MIDDLE EAST TECHNICAL UNIVERSITY

INDUSTRIAL ENGINEERING DEPARTMENT

ENGINEERING MANAGEMENT PROGRAM

EM 599 - TERM PROJECT

FINAL REPORT

Decision Support System for Unmanned Aerial Vehicle Usage in Search and Scanning Operations

Submitted by : Hürol Volkan Kaşıkaralar

Advisor : Prof. Dr. Cem İyigün

11.07.2021

Table of Contents

Table of Figures

1. Introduction

Unmanned aerial vehicles (UAV) are one of the reforming inventions of the 21st century. Although they are still a developing product, they have become an indispensable part of daily life in the military sense. However, they started to be a part of daily civilian life too. UAVs started to be used in search and rescue and aerial scanning operations with a limited number. As the world develops, more and more UAVs will participate in operations collaboration with other UAVs. This collaboration will require efficiency, as so optimization systems will be needed for UAV operations. This paper suggests a **decision support system (DSS)** for the UAV operators to solve optimization problems in the field without requiring any advanced mathematical computation. DSS will use the precalculated models and allow an operator to decide on the requirements of operations by selecting the time to complete the operation. Therefore, this paper will suggest DSS and explain how it can be designed, computed, and give examples for possible scenarios.

1.1. Company Information

Selvi Technology was founded in 2010 in Ankara. At an early age, the company focused on web-based software solutions and customer-oriented software programs. In 2017 company decided to invest in drone technologies and machine learning development.

Today, the company conducts machine learning-based projects to leading companies of Turkish defense industries and civil corporations. Some of these projects aim to improve operators' safety, stability of equipment, and security of assets. Having knowledge of drone technologies, the company produces various products and participates in various projects for the **Presidency of Defence Industries (SSB)**. These products and projects aim to gain stability to the assets currently operating in **the Turkish Armored Forces (TAF)** and add new equipment to the TAF's inventory.

1.2. Project Motivation

UAVs will take part in search and rescue and aerial scanning operations with a higher number than today. There will be fleets and squadrons of UAVs fly together during these operations. Governments, businesses, and operators will require optimized time planning, path planning, and equipment. However, one of the downsides of creating optimization models for these operations is that generating a mathematical model is complex and hard to implement. Additionally, search and rescue operations need quick decision-making and acting. Waiting for an engineer to analyze and create a model is unacceptable. The motivation behind creating a DSS is to give required optimized values with close approximations. As so, operators can decide on the content of the operations quickly on the field. Even if the DSS system cannot give them an exact solution, it can share tolerable results.

2. Problem Definition

Search and rescue operations are among the most critical topics for countries, diplomatic organizations, and private businesses. These types of operations mostly conduct during an unwanted event or a disaster. Since a disaster can occur at any moment in time, operators must be ready almost all the time. The purpose of these operations is to reach the injured persons as soon as possible to save lives. These operations can vary from mountain rescue to building rescue; however, this paper concerns air to surface search operations.

Today, most air to surface search operations conduct with helicopters (Figure 2) or planes (Figure 1) to find the victims. Helicopters are suitable for almost all environments because of their ability to move with six degrees of freedom; however, planes are usually included for locating large vessels in open areas, such as ships in the open waters. There are several downsides to using these vessels. Firstly, they have high operating and maintaining costs. Because of the rough conditions of search and rescue operations, most vessels preferred to conduct these operations are military-grade, which makes them costly to operate and non-profit businesses. So only countries, internationally supported organizations, and several high-level companies conduct these operations. Secondly, there should be back up equipment in case of multiple misfortune, and they must always be ready too, which increases costs. Thirdly, the personnel should be well trained. These operations need craftmanship, a kind of profession. Fourthly, time of the day. Air search operations need to be conducted in daylight because of the visibility and safety of the operators. Night operations acquire additional equipment such as thermal cameras and sensors, which is costly, and the most compatible pieces of equipment are military grade. Equipment class creates additional acceptance requirements since only a few countries sell that kind of equipment. Day light constraint effect limits of operatable time almost haft of a day to three-quarters of a day depending on the season. Fifthly, weather conditions are decisive to participate aerial vehicles to join the search operations. If the weather is dangerous to operate aerial vehicles, those vehicles stay in the airstrips or hangars. Because of these constraints, air vessels include in search and rescue operations if no other way or additional support depends on the size of the disaster. Even if air vessels are included in a search operation, they are insufficient because of a lack of numbers.

Figure 2. Rescue Helicopter

Figure 1. Rescue Plane

Areal scanning is one of the missions that require air to surface screening operations. These operations variate from businesses to hobbies and scientific research to military operations. Examining a construction site, analyzing a sizeable ranch, screening a military operation can be given as an example to these operations.

Today, most of the areal scanning operations use private small-size helicopters or planes. Since there is no rush to scan the entire area in a limited time, the most concerning constraint are money. Private companies conduct these operations to gain profit. Some pioneer companies, who invested in UAVs in the past, use drones to conduct these operations. However, since there are no fully comprehensive regulations, most companies hesitate to invest or use these UAV technologies, and the ones who have UAV technologies use them if there is no legal binding.

UAVs (Figure 3) is one of the rising technologies of the 21st century. Its primary purpose is to remove the dependency of an onboard pilot inside the aircraft. Countries such as the US and Israel invested in these technologies to create unmanned warplanes. Early designs could only be used in surveillance and tracking operations with high operations costs and ineffective capabilities. As the development continued, weaponized UAVs were created. These devices gave the ability to the air forces to replace some missions of warplanes.

Figure 3. MQ-9 Reaper

United States Air Force (USAF) significantly decreased the cost of surveillance and support missions by using UAVs in Iraq and the Afghan war. As they continued to invest in UAVs, they integrated such systems that allowing to control UAVs from another side of the planet by active satellite control, which allowed the army to deploy fewer personnel in the war zone and build fewer airbases. US Air Forces' active usage in Iraq and Afghanistan helped the world understand the importance of UAVs. Most countries decided to purchase these systems from the producing countries even if they are not in an active war. UAV systems were expensive; however, it was less costly than operating a warplane, and it was easy to train UAV pilots than a warplane pilot. Countries preferred losing a UAV rather than losing a warplane and pilot. Countries that were not able to purchase from the US or Israel started their UAV projects. Also, the ones who see purchasing from US or Israel pricy and who are unwilling to give their independent decisions on producing countries consent started their projects. As in today, many nations in the world started to produce their UAVs, such as China, Russia, and Turkey. Losing the monopoly of

producing UAVs, the US started increasing investments and developed further projects, such as Boeing MQ-25 Stingray as unmanned aerial refueling drones and Boeing X-37 (Figure 4) unmanned spaceplane.

Figure 4. Boeing X-37

On the other hand, increasing shareholders in the UAV market, competition, and mass production decreased the costs of UAVs significantly today. The decrease in the cost allowed the usage of UAVs in civil missions such as search and rescue and areal scanning. Unweaponized systems started to purchase by private organizations, and militaries started to help civil missions. However, civil business lacks personnel and equipment since most of the education centers and producers controlled by militaries. Nonetheless, the civilian usage of UAVs will increase in the following years.

Furthermore, UAVs can contribute to the search and rescue operations and areal scanning missions practically. Since UAVs are much cheaper to purchase and operate than a helicopter or a plane, there can be multiple of them in a single operation. Also, they can take place in more operations, extending the air to surface missions. For search and rescue operations, more UAVs can be deployed than helicopters or planes, which decreases the search time and allows authorities to help victims of the disaster quickly. UAVs can totally replace the planes and help to decrease helicopter usage by only extracting victims or send personnel to the ground. Having the ability to operate in numerous numbers simultaneously also decreases the effect of the daytime. Also, recent technological advancements allow UAVs to launch without an airstrip. They can be flown by throwing them from the hand (Figure 5) or a catapult. Catapult technology increases the number of flyable areas. However, one bad side to replace man piloted air vehicles with UAVs is that weather conditions are much more decisive for UAVs since they are more affected than the helicopters and planes to the bad weather conditions. The reason for that is they have significantly weaker equipment than helicopters or planes.

Figure 5. RQ-11 Hand Launch

To be able to use UAVs in the search and rescue and aerial scanning operations, there should be at least one temporary established base to control and maintain the UAVs. In these bases' UAVs can be thrown, collect and refuel during the operations. There can be multiple UAVs at air simultaneously, as shown in Figure 6, and UAVs who have depleted fuel can return to base and be thrown again after the refueling. There are several vital facts while conducting these operations. Firstly, UAVs should return to the base after completing their mission or has a critical problem like a malfunction or low fuel. As shown in Figure 6, every UAV returns to the base eventually. Secondly, there is no need to scan an area if another UAV already scanned the area. As shown in Figure 6, each UAV returns to the base after completing a scan on the unscanned areas assigned to them. Not scans the already scanned fields. Thirdly, UAVs should reach the unscanned areas as soon as possible to reduce the time on the road and reduce fuel consumption. In Figure 6, every UAV flies to the target area using the shortest path. Fourthly, the areas, which UAVs should scan might have different structures. Figure 6 the forested land needs more time to scan than the plain field and the sea. Fifth, it is inefficient to return a UAV to base after completing a scan in an area if it still has a considerable amount of fuel in its depot. That UAV can be used to scan another area before returning to the bases. In Figure 6, one of the UAVs' starts scanning the plain field after completing a scan on the sea. Sixth, multiple UAVs can scan the same area at the same time to reduce spent time. In Figure 7, two UAVs search for the forested field to complete scanning the area in less time. Black and blue arrows indicates separate UAVs. Seventh, if an area has a dangerous environment for the UAVs, those areas must be left behind. Because if an area is too dangerous to operate, then there is no reason to lose a UAV in that area. More advance aerial vehicles can conduct operations in those areas, such as helicopters. In Figure 8, operators decided to skip scanning mountain regions to secure the integrity of the UAVs. Eighth, operators must leave those regions behind if an area is far from the UAVs' operatable range. As in Figure 8, the wasteland was skipped by the operator because of the range issues. However, if those regions must be searching too, then base location should be selected considering all areas or another base can be established, as shown in Figure 9.

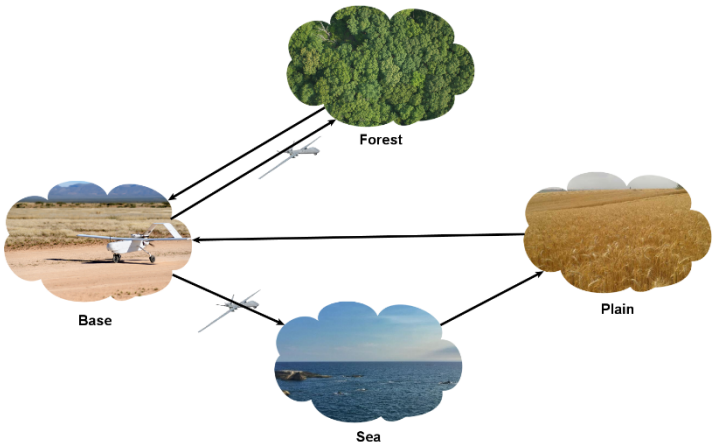

Figure 6. Representation of the Surveillance Area with Unique Paths for Area

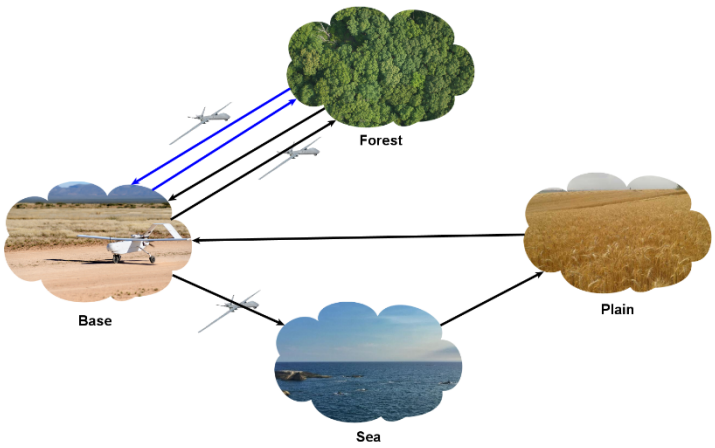

Figure 7. Representation of the Surveillance Area with Multiple Paths for Area

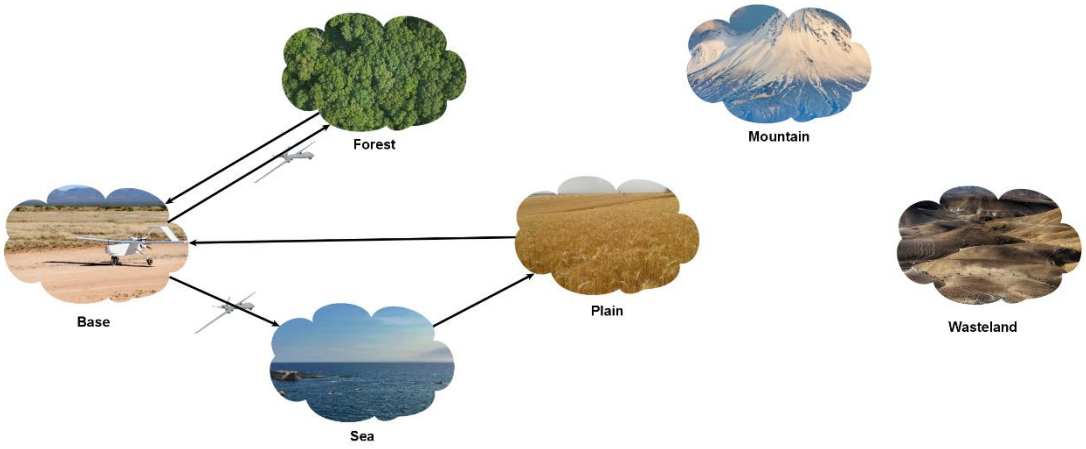

Figure 8. Representation of the Surveillance Area without Coverable Area

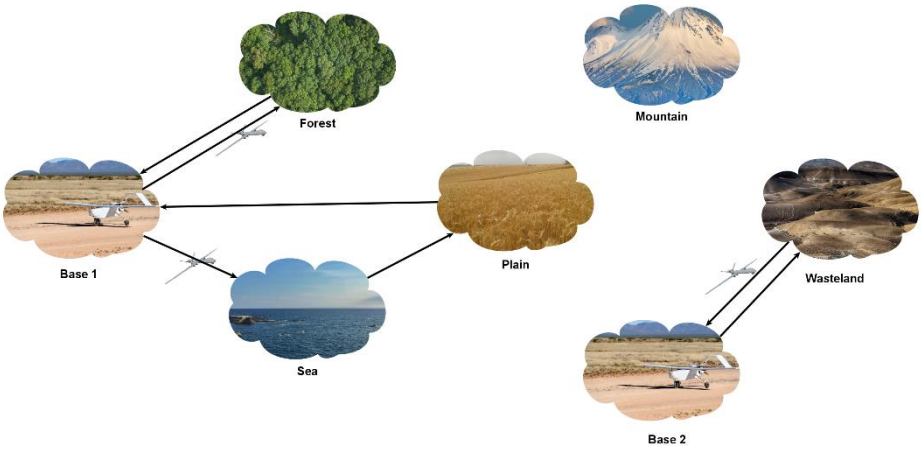

Figure 9. Representation of the Surveillance Area with Multiple Bases

Actually, operating multiple vehicles is much complicated than operating a single one. As so, operators on the ground should select how many UAVs they should include in the operations. For search and

rescue operations, operators should choose the number of UAVs that scan the entire operation area in a minimum time. On the other hand, for aerial scanning, the minimum cost of the mission is the main decision parameter since establishing a base for UAVs, paying salaries for operators, and maintaining the UAVs adds additional costs to the mission. So that, operating with one UAV for the entire mission probably is not the most profitable option. Also, there can be a time limitation for operators to scan the entire area. Operators for both types of operations must find the optimal solution depending on their requirements. They need to know every aspect of the operation, such as operating area size, type of the fields, and available flight time of the UAVs, and dispersion of the areas. Considering all the variables, operators should create a system that gives them the optimal solution to reach their goals. Since these types of problems are engineering problems, operators should hire an engineer to do the work, which costs time and money, most of the time infeasible.

This paper suggests optimizing operation in the field, which is a DSS. Since operators already know the number of UAVs they have, their flight time capabilities, and their simple guess about the condition of the operation field, they can learn the rough guess about operation completion time. DSS aims to approximate the completion time of operation by checking the known parameters and allowing operators to choose how many UAVs they should operate for the operation. For search and rescue operations, an operator can minimize the search time by using an efficient number of UAVs. For aerial scanning operations, an operator can calculate the minimum cost of operation by using the approximated operation time output of the DSS.

2.1. Problem Modeling

The first, to create the DSS, the problem defined needed to be solved using mathematical models, which requires the problem to be analyzed in detail and convert into a mixed-integer programming (MIP) model. MIP model can be created in such a way that it can calculate the optimal completion time of an operation depending on the given variables. Calculating all probable combinations can be sufficient to create the DSS. However, the accuracy of the DSS in the field depends on the complexity of the model created, such as including physical effects like stochasticity of the wind, probably of malfunctioning parts, the velocity of the UAVs, and perfect necessary time estimations of the fields. This paper will assume some of the complex effects as non-binding and simulate some effects using known probabilistic distributions.

Undoubtedly, to create a reliable MIP model, similar articles, which have similar aspects with the defined problem analyzed. According to the articles found, the defined problem can be classified as a **multi-trip cumulative capacitated vehicle routing problem** (MT-CCVRP). Since problems, the main goal is to use multiple UAVs, which can make multiple flights over the mission, satisfies the multi-trip addition. Not visiting an already scanned area indicates the cumulative as adding the time flown into the satisfied search time, in other words, decreasing the area necessary to search. Capacitated indicates the

capacity of the UAVs, such as their possible flyable time. Vehicle routing is the label for UAVs to travel between the sides. Depending on the articles, MT-CCVRP problems are primarily created and solved for logistical and supplement operations. However, MT-CCVRP can solve the search and rescue operations [1] and is adaptable to aerial scanning operations. According to the MT-CCVRP, the total area that requires operation can be represented as a grid map, and single fields that require scanning can be represented as nodes of the grid map [2]. Necessary time can be given to the nodes to represent the necessary time to complete scanning on that fields [2]. The necessary time given to that fields can be proportional to the density of the field. Additionally, adding time information to the roads between the nodