1015 | -100.6114 |
| 60 | | | 32.8069 | -99.4626 | 31.1001 | -100.6090 |
| 61 | | | 32.8060 | -99.4597 | 31.0999 | -100.6066 |
| 62 | | | 32.8034 | -99.4567 | 31.1323 | -100.6519 |
| 63 | | | 32.8023 | -99.4538 | 31.1330 | -100.6492 |
| 64 | | | 32.8015 | -99.4475 | 31.1322 | -100.6463 |
| 65 | | | 32.8012 | -99.4447 | 31.1310 | -100.6434 |
| 66 | | | 32.8007 | -99.4409 | 31.1306 | -100.6407 |
| 67 | | | 32.8011 | -99.4364 | 31.1295 | -100.6373 |
| 68 | | | 32.8012 | -99.4324 | 31.1264 | -100.6334 |
| 69 | | | 32.7997 | -99.4292 | 31.1252 | -100.6309 |
| 70 | | | 32.7986 | -99.4265 | 31.1198 | -100.6265 |
| 71 | | | 32.7968 | -99.4238 | 31.1225 | -100.6194 |
| 72 | | | 32.7905 | -99.4204 | 31.1196 | -100.6152 |
| 73 | | | | | 31.1514 | -100.6559 |
| 74 | | | | | 31.1504 | -100.6532 |
| 75 | | | | | 31.1490 | -100.6496 |
| 76 | | | | | 31.1510 | -100.6464 |
| 77 | | | | | 31.1532 | -100.6433 |

| | | |
|---|---|---|
| 78 | 31.1560 | -100.6404 |
| 79 | 31.1543 | -100.6367 |
| 80 | 31.1538 | -100.6317 |
| 81 | 31.1530 | -100.6285 |
| 82 | 31.1640 | -100.6378 |
| 83 | 31.1643 | -100.6353 |
| 84 | 31.1660 | -100.6325 |
| 85 | 31.1656 | -100.6299 |
| 86 | 31.1660 | -100.6274 |
| 87 | 31.1655 | -100.6248 |
| 88 | 31.1652 | -100.6222 |
| 89 | 31.1613 | -100.6194 |
| 90 | 31.1620 | -100.6156 |
| 91 | 31.1653 | -100.6114 |
| 92 | 31.1589 | -100.6122 |
| 93 | 31.1589 | -100.6083 |
| 94 | 31.1617 | -100.6012 |
| 95 | 31.1664 | -100.5965 |
| 96 | 31.1519 | -100.6105 |
| 97 | 31.1568 | -100.6036 |
| 98 | 31.1661 | -100.5925 |
| 99 | 31.1628 | -100.5890 |
| 100 | 31.1539 | -100.5877 |

### 2.4.2 Routing solutions

We solve the three case studies under the same system parameters. In Figures 2.5 – 2.7, we plot the routing solutions of the drone and ground vehicle. The clusters are marked with the shaded color. For the small-sized wind farm, the 34 wind turbines are grouped into 6 clusters. For the medium-sized wind farm, the 72 wind turbines are grouped into 13 clusters. For the large-sized wind farm, the 100 wind turbines are grouped into 21 clusters.

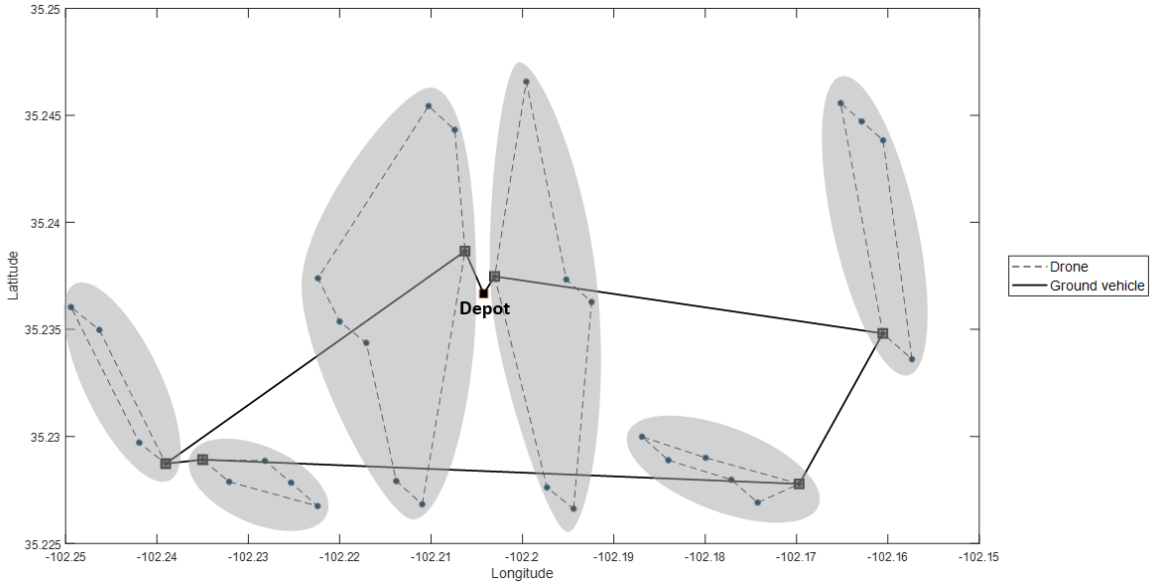



**Figure 2.5** Small-sized wind farm routing solution for the drone and ground vehicle

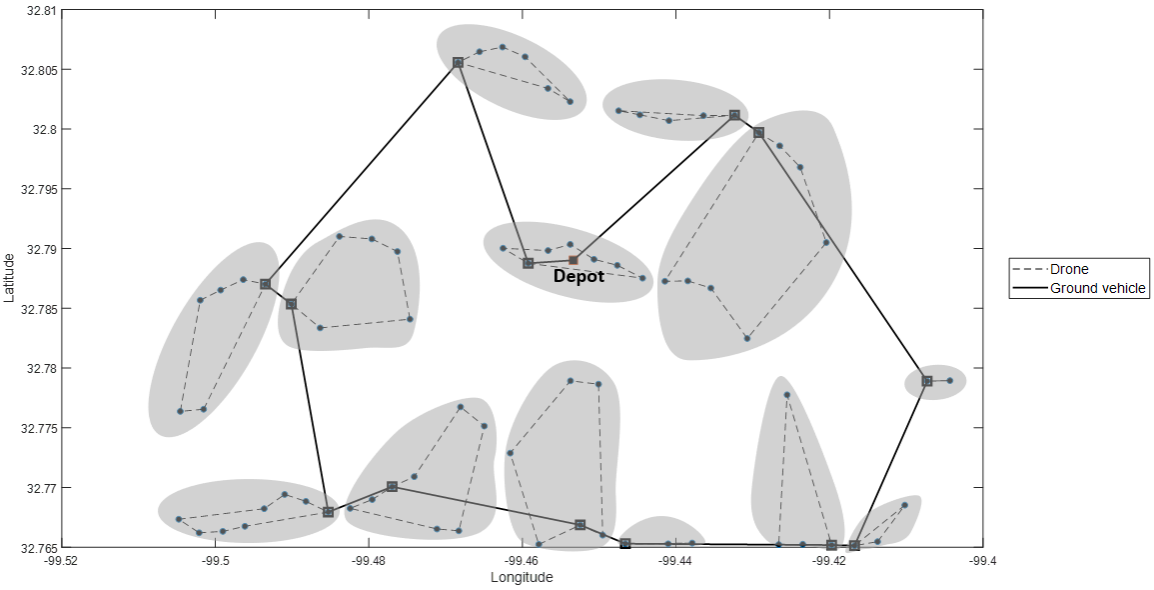



**Figure 2.6** Medium-sized wind farm routing solution of the drone and ground vehicle

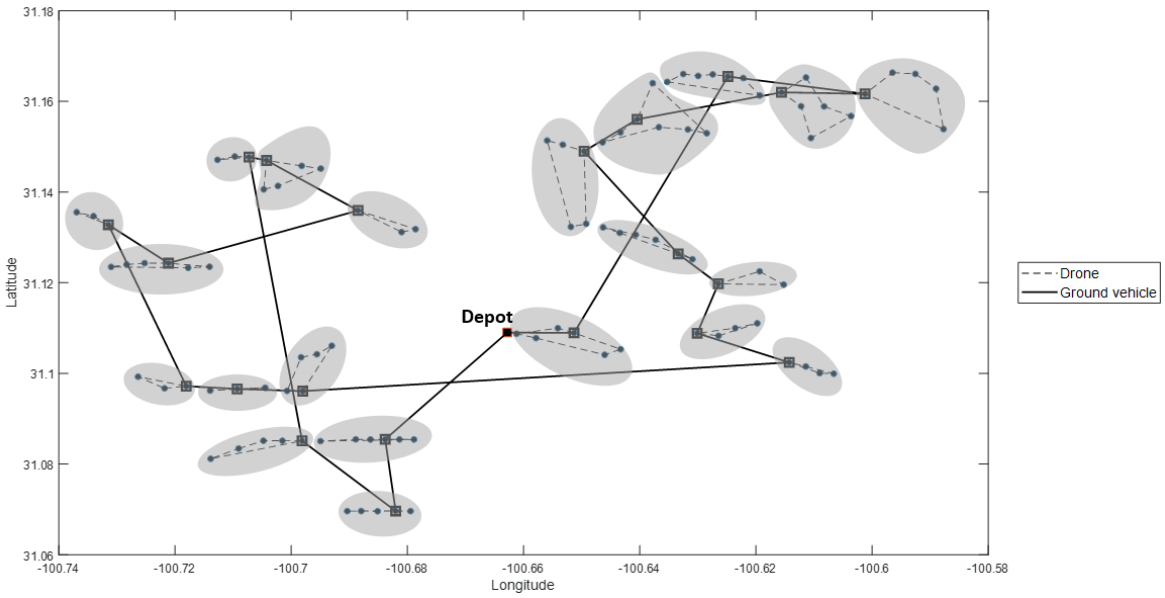



**Figure 2.7** Large-sized wind farm routing solution of the drone and ground vehicle

### 2.4.3 Effect of clustering and ground vehicle routing

To study the effects of the clustering of the wind turbines, we compare the results to a model that assumes no clustering of the wind turbines so the drone flies to inspect the closest wind turbine after taking-off from the ground vehicle. In this case, the ground vehicle needs to visit all wind turbines. The resulting ground vehicle routing optimization problem is the classical TSP which we solve using the CPLEX solver. We named this model M1. Similarly, to study the effects of the ground vehicle routing optimization, we compare the results to a model, named M2, that assumes that the wind turbines are clustered, and the routing of the drone is optimized using the "intlinprog" function of MATLAB. However, we use a greedy approach for determining the route of the ground vehicle. At each cluster, the wind turbine closest to the centroid is selected as a stop

location of the ground vehicle. After inspection, the ground vehicle moves to the stop location closest to its current location. Our proposed approach is named M3.

### 2.4.4 Results of the total operation time

First, we study the results of the total operation time under the different models. We calculate the percentage of time reduction of M2 and M3 with respect to M1. The results are shown in Table 2.3. From the results, we observe that M3 outperforms M2 for the three wind farm sizes. The time reduction in minutes increases approximate linearly as we study larger sized wind farms. The largest percentage of time reduction (36.8%) is obtained in the small-sized wind farm (Case 1). Similarly, the lowest percentage of time reduction (34.5%) is obtained in the large-sized wind farm (Case 3).

**Table 2.3** Comparative results of the total operation time

| Size | Model | $T^D$ (min) | $T^G$ (min) | $p$ (min) | $T$ (min) | Time reduction (min) | Time reduction (%) |
|---|---|---|---|---|---|---|---|
| Small | M1 | 170 | 68 | 170 | 408 | - | - |
| | M2 | 189 | 54 | 30 | 272 | 135 | 33.2 |
| | M3 | 189 | 39 | 30 | 258 | 150 | 36.8 |
| Medium | M1 | 360 | 98 | 360 | 818 | - | - |
| | M2 | 399 | 72 | 65 | 537 | 281 | 34.4 |
| | M3 | 399 | 60 | 65 | 525 | 293 | 35.9 |
| Large | M1 | 500 | 235 | 500 | 1,235 | - | - |
| | M2 | 556 | 180 | 105 | 841 | 394 | 31.9 |
| | M3 | 556 | 148 | 105 | 808 | 426 | 34.5 |

### 2.4.5 Effect of the flight speed

To study the effect of the flight speed, we solve all problems and using seven different flight speeds from 10 to 40 mph in steps of 5 mph. In Figures 2.8 – 2.10, we plot the percentage of time reduction for models M2 and M3 for each wind farm cases. In the three wind farm sizes, we observe that the percentage of time reduction of M2 and M3 increases as the drone flies faster. The difference in time reduction between M2 and M3 remains practically unchanged as the drone flies faster. In the small-sized wind farm, the slopes decrease significantly when the flight speed becomes larger than 20 mph which is the speed of the ground vehicle. In the medium- and large-sized wind farms, the same change of slope occurs at 25 mph. In all three wind farm sizes, the percentage of time reduction of our approach (M3) is always greater than of model M2 for all values of the flight speed.

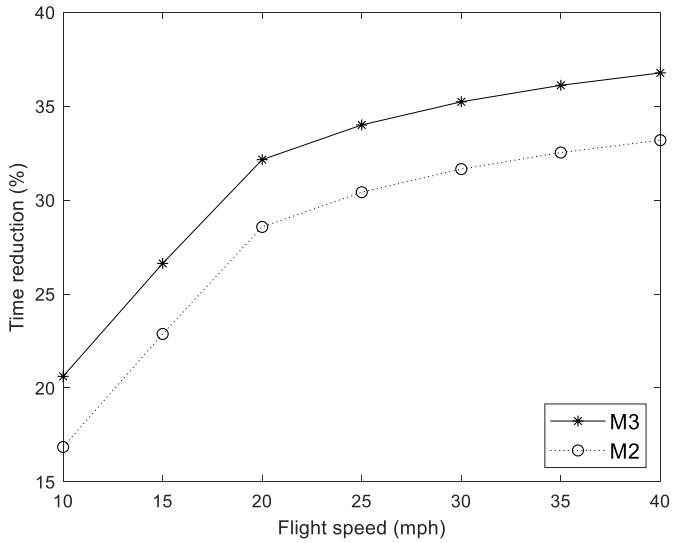



**Figure 2.8** Effect of changing the flight speed in the small-sized wind farm

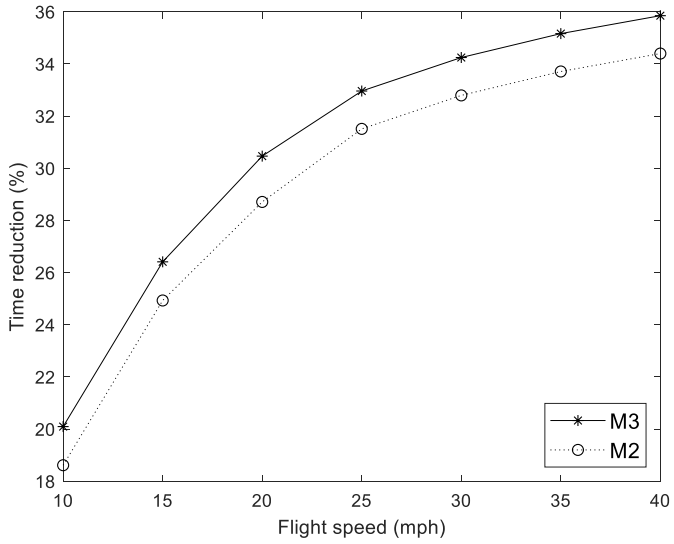



**Figure 2.9** Effect of changing the flight speed in the medium-sized wind farm

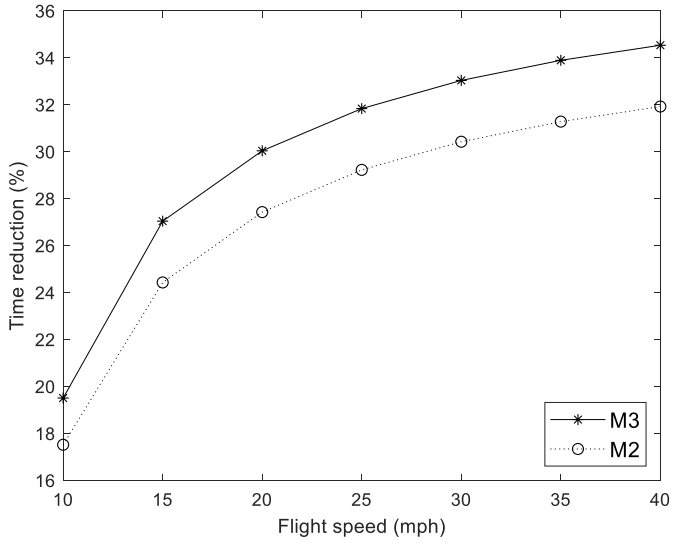



**Figure 2.10** Effect of changing the flight speed in the large-sized wind farm

### 2.4.6 Effect of the flight endurance

To study the effect of the flight endurance, we solve again all problems and use seven different flight endurances from 20 to 50 minutes in steps of 5 minutes. In Figures 2.11 – 2.13, we

plot the percentage of time reduction for each wind farm case. In the three wind farm cases, we observe that the percentage of time reduction of M2 and M3 increases as the flight endurance increases. In the small-sized wind farm, no change in time reduction is observed when the flight endurance changes from 35 to 40 minutes and from 45 to 50 minutes. The same situation occurs for the medium-sized farm when the flight endurance changes from 25 to 30 minutes and for the large-sized farm when it changes from 40 to 50 minutes. In all three wind farm sizes, the percentage of time reduction for our approach (M3) is always greater than of model M2 for all values of the flight endurance.

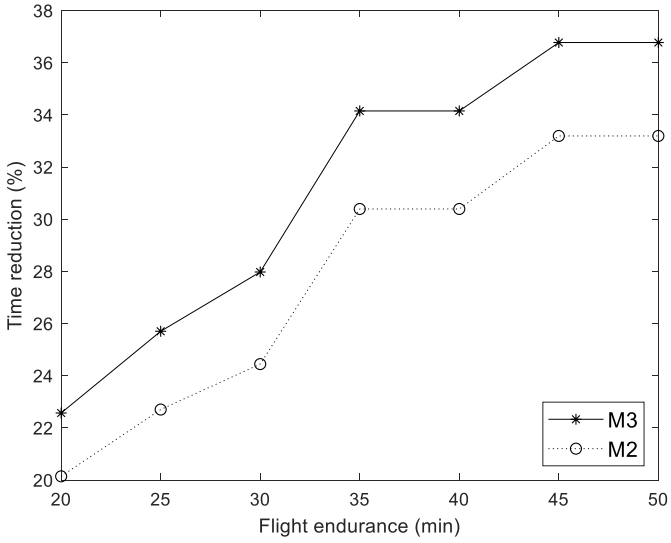



**Figure 2.11** Effect of changing the flight endurance in the small-sized wind farm

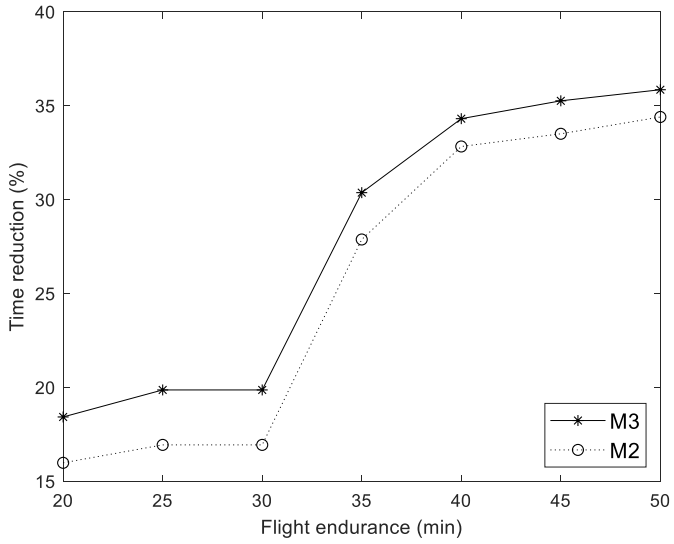



**Figure 2.12** Effect of changing the flight endurance in the medium-sized wind farm

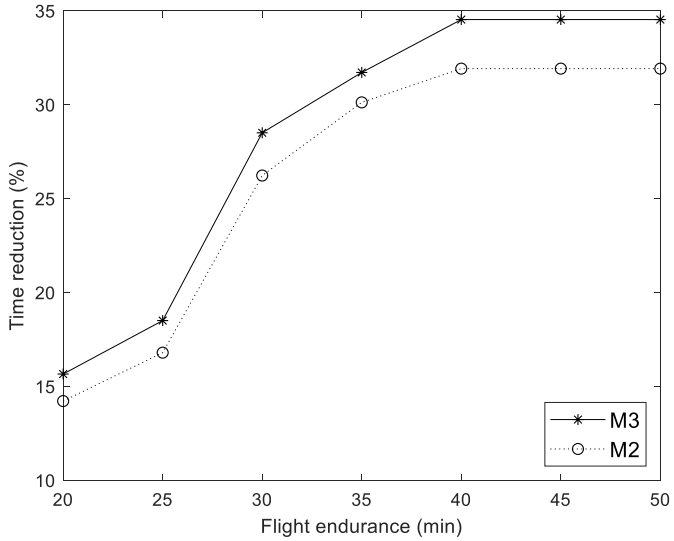



**Figure 2.13** Effect of changing the flight endurance in the large-sized wind farm

## 2.5 Conclusion

We proposed a routing optimization model for a drone and ground vehicle to reduce the total operation time for inspecting a set of wind turbines. We solved the optimization model by

using the K-means clustering algorithm and two integer linear programming formulations. Three wind farm size cases were created by using real locations of wind farms. We developed two other experimental models to evaluate the proposed model. We tested these models in the three cases. The experimental results showed that the proposed model reduces the total operation time of the inspection for the three wind farm sizes when compared to no clustering or the use of a greedy approach for determining the route of the ground vehicle. In addition, we studied the effects of the flight speed and endurance on the time reduction. The results showed that the total operation time can be reduced significantly as the flight speed and endurance are increased. Given a budget and technical constraints to operate the drone, these results can be of use to practitioners to determine the technical specifications of the drone. In all three wind farm sizes, our approach outperformed the model M2 for all tested values of the flight speed and endurance. This result shows the robustness of our approach to changes to the system parameters.

## References


[1] U.S. Department of Energy, Energy Information Administration, "Annual energy outlook 2018 with projections to 2050," 2018. [Online]. Available: https://www.eia.gov/outlooks/aeo/pdf/AEO2018.pdf. [Accessed Oct. 28, 2018].

[2] J. Deign, "Fully automated drones could double wind turbine inspection rates," *New Energy Update*, Sep. 27, 2016. [Online]. Available: http://newenergyupdate.com/wind-energy-update/fully-automated-drones-could-double-wind-turbine-inspection-rates. [Accessed Oct. 28, 2018].

[3] M. Capots, "Self-flying drones and wind turbine blades: the new way of collecting data," *Office of Energy Efficiency & Renewable Energy*, Feb. 23, 2017. [Online]. Available: https://www.energy.gov/eere/articles/self-flying-drones-and-wind-turbine-blades-new-way-collecting-data. [Accessed Oct. 28, 2018].

[4] M. Stokkeland, K. Klausen, and T. A. Johansen, "Autonomous visual navigation of unmanned aerial vehicle for wind turbine inspection," in *Proceedings of the International Conference on Unmanned Aircraft Systems*, 2015, pp. 998-1007.

[5] B. E. Schäfer, D. Picchi, T. Engelhardt, and D. Abel, "Multicopter unmanned aerial vehicle for automated inspection of wind turbines," in *Proceedings of the Mediterranean Conference on Control and Automation,* 2016, pp. 244-249.

[6] C. Galleguillos, A. Zorrilla, A. Jimenez, L. Diaz, Á. L. Montiano, M. Barroso, A. Viguria, and F. Lasagni, "Thermographic non-destructive inspection of wind turbine blades using unmanned aerial systems," *Plastics, Rubber Composites*, 44(3), pp. 98-103, 2015.

[7] G. Collins, A. Clausse, and D. Twining, "Enabling technologies for autonomous offshore inspections by heterogeneous unmanned teams," in *Proceedings of the OCEANS 2017 - Aberdeen*, 2017.

[8] C. C. Murray and A. G. Chu, "The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery," *Transportation Research Part C: Emerging Technologies*, 54, pp. 86-109, 2015.

[9] A. Ponza, "Optimization of Drone-Assisted Parcel Delivery," M.S. thesis, University of Padova, Italy, 2016.

[10] Q. M. Ha, Y. Deville, Q. D. Pham, and M. H. Hà, "On the min-cost traveling salesman problem with drone," *Transportation Research Part C: Emerging Technologies*, 86, pp. 597-621, 2018.

[11] J. C. de Freitas and P. H. V. Penna, "A randomized variable neighborhood descent heuristic to solve the flying sidekick traveling salesman problem," *Electronic Notes in Discrete Mathematics*, 66, pp. 95-102, 2018.

[12] S. Kim and I. Moon, "Traveling salesman problem with a drone station," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 49(1), pp. 42-52, 2018.

[13] M. Marinelli, L. Caggiani, M. Ottomanelli, and M. Dell'Orco, "En route truck–drone parcel delivery for optimal vehicle routing strategies," *IET Intelligent Transport Systems*, 12(4), pp. 253-261, 2018.

[14] C. Gambella, A. Lodi, and D. Vigo, "Exact solutions for the carrier–vehicle traveling salesman problem," *Transportation Science*, 52(2), pp. 320-330, 2017.

[15] N. Agatz, P. Bouman, and M. Schmidt, "Optimization approaches for the traveling salesman problem with drone," *Transportation Science*, 52(4), pp. 965-981, 2018.

[16] P. Bouman, N. Agatz, and M. Schmidt, "Dynamic programming approaches for the traveling salesman problem with drone," *Networks*, 72(4), pp. 528-542, 2018.

[17] N. Boysen, D. Briskorn, S. Fedtke, and S. Schwerdfeger, "Drone delivery from trucks: Drone scheduling for given truck routes," *Networks*, 72(4), pp. 506-527, 2018.

[18] I. Hong, M. Kuby, and A. T. Murray, "A range-restricted recharging station coverage model for drone delivery service planning," *Transportation Research Part C: Emerging Technologies*, 90, pp. 198-212, 2018.

[19] S. M. Ferrandez, T. Harbison, T. Weber, R. Sturges, and R. Rich, "Optimization of a truck-drone in tandem delivery network using k-means and genetic algorithm," *Journal of Industrial Engineering and Management*, 9(2), pp. 374-388, 2016.

[20] Y. S. Chang and H. J. Lee, "Optimal delivery routing with wider drone-delivery areas along a shorter truck-route," *Expert Systems with Applications*, 104, pp. 307-317, 2018.

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

[22] K. Dorling, J. Heinrichs, G. G. Messier, and S. Magierowski, "Vehicle routing problems for drone delivery," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(1), pp. 70-85, 2017.

[23] X. Wang, S. Poikonen, and B. Golden, "The vehicle routing problem with drones: several worst-case results," *Optimization Letters*, 11(4), pp. 679-697, 2017.

[24] S. Poikonen, X. Wang, and B. Golden, "The vehicle routing problem with drones: Extended models and connections," *Networks,* 70(1), pp. 34-43, 2017.

[25] S. Chowdhury, A. Emelogu, M. Marufuzzaman, S. G. Nurre, and L. Bian, "Drones for disaster response and relief operations: A continuous approximation model," *International Journal of Production Economics,* 188, pp. 167-184, 2017.

[26] B. Rabta, C. Wankmüller, and G. Reiner, "A drone fleet model for last-mile distribution in disaster relief operations," *International Journal of Disaster Risk Reduction,* 28, pp. 107-112, 2018.

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

[28] K. Sundar and S. Rathinam, "Algorithms for routing an unmanned aerial vehicle in the presence of refueling depots," *IEEE Transactions on Automation Science and Engineering,* 11(1), pp. 287-294, 2014.

[29] K. Sundar and S. Rathinam, "Algorithms for heterogeneous, multiple depot, multiple unmanned vehicle path planning problems," *Journal of Intelligent & Robotic Systems,* 88(2-4), pp. 513-526, 2017.

[30] H. Savuran and M. Karakaya, "Efficient route planning for an unmanned air vehicle deployed on a moving carrier," *Soft Computing,* 20(7), pp. 2905-2920, 2016.

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

[32] F. Semet, "A two-phase algorithm for the partial accessibility constrained vehicle routing problem," *Annals of Operations Research,* 61(1), pp. 45-65, 1995.

[33] S. Salhi and G. Nagy, "A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling," *Journal of the Operational Research Society*, 50(10), pp. 1034-1042, 1999.

[34] I. D. Giosa, I. L. Tansini, and I. O. Viera, "New assignment algorithms for the multi-depot vehicle routing problem," *Journal of the Operational Research Society*, 53(9), pp. 977-984, 2002.

[35] P. Denholm, M. Hand, M. Jackson, and S. Ong, National Renewable Energy Laboratory, "Land use requirements of modern wind power plants in the United States," Technical Report NREL/TP-6A2-45834, 2009.

[36] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, 28(2), pp. 129-137, Mar. 1982.

[37] G. Dantzig, R. Fulkerson, and S. Johnson, "Solution of a large-scale traveling-salesman problem," *Journal of the Operations Research Society of America*, 2(4), pp. 393-410, 1954.

[38] M. Fischetti, J. J. S. González, and P. Toth, "A branch-and-cut algorithm for the symmetric generalized traveling salesman problem," *Operations Research*, 45(3), pp. 378-494, 1997.

[39] C. E. Miller, A. W. Tucker, and R. A. Zemlin, "Integer programming formulation of traveling salesman problems," *Journal of the ACM*, 7(4), pp. 326-329, 1960.

[40] American Wind Energy Association, *AWEA state wind energy facts*, 2018. [Online]. Available: https://www.awea.org/resources/fact-sheets/state-facts-sheets. [Accessed Oct. 28, 2018].

# Chapter 3 Inspecting Wind Farms with A Multi-Drone System

## 3.1 Abstract

As drone, or also known as unmanned aircraft system (UAS), technology continues to mature, there has been increasing research works to use drones in various civil applications. One potential application is inspecting wind farms because traditionally a technician uses rope access to inspect wind turbine blades spending a lot of time and wind turbines spread over a large area of land. With the drones, we expect to reduce significantly the inspection time. Previously, we introduced a routing optimization model to minimize the total operation time for inspecting a wind farm with one drone and one ground vehicle. In this chapter, we extend this optimization model by considering the visual line-of-sight (VLOS) regulation and a multi-drone system. The optimization model is decomposed into two steps. The first step is clustering the wind turbines and optimizing the multi-drone routes by solving the multiple traveling salesman problem (mTSP) using a mixed integer linear programming model. The second step is optimizing the ground vehicle route by solving the equality generalized traveling salesman problem (E-GTSP) using an integer linear programming model. We conduct three case studies with the proposed model. The studies show how to efficiently inspect wind farms by using drone technology including the optimal multi-drone routes, ground vehicle route, number of batteries, and drone specifications.

**Keywords:** Drone, Unmanned aircraft system, Wind turbine inspection routing, Mixed integer linear programming, Multiple traveling salesman problem, Equality generalized traveling salesman problem

## 3.2 Introduction

As drone technology continues to mature, there has been increasing research works to use drones in various civil applications. An article posted by McKinsey & Company shows that the drone technology already reaches to maturity specially in surveillance, operations, entertainment/advertising applications [1]. Keeping up with this technology improvement, the number of drone-related research publications in the Web of Science database has exponentially increased since 2015 [2]. One potential application is inspecting wind farms because traditionally a technician uses rope access to inspect wind turbine blades spending a lot of time and wind turbines spread over a large area of land. With the drones, we expect to reduce significantly the inspection time.

Previously, we introduced a routing optimization model to minimize the total operation time for inspecting a wind farm with one drone and one ground vehicle. In this chapter, we extend this optimization model by considering the visual line-of-sight (VLOS) regulation in the United States and a multi-drone system carried by the ground vehicle. In the United States, it is mandatory that a drone must fly within the VLOS of the remote pilot or observer during daylight hours [3]. The multi-drone routing problem is a new variant of the multiple traveling salesman problem (mTSP). The interested reader is referred to [4] that reviewed various formulations, solution procedures, and applications of the mTSP.

Parcel delivery is the most popular area of using a multi-drone system. In [5], the authors proposed the parallel drone scheduling TSP (PDSTSP) to find the optimal assignments of the parcels delivered by a ground vehicle and multiple drones for minimizing the total delivery time in the case of a distribution center located near customers. The optimization problem was solved using both a mixed integer linear programming (MILP) model and heuristic approaches for 10- and 20-customer cases. [6] applied the K-means clustering algorithm to group customers and determine centroid locations that are used as stop locations of the ground vehicle. In each cluster, multiple drones were operated to deliver parcels. The ground vehicle routing problem for minimizing the total delivery time was solved as a TSP by using a genetic algorithm (GA). [7] introduced the target allocation problem by a multi-drone system that delivers parcels from a warehouse to target locations. The authors presented the multi-objective optimization problem involving distance and flight time, parcel weight, and parcel volume. The problem was solved by using a GA. For these papers, a drone delivers a parcel from the distribution center or ground vehicle to customers at a time such that there was no need to optimize the multi-drone routes.

However, a drone could make multiple visits in a route. For a single-drone system, [8] introduced the drone routing problem to minimize the total fuel consumption. The problem was formulated as a MILP problem and solved using two heuristic algorithms based on the LKH and k-opt heuristics. In [9], a ground vehicle and a drone routes were optimized cooperatively to minimize the time for intelligence, surveillance, or reconnaissance tasks by using an integer linear program and two heuristic algorithms. [10] introduced a different use of the drone that resupply parcels to a ground vehicle from the fulfillment center while the ground vehicle delivers the parcel to customers. The ground vehicle routing problem with drone resupply was proposed and solved by using two heuristic approaches. For a multi-drone system, [11] proposed two multi-drone

routing problems to minimize the cost of drones and energy, and minimize the total delivery time, respectively, while a ground vehicle was not considered. The authors used a MILP model and simulated annealing algorithm to solve these problems. [12] considered multiple visits for heterogenous drones equipped with different sensors in surveillance. The multi-drone optimization problem was formulated using a MILP model. The author developed a branch-and-cut algorithm to solve the optimization problem. [13] studied a multi-truck, multi-drone, and multi-depot system in logistics operations. The author extended the PDSTSP by considering that, after a drone drops a parcel to customers, the drone can directly return to a depot or fly to another customer for pickup. The problem was solved with a constraint programming method. [14] considered nonlinear relationships between drones' battery energy consumption and flight distance in the objective function that is to minimize an overall transportation cost. The multi-drone routing problems were solved by branch-and-cut algorithms.

The main contributions of this chapter are as follows. First, we formulate the multi-drone and ground vehicle routing problem to minimize the total operation time for inspecting wind turbines. Second, the VLOS regulation is considered in our model to make it more practical. Third, we examine effects of total operation time, the number of swapping with new battery, and different parameters including the number of drones, number of clusters, and flight endurance for three case studies.

The remainder of this chapter is organized as follows. In Section 3.3, we describe the proposed optimization model and solution approach. Section 3.4 presents the experimental results of different parameters for three case studies. Finally, in Section 3.5, we state our conclusion.

## 3.3 Proposed optimization model and solution approach

To inspect wind turbines, we use *U* identical drones and one ground vehicle. We assume that there are *n* identical wind turbines and it take *s* minutes for inspecting each one by using one drone. The drones fly at a constant speed of *FS* mph and has a flight endurance of *FE* minutes (maximum flight time). To comply with the VLOS regulation, the drones cannot fly beyond *VS* miles. The drone can have multiple visits in a route. Also, the drone can have multiple routes by returning to the stop location of the ground vehicle where it takes off and flies again to inspect another wind turbine after swapping with new battery. The ground vehicle moves from cluster to cluster at a constant speed of *GS* mph while carries *U* drones and drone pilots, and enough number of the fully charged batteries. The ground vehicle can park at only one of the designated locations near a wind turbine. Thus, in our model, a wind turbine location is used as a stop location of the ground vehicle. At each stop, the drone pilots conduct pre and post-flight procedures for *p* minutes. The optimization model is decomposed into two steps, based on the rule "drones first, ground vehicle second". The first step optimizes the multi-drone routes in each cluster. The second step optimizes the ground vehicle route with outcomes from the first step.

### 3.3.1 Optimizing the multi-drone routes

The multi-drone routing problem considers the set $N = \{0, 1, 2, \dots, n\}$ that includes one depot (0) and *n* wind turbines $(1, 2, \dots, n)$. First, we group the wind turbines into *k* clusters by using the K-means clustering algorithm, also known as the Lloyd's algorithm [15]. The set of the wind turbines in each cluster is denoted by $N^a$ where $a = 1, 2, \dots, k$. To determine the value of the parameter *k*, we start by considering only one cluster ($k = 1$). Then, we examine whether a wind

turbine location is feasible to be used as a stop location of the ground vehicle by performing the following procedure. Suppose that five wind turbines are grouped into cluster 1. We can denote them by the set $N^1 = \{2, 3, 4, 10, 12\}$. We set the first wind turbine in $N^1$, which is wind turbine 2, as the depot wind turbine defined as the place where drones are deployed and returned in cluster 1. Then, we compute the distances and flight times from the depot wind turbine 2 to all the other wind turbines 3, 4, 10, and 12. If all distances (***D***) are less than or equal to *VS* miles and all fight times (***F***) are less than or equal to *FE* minutes, the location of the depot wind turbine 2 will be put into the set of the feasible stop locations of the ground vehicle denoted by *S*. We apply this procedure to the other wind turbines. After that, if all *k* clusters do not have at least one feasible stop location, we increase $k$ by 1 and repeat the clustering loop. Otherwise, we terminate the clustering loop and save two clustering outcomes. One outcome is the cluster set $K = \{1, 2, \dots, k\}$. Another outcome is the cluster assignment vector $\boldsymbol{C}$ that has $n + 1$ dimension. The first element in $\boldsymbol{C}$ always has 0 representing the depot. The rest of the elements in $\boldsymbol{C}$ indicate the cluster identification number assigned to each wind turbine. For example, if we obtain the cluster assignment vector $\boldsymbol{C} = [0, 5, 3, 4, 1, 2]$, it is interpreted that wind turbine 1 is assigned to cluster 5 and wind turbine 2 is assigned to cluster 3, and so on.

Once we determine k, we find the optimal operation time of the drones for each cluster by solving the multi-drone routing problem using the MILP model described in Eqs. (3.1) – (3.24). We formulate this optimization problem based on the work of [11].

$$\text{Min } T^D \tag{3.1}$$

Subject to

$$\sum_{\substack{j \in N^a \\ i \neq j}} z_{ij} = 1 \quad \forall i \in \{N^a : i \neq 0\} \tag{3.2}$$

$$\sum_{\substack{j\in N^a \\ i\neq j}} z_{ij} - \sum_{\substack{j\in N^a \\ i\neq j}} z_{ji} = 0 \quad \forall i \in N^a \tag{3.3}$$

$$\sum_{\substack{j\in N^a \\ j\neq 0}} r_{ij} \leq z_{i0} \quad \forall i \in \{N^a : i \neq 0\} \tag{3.4}$$

$$\sum_{\substack{j\in N^a \\ j\neq 0}} r_{ji} \leq z_{0i} \quad \forall i \in \{N^a : i \neq 0\} \tag{3.5}$$

$$\sum_{\substack{i\in N^a \\ i\neq 0}} z_{0i} - \sum_{\substack{(i,j)\in N^a\times N^a \\ i\neq 0 \\ j\neq 0 \\ i\neq j}} r_{ij} \leq U \tag{3.6}$$

$$t_i^{co} - t_j^{co} + s + \frac{d_{ij}}{v} \leq M \times \left(1 - z_{ij}\right) \quad \forall (i,j) \in N^a \times N^a, j \neq 0, i \neq j \tag{3.7}$$

$$t_i^{co} - e_i^{co} + s + \frac{d_{i0}}{v} \leq M \times (1 - z_{i0}) \quad \forall i \in \{N^a : i \neq 0\} \tag{3.8}$$

$$e_i^{co} - t_j^{co} + w + \frac{d_{0j}}{v} \leq M \times \left(1 - r_{ij}\right) \quad \forall (i,j) \in N^a \times N^a, i \neq 0, j \neq 0, i \neq j \tag{3.9}$$

$$t_0^{co} = 0 \tag{3.10}$$

$$e_i^{co} \leq M \times z_{i0} \quad \forall i \in \{N^a : i \neq 0\} \tag{3.11}$$

$$t_i^{co} + s + \frac{d_{i0}}{v} \leq T^D \quad \forall i \in \{N^a : i \neq 0\} \tag{3.12}$$

$$t_i^{nb} - t_j^{nb} + s + \frac{d_{ij}}{v} \leq M \times \left(1 - z_{ij}\right) \quad \forall (i,j) \in N^a \times N^a, j \neq 0, i \neq j \tag{3.13}$$

$$t_i^{nb} - e_i^{nb} + s + \frac{d_{i0}}{v} \leq M \times (1 - z_{i0}) \quad \forall i \in \{N^a : i \neq 0\} \tag{3.14}$$

$$t_0^{nb} = 0 \tag{3.15}$$

$$e_i^{nb} - FE \leq M \times (1 - z_{i0}) \quad \forall i \in \{N^a : i \neq 0\} \tag{3.16}$$

$$e_i^{nb} \leq M \times z_{i0} \quad \forall i \in \{N^a : i \neq 0\} \tag{3.17}$$

$$z_{ij} \in \{0,1\} \quad \forall (i,j) \in N^a \times N^a \tag{3.18}$$

$$r_{ij} \in \{0,1\} \quad \forall (i,j) \in N^a \times N^a \tag{3.19}$$

$$t_i^{co} \geq 0 \quad \forall i \in N^a \tag{3.20}$$

$$e_i^{co} \geq 0 \quad \forall i \in N^a \tag{3.21}$$

$$t_i^{nb} \geq 0 \quad \forall i \in N^a \tag{3.22}$$

$$e_i^{nb} \geq 0 \quad \forall i \in N^a \tag{3.23}$$

$$T^D \geq 0 \tag{3.24}$$

In this model, we re-index all wind turbines in a cluster ($N^a$) with the number starting from zero. The first wind turbine is the depot wind turbine and given by $N_0^a$. We define the decision variables as follows. Let $z_{ij} \in \{0,1\}$ equal one if the drones fly from wind turbine *i* to wind turbine *j*. Constraints (3.18) enforce that $z_{ij}$ is a binary variable. Let $r_{ij} \in \{0,1\}$ equal one if the drones fly from wind turbine *i* to the depot wind turbine, swaps with new battery, then flies to wind turbine *j* for a new route. Constraints (3.19) enforce that $r_{ij}$ is a binary variable. We track the cumulative operation time at each wind turbine. $t_i^{co}$ represents the time that wind turbine *i* is visited by drones. Also, we track the cumulative operation time at each wind turbine for each route. $e_i^{co}$ represents the time that drones return to the depot wind turbine directly from wind turbine *i* given each route. In other words, $e_i^{co}$ indicates the end time of the route by drones. To limit the flight time for each route by a battery capacity, we define two auxiliary decision variables. $t_i^{nb}$ represents the time that wind turbine *i* is visited by drones after swapping with new battery. $e_i^{nb}$ represents the time that drones return to the depot wind turbine directly from wind turbine *i* after swapping with new battery. Note that $t_i^{nb}$ and $e_i^{nb}$ are reset to zero when drones return to the depot wind turbine. Constraints (3.20) – (3.24) enforce that all time variables are greater than or equal to zero.

The objective function (3.1) minimizes the operation time by the drones, $T^D$, which represents the sum of the total flight time and inspection time of the drones over all wind turbines in a cluster. Constraints (3.2) ensure that drones visit each wind turbine exactly once, except for the depot wind turbine. Constraints (3.3) ensure that drones landing to the wind turbine *i* must also take off from the same wind turbine *i*.

Constraints (3.4) – (3.6) are about multiple routes of the drones by swapping with new battery at the depot wind turbine. Constraints (3.4) enforce that if drones return to the depot wind turbine from wind turbine *i*, the drones can fly to wind turbine *j* after swapping with new battery at the depot wind turbine. Constraints (3.5) enforce that if drones fly from the depot wind turbine to wind turbine *i*, the drones may previously return to the depot wind turbine from another wind turbine *j*. Constraints (3.6) limit the number of drones to *U*.

Constraints (3.7) – (3.12) are to track the cumulative operation time. Constraints (3.7) keep track of the time $t_i^{co}$ that wind turbine *i* is visited by drones with a sufficiently large number *M*. Constraints (3.8) keep track of the time $e_i^{co}$ that drones return to the depot wind turbine directly from wind turbine *i*. Constraints (3.9) ensure that $t_i^{co}$ and $e_i^{co}$ are cumulatively tracked after drones return to the depot wind turbine, swaps with new battery, and flies to another wind turbine. Constraints (3.10) enforce that the time at the depot wind turbine is zero. Note that constraints (3.10) no longer work after drones depart from the depot wind turbine because of constraints (3.9). Thus, constraints (3.10) enforce that only initial time at the depot wind turbine is zero. Constraints (3.11) enforce that if drones do not fly from wind turbine *i* to the depot wind turbine, then $e_i^{cb}$ is zero. The operation time by the drones is determined by constraints (3.12).

Constraints (3.13) keep track of the time $t_i^{nb}$ that wind turbine *i* is visited by drones after swapping with new battery. Constraints (3.14) keep track of the time $e_i^{nb}$ that drones return to the depot wind turbine directly from wind turbine *i* after swapping with new battery. Constraints (3.15) enforce that the time are set to zero whenever the drones return to the depot wind turbine. Because there are no corresponding constraints to constrains (3.9) for $t_i^{co}$ and $e_i^{co}$, the times are initialized to zero whenever drones return to the depot wind turbine, swaps with new battery, and flies to another wind turbine. Constraints (3.16) ensure that the flight time for each route must be less than the flight endurance. Constraints (3.17) enforce that if drones do not fly from wind turbine *i* to the depot wind turbine, then $e_i^{nb}$ is zero.

### 3.3.2 Optimizing the ground vehicle route

The ground vehicle routing problem is the equality generalized traveling salesman problem (E-GTSP) [16]. The objective is to minimize the operation time by the ground vehicle, $T^G$ by determining the shortest route that the ground vehicle departs from the depot location, visit all *k* clusters exactly once, and returns to the depot location. From the previous step, $T_i^D$ is obtained for wind turbine *i* of which location is in *S*. We set that $T_i^D$ is a sufficiently large number *M* for wind turbine *i* of which location is not in *S*. With this setting, we tweak the model introduced in Chapter 2 to solve the ground vehicle routing problem by including three additional terms in the objective function as follows.

$$\text{Min } T = \sum_{\substack{i \in N \\ i \neq j}} \sum_{\substack{j \in N \\ i \neq j}} t_{ij} x_{ij} + \sum_{\substack{i \in N \\ i \neq j}} \sum_{\substack{j \in N \\ i \neq j}} T_j^D x_{ij} + p \times k \tag{3.25}$$

Subject to Eqs. (2.7) – (2.18) in Chapter 2. The objective function (3.25) minimizes the total operation time (*T*) that is sum of the operation time of the ground vehicle (first term), the operation

time of the drones (second term), and the total time for conducting pre- and post-flight procedures over the *k* clusters (third term). The pseudocode of our solution approach is shown in Figure 3.1.

```
1: // Clustering
2: For k = 1 to n do
3:   Group wind turbines into k clusters by using a K-means clustering algorithm
4:   For a = 1 to k do
5:     For i = 1 to m do
6:       D ← dist(i, N^a)  // Compute the distances from the depot wind turbine i to all the other wind turbines in cluster a
7:       F ← ft(i, N^a)  // Compute the flight times from the depot wind turbine i to all the other wind turbines in cluster a
8:       If ( (max(D) ≤ VS) & (max(F) ≤ FE)) then
9:         Put the location of the depot wind turbine i into S
10:      End if
11:    End for
12:  End for
13:  If (all k clusters have at least one feasible stop location) then
14:    Save K and C
15:    Brake from the loop (line no from 2 to 17)
16:  End if
17: End for
18: // Optimizing the multi-drone routes
19: For i = 1 to n do
20:  If (the location of the wind turbine i is in S) then
21:    T_i^D ← optim_mdrones(i, C)  // Find the optimal operation time of the drones flying from wind turbine i
22:  Else
23:    T_i^D ← M  // Set the operation time of the drones flying from wind turbine i as a sufficiently large number
24:  End if
25: End for
26: // Optimizing the ground vehicle route
27: T ← optim_gv(T^D, C) + p × k  // Find the optimal total operation time
```

**Figure 3.1** Pseudocode of the solution approach

## 3.4 Experimental results

Two decomposed optimization problems are coded in MATLAB and solved using the CPLEX 12.8 solver. We use the K-means clustering function available in MATLAB. Because two problems are NP-hard problems, we solve them with a 5-min time limit. All case studies are run on a computer with an Intel® Core™ i5-8250U (1.60 GHz) processor with 8 GB of RAM memory.

### 3.4.1 System parameters and case studies

Table 3.1 shows the parameters of the system components including drone, ground vehicle, and regulation. To test our proposed models, we use same wind farms introduced in Chapter 2. Thus, a small-sized wind farm has 34 wind turbines across a 6-mile square region. A medium-sized wind farm has 72 wind turbines across a 15-mile square region. A large-sized wind farm has 100 wind turbines across a 55-mile square region. The depots are located approximately in the center of the wind farms.

**Table 3.1** System parameters

| | |
|---|---|
| Drone | |
| Flying speed (*FS*) | 40 mph |
| Flight endurance (*FE*) | 30 min |
| Inspection time for a wind turbine (*s*) | 5 min |
| Waiting time for swapping with new battery (*w*) | 2 min |
| Time for conducting pre- and post-flight procedures (*p*) | 5 min |
| Ground vehicle | |
| Moving speed (*GS*) | 20 mph |
| Regulation | |
| Visual line-of-sight (*VS*) | 2 miles |

### 3.4.2 Routing solutions

We use the same system parameters and solve the optimization model on the three case studies. The routing solutions when we use two drones and one ground vehicle are depicted in Figures 3.2 – 3.4. We mark the clusters with the shaded color. Figure 3.2 shows that the 34 wind turbines in the small-sized wind farm are clustered into 2 groups. Figure 3.3 shows that the 72 wind turbines in the medium-sized wind farm are clustered into 3 groups. Figure 3.4 shows that the 100 wind turbines in the large-sized wind farm are clustered into 6 groups.

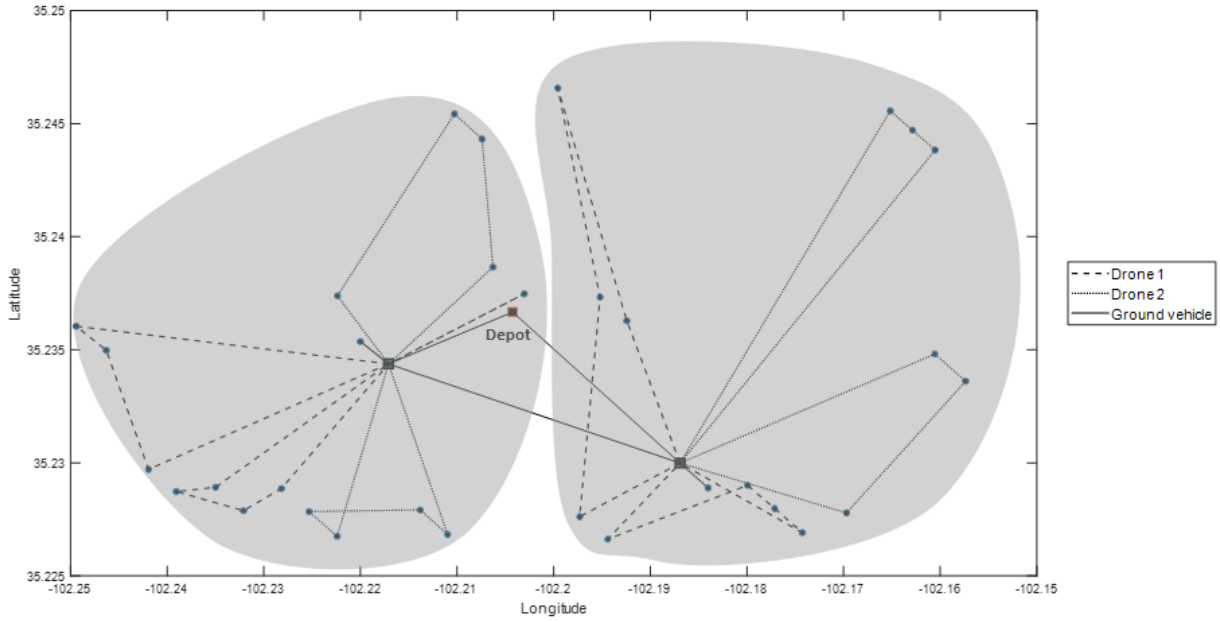



**Figure 3.2** Small-sized wind farm routing solution for two drones and one ground vehicle

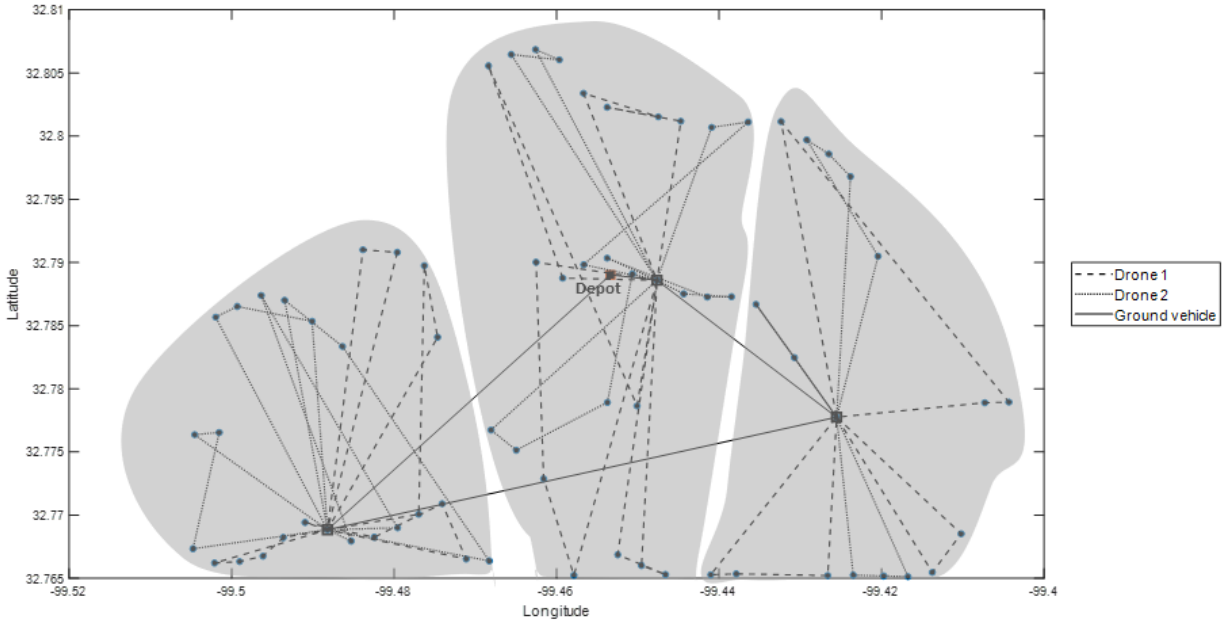



**Figure 3.3** Medium-sized wind farm routing solution for two drones and one ground vehicle

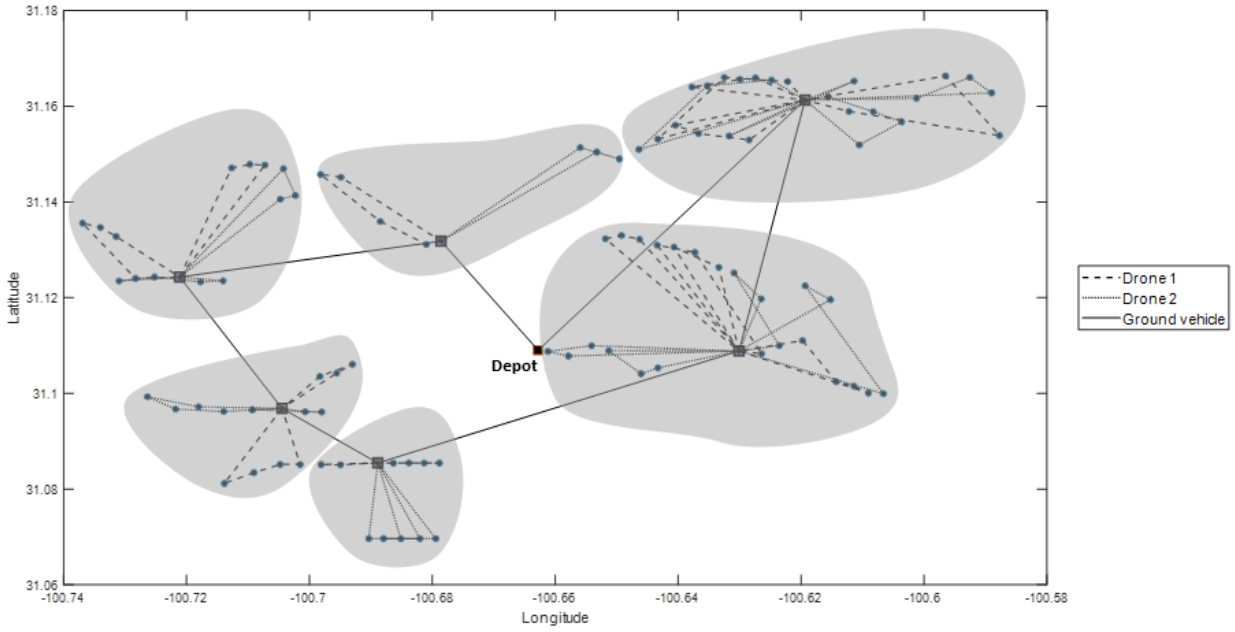



**Figure 3.4** Large-sized wind farm routing solution for two drones and one ground vehicle

### 3.4.3 Results of the total operation time

We test the optimization model on three case studies by changing the number of drones from two to five and examine the total operation time for three wind farms. As shown in Figure 3.5, the total operation time decreases as the number of drones increases. We obtain the most significantly reduced time when we use three drones instead of two drones. Also, we observe that, for the small-sized wind farm, using three drones (84 minutes) reduces the total operation time of using one drone (250 minutes) by two-thirds. Note that the result of using one drone is not reported for the medium- and large-sized wind farms because we cannot achieve feasible solutions with a 5-min time limit.

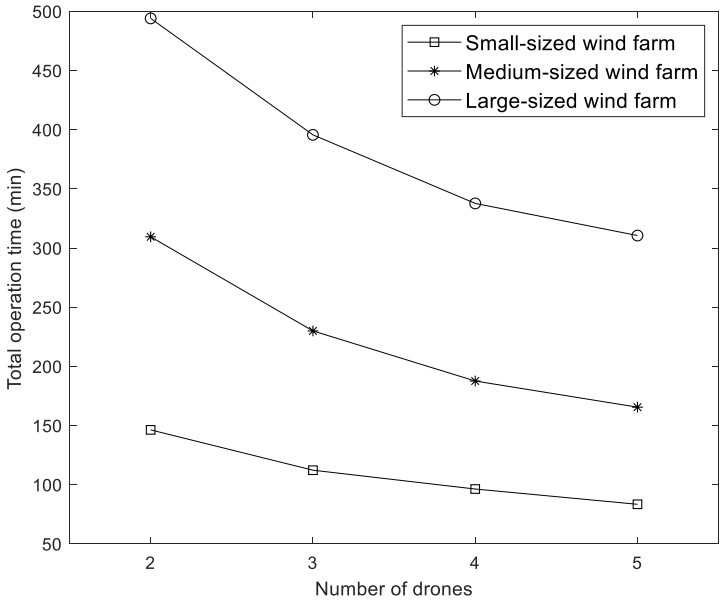



**Figure 3.5** Results of the total operation time

### 3.4.4 Results of swapping with new battery

We examine results of swapping with new battery to know that how many new batteries should be stored in the ground vehicle. When we inspect the small-sized wind farm with five drones, there is no need to carry new battery. However, when we inspect the large-sized wind farm with two drones, we should have 18 batteries to complete the inspection within the expected time. Notice that the numbers of swapping with new battery are not evenly distributed over clusters, except the case of using three drones for the small-sized wind farm.

**Table 3.2** Results of swapping with new battery

| | Number of drones | Cluster Index | Number of swapping with new battery | |
|---|---|---|---|---|
| Small-sized wind farm | 2 | 1 | 4 | 7 |
| | | 2 | 3 | |
| | 3 | 1 | 3 | 6 |
| | | 2 | 3 | |
| | 4 | 1 | 4 | 4 |
| | | 2 | 0 | |
| | 5 | 1 | 0 | 0 |
| | | 2 | 0 | |
| Medium-sized wind farm | 2 | 1 | 6 | 16 |
| | | 2 | 4 | |
| | | 3 | 6 | |
| | 3 | 1 | 6 | 15 |
| | | 2 | 3 | |
| | | 3 | 6 | |
| | 4 | 1 | 4 | 11 |
| | | 2 | 3 | |
| | | 3 | 4 | |
| | 5 | 1 | 5 | 11 |
| | | 2 | 0 | |
| | | 3 | 6 | |
| Large-sized wind farm | 2 | 1 | 2 | 18 |
| | | 2 | 6 | |
| | | 3 | 2 | |
| | | 4 | 6 | |
| | | 5 | 0 | |
| | | 6 | 2 | |
| | 3 | 1 | 3 | 18 |
| | | 2 | 6 | |
| | | 3 | 3 | |
| | | 4 | 6 | |
| | | 5 | 0 | |
| | | 6 | 0 | |
| | 4 | 1 | 0 | 8 |
| | | 2 | 4 | |
| | | 3 | 0 | |
| | | 4 | 4 | |
| | | 5 | 0 | |
| | | 6 | 0 | |

| | | | |
|---|---|---|---|
| 5 | 1 | 0 | 10 |
| | 2 | 5 | |
| | 3 | 0 | |
| | 4 | 5 | |
| | 5 | 0 | |
| | 6 | 0 | |

### 3.4.5 Effect of the number of stop locations

We study the effect of the number of stop locations (i.e. the number of clusters) on the total operation time when drones fly beyond VLOS for the small-sized wind farm. Figure 3.6 shows that we obtain continuously reduced total operation time as the number of stop locations decreases. Thus, the shortest total operation time are occurred when we use 2 stop locations. Note that there is no feasible solution for 1 stop location. In addition, we confirm that all total operation time are less than those with VLOS requirement. For example, it takes 72 minutes to inspect a wind farm in using five drones without VLOS requirement while it takes 84 minutes with VLOS requirement.

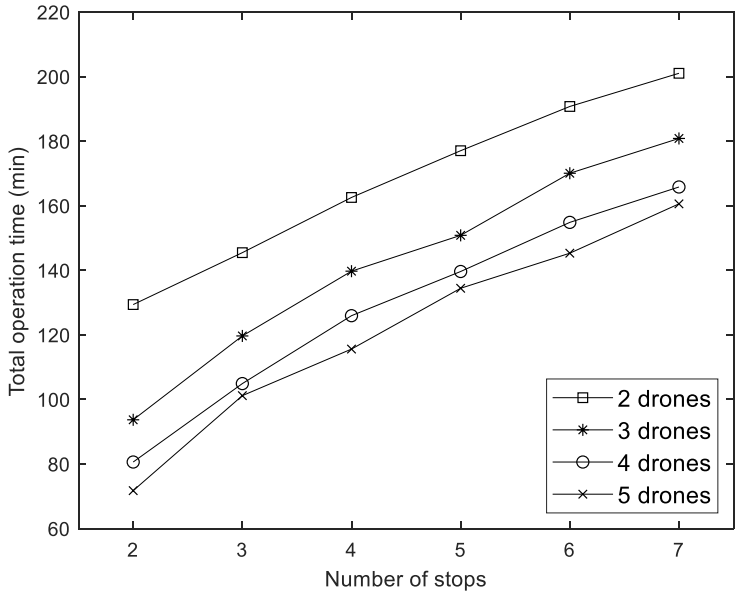



**Figure 3.6** Effect of the number of stops in the small-sized wind farm

### 3.4.6 Effect of the flight endurance

We study the effect of the flight endurance by changing it from 20 to 50 minutes in steps of 5 minutes. Figure 3.7 shows the results when we use two drones and one ground vehicle on three case studies. Overall, as we operate drones with longer flight endurance, the total operation time decreases except two instances: (i) the total operation time increases a little by 0.3 minutes when we change the flight endurance from 40 to 45 minutes for the small-sized wind farm and (ii) the total operation time increases a little by 0.5 minutes when we change the flight endurance from 45 to 50 minutes for the medium-sized wind farm. We achieve the biggest saving of the total operation time by changing the flight endurance from 20 to 25 minutes for all three wind farms. Notice that the total operation time remains almost unchanged with the flight endurance of more than 35 minutes.

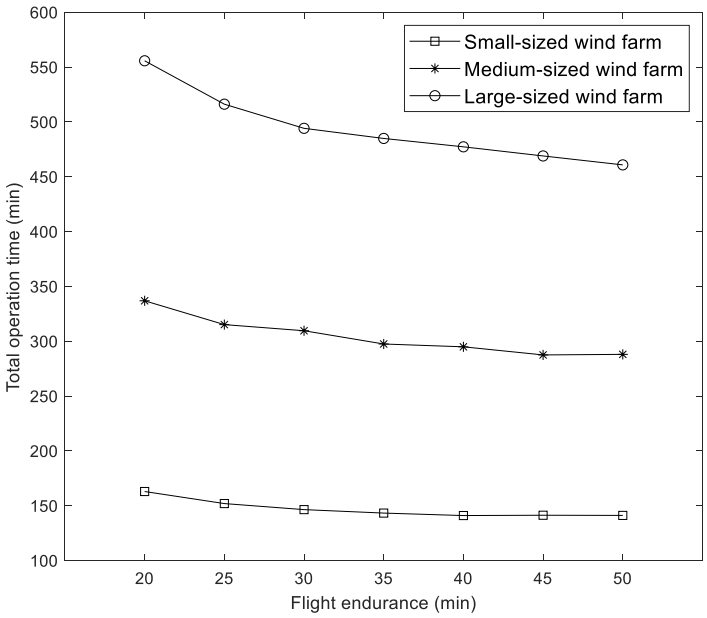



**Figure 3.7** Effect of changing the flight endurance for two drones and one ground vehicle

## 3.5 Conclusion

We introduced the optimization model for inspecting wind farms with a multi-drone system. One ground vehicle was used to carry drones to the wind turbines which are located beyond the maximum flight distance, resulting that all wind turbines are inspected. The model considered the VLOS regulation to give practical applications. We solved the optimization model by decomposing two steps which are optimizing the multi-drone routes and ground vehicle route. We tested the model on three case studies. The routing solutions confirmed that drones are used much more than the ground vehicle while minimizing the number of stop locations. The results of the total operation time showed that we obtain the shortest total operation time when we use the maximum number of drones. The effect of swapping with new battery guided us how many batteries should be prepared. The effect of the number of stop locations showed optimal solutions when drones fly beyond VLOS. The effect of the flight endurance implied the appropriate choice of the drones' flight endurances given a budget and operation time constraints. Overall, these studies inform how to efficiently inspect wind farms by using drone technology including the optimal multi-drone routes, ground vehicle route, number of batteries, and drone specifications.

## References


[1] P. Cohn, A. Green, M. Langstaff, and M. Roller, "Commercial drones are here: The future of unmanned aerial systems," *McKinsey & Company*, Dec. 2017. [Online] Available: https://www.mckinsey.com/industries/capital-projects-and-infrastructure/our-insights/commercial-drones-are-here-the-future-of-unmanned-aerial-systems?cid=other-eml-alt-mip-mck-oth-1712. [Accessed May 3, 2019].

[2] D. Chabot, "Trends in Drone Research and Applications as the Journal of Unmanned Vehicle Systems Turns Five," *Journal of Unmanned Vehicle Systems*, 6(1), pp. vi-xv, 2018.

[3] U.S. Federal Aviation Administration, "Summary of small unmanned aircraft rule (part 107)," 2016. [Online]. Available: https://www.faa.gov/uas/media/Part_107_Summary.pdf. [Accessed May 3, 2019].

[4] T. Bektas, "The multiple traveling salesman problem: an overview of formulations and solution procedures," *Omega*, 34, pp. 209-219, 2006.

[5] C. C. Murray and A. G. Chu, "The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery," *Transportation Research Part C: Emerging Technologies*, 54, pp. 86-109, 2015.

[6] S. M. Ferrandez, T. Harbison, T. Weber, R. Sturges, and R. Rich, "Optimization of a truck-drone in tandem delivery network using k-means and genetic algorithm," *Journal of Industrial Engineering and Management*, 9(2), pp. 374-388, 2016.

[7] K. T. San, E. Y. Lee, and Y. S. Chang, "The delivery assignment solution for swarms of UAVs dealing with multi-dimensional chromosome representation of genetic algorithm," in *Proceedings of the IEEE 7th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference*, 2016.

[8] K. Sundar and S. Rathinam, "Algorithms for routing an unmanned aerial vehicle in the presence of refueling depots," *IEEE Transactions on Automation Science and Engineering*, 11(1), pp. 287-294, 2014.

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

[10] I. Dayarian, M. Savelsbergh, and J-P Clarke, "Same-day delivery with drone resupply," *Optimization Online*, 2017.

[11] K. Dorling, J. Heinrichs, G. G. Messier, and S. Magierowski, "Vehicle routing problems for drone delivery," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(1), pp. 70-85, 2017.

[12] K. Sundar and S. Rathinam, "Algorithms for heterogeneous, multiple depot, multiple unmanned vehicle path planning problems," *Journal of Intelligent & Robotic Systems*, 88(2-4), pp. 513-526, 2017.

[13] A. M. Ham, "Integrated scheduling of *m*-truck, *m*-drone, and *m*-depot constrained by time-window, drop-pickup, and *m*-visit using constraint programming," *Transportation Research Part C*, 91, pp. 1-14, 2018.

[14] C. Cheng, Y. Adulyasak, and L-M Rousseau, "Formulations and exact algorithms for drone routing problem," Technical report, 2018.

[15] S. Lloyd, “Least squares quantization in PCM,” *IEEE Transactions on Information Theory*, 28(2), pp. 129-137, Mar. 1982.

[16] M. Fischetti, J. J. S. González, and P. Toth, “A branch-and-cut algorithm for the symmetric generalized traveling salesman problem,” *Operations Research*, 45(3), pp. 378-494, 1997.

## Conclusion

We introduced three optimization models for the inspection of electric power system assets using drones. First, we proposed a PSO-based algorithm and a SA-based algorithm to find an efficient drone flight path for inspecting electric transmission towers involving three performance ratios: (i) flight time, (ii) image quality, and (iii) tower coverage. The experimental results showed that the PSO-based algorithm outperforms the SA-based algorithm in terms of the average total performance ratio and computational time. Also, it showed that the proposed model can provide a flight path that comprises a good balance over the three performance ratios.

Second, we proposed a routing optimization model for a drone and ground vehicle to reduce the total operation time for inspecting a set of wind turbines. We solved the optimization model by using the K-means clustering algorithm and two integer linear programming formulations. The experimental results showed that the proposed model reduces the total operation time of the inspection for the three wind farm sizes when compared to no clustering or the use of a greedy approach for determining the route of the ground vehicle. In addition, studying effects of the flight speed and endurance showed the robustness of our approach to changes to the system parameters.

Third, we proposed the optimization model for inspecting wind farms with a multi-drone system and considered the VLOS regulation to give practical applications. We solved the optimization model by decomposing two steps which are optimizing the multi-drone routes and

ground vehicle route. The routing solutions confirmed that drones are used much more than the ground vehicle while minimizing the number of stop locations. Overall, our studies inform how to efficiently inspect electric transmission towers and wind farms by using drone technology. Moreover, given a budget and technical constraints to operate the drone, these results can be of use to practitioners to determine the technical specifications of the drone.

## Appendix A: Nomenclature for Chapter 1

*Indices*

| | |
|---|---|
| $i$ | Particle (flight path) in the PSO |
| $j$ | Waypoint in the particle |
| $t$ | Time step |

*Sets*

| | |
|---|---|
| $\Phi$ | Set of all coordinates within the flying area |
| $\Phi_S$ | Set of surface coordinates of the tower |
| $\Phi_N$ | Set of coordinates of the no-fly zone |
| $\Phi_C$ | Set of candidates' coordinates for a waypoint $\Phi_C = \Phi - \Phi_N$ |

*Parameters*

| | |
|---|---|
| **D** | Maximum *x*, *y*, and *z* of the flying area |
| **Ω** | Weight vector of the objective function $(\omega_1, \omega_2, \omega_3)$ |
| **e** | Unit vector of each coordinate in a search space |
| $FS$ | Flying speed of the drone (m/s) |
| $H$ | Hovering time of the drone at a waypoint (sec) |
| $FT^{max}$ | Maximum flight time (sec) |
| $FD^{max}$ | Maximum flight distance (m) |
| $N^s$ | Total number of surface elements of the tower |
| $(X^{tc}, Y^{tc}, 0)$ | Tower center-base coordinates (m) |
| $G^{max}$ | Maximum ground sampling distance (mm) |
| $FL$ | Focal length of the camera (mm) |
| $L$ | Height of the image sensor (mm) |
| $N^p$ | Vertical number of pixels in the image |
| $N^{max}$ | Maximum number of waypoints |
| $N$ | Number of waypoints |
| $M$ | Number of particles |
| $C_1, C_2$ | Learning factors |
| $W$ | Inertial weight |
| $T_0$ | Initial temperature |
| $N_S$ | Number of cycles |
| $N_T$ | Number of step vector adjustments |
| $N_\varepsilon$ | Number of successive temperature reductions |
| $N_{eval}$ | Maximum number of objective function evaluations |

| | |
|---|---|
| $r_T$ | Reduction coefficient |
| $\varepsilon$ | Tolerance for stopping iterations |

*Variables*

| | |
|---|---|
| $\mathbf{fp}_i$ | Position of flight path (particle) *i* |
| $\mathbf{wp}_{i,j}$ | Coordinates of waypoint *j* in flight path (particle) *i* |
| $\mathbf{fv}_i$ | Velocity of particle *i* |
| $\mathbf{wv}_{i,j}$ | Velocity of waypoint *j* in particle *i* |
| $\mathbf{R}$ | Rotation matrix |
| $\mathbf{T}$ | Translation matrix |
| $\mathbf{TR}$ | Transformation matrix |
| $\mathbf{P}_i$ | Position of particle *i* |
| $\mathbf{V}_i$ | Velocity of particle *i* |
| $\mathbf{PB}_i^t$ | Best position of particle *i* at time *t* |
| $\mathbf{GB}^t$ | Position of the global best particle at time *t* |
| $\mathbf{s}$ | Step vector |
| $\mathbf{x}$ | Decision variable vector |
| $\mathbf{x}'$ | New decision variable vector |
| $pr_T$ | Total performance ratio of a flight path |
| $pr_1$ | Performance ratio for the total flight time |
| $pr_2$ | Performance ratio for the inspection coverage |
| $pr_3$ | Performance ratio for the image resolution |
| $ft_i$ | Flight time of particle *i* |
| $fd_i$ | Flight distance of particle *i* |
| $wd_i$ | Flight distance from the drone home-base to the last waypoint of particle *i* |
| $S^{rc}$ | Real coordinates (3D) of a tower surface element |
| $S^{cc}$ | Camera coordinates (3D) of a tower surface element |
| $S^{ic}$ | Image coordinates (2D) of a tower surface element |
| $(x, y, z)$ | Real coordinates |
| $(u, v, w)$ | Camera coordinates |
| $(x', y')$ | Image coordinates |
| $n_i^s$ | Total surface elements covered by the images taken during flight path (particle) *i* |
| $g_{i,j}$ | Ground sampling distance of the images taken at waypoint *j* in particle *i* |
| $v^{max}$ | Maximum velocity of a particle |
| $r, r_1, r_2$ | Random numbers in [0,1] |
| $T$ | Temperature |
| $p$ | Acceptance probability |
| $m$ | Number of accepting a new decision variable vector |

## Appendix B: Nomenclature for Chapter 2

*Indices*

| | |
|---|---|
| $i, j, l$ | Wind turbines |
| $a, b$ | Clusters |

*Sets*

| | |
|---|---|
| $N$ | Set of the wind turbines and depot |
| $N^a$ | Set of the wind turbines in cluster $a$ |
| $S$ | Subset of $N^a$ |
| $K$ | Set of clusters |

*Parameters*

| | |
|---|---|
| $FS$ | Flying speed of the drone (mph) |
| $FE$ | Flight endurance of the drone (min) |
| $GS$ | Moving speed of the ground vehicle (mph) |
| $n$ | Number of wind turbines |
| $s$ | Inspection time for a wind turbine (min) |
| $p$ | Time for conducting pre- and post-flight procedures (min) |

*Variables*

| | |
|---|---|
| $\mathbf{C}$ | Cluster assignment vector |
| $T$ | Total operation time (min) |
| $T^D$ | Operation time of the drone (min) |
| $T^G$ | Operation time of the ground vehicle (min) |
| $k$ | No of clusters |
| $m$ | Number of wind turbines in a cluster |
| $x_{ij}$ | Binary decision variable that equals 1 if the ground vehicle moves from wind turbine $i$ to wind turbine $j$, and 0 otherwise |
| $y_{ab}$ | Binary decision variable that equals 1 if the ground vehicle moves from cluster $a$ to cluster $b$, and 0 otherwise |
| $z_{ij}$ | Binary decision variable that equals 1 if the drone flies from wind turbine $i$ to wind turbine $j$, and 0 otherwise |
| $u$ | Integer variable representing the sequence number in which cluster is visited by the ground vehicle |
| $t_{ij}$ | Moving time between wind turbine $i$ to wind turbine $j$ by the ground vehicle |
| $f_{ij}$ | Flight time between wind turbine $i$ to wind turbine $j$ by the drone |

## Appendix C: Nomenclature for Chapter 3

*Indices*

| | |
|---|---|
| $i, j, l$ | Wind turbines |
| $a, b$ | Clusters |

*Sets*

| | |
|---|---|
| $N$ | Set of the wind turbines and depot |
| $N^a$ | Set of the wind turbines in cluster $a$ |
| $K$ | Set of clusters |
| $S$ | Set of the feasible stop locations of the ground vehicle |

*Parameters*

| | |
|---|---|
| $FS$ | Flying speed of the drone (mph) |
| $FE$ | Flight endurance of the drone (min) |
| $GS$ | Moving speed of the ground vehicle (mph) |
| $U$ | Number of drones |
| $M$ | Sufficiently large number |
| $n$ | Number of wind turbines |
| $s$ | Inspection time for a wind turbine (min) |
| $w$ | Waiting time for swapping with new battery (min) |
| $p$ | Time for conducting pre- and post-flight procedures (min) |

*Variables*

| | |
|---|---|
| $\mathbf{C}$ | Cluster assignment vector |
| $\mathbf{D}$ | Distance vector between the depot wind turbine and other wind turbines in a cluster |
| $\mathbf{F}$ | Flight time vector between the depot wind turbine and other wind turbines in a cluster |
| $T$ | Total operation time (min) |
| $T^D$ | Operation time of the drone (min) |
| $T^G$ | Operation time of the ground vehicle (min) |
| $k$ | No of clusters |
| $x_{ij}$ | Binary decision variable that equals 1 if the ground vehicle moves from wind turbine $i$ to wind turbine $j$, and 0 otherwise |
| $y_{ab}$ | Binary decision variable that equals 1 if the ground vehicle moves from cluster $a$ to cluster $b$, and 0 otherwise |
| $z_{ij}$ | Binary decision variable that equals 1 if the drone flies from wind turbine $i$ to wind turbine $j$, and 0 otherwise |

| | |
|---|---|
| $r_{ij}$ | Binary decision variable that equals 1 if the drone flies from wind turbine *i* to the depot wind turbine, gains a new battery, then flies to wind turbine *j* for a new route, and 0 otherwise |
| $u$ | Integer variable representing the sequence number in which cluster is visited by the ground vehicle |
| $t_{ij}$ | Moving time between wind turbine *i* to wind turbine *j* by the ground vehicle |
| $t_i^{co}$ | Time that wind turbine *i* is visited by a drone |
| $t_i^{nb}$ | Time that wind turbine *i* is visited by a drone after swapping with new battery |
| $e_i^{co}$ | Time that a drone returns to the depot wind turbine directly from wind turbine *i* |
| $e_i^{nb}$ | Time that a drone returns to the depot wind turbine directly from wind turbine *i* after swapping with new battery |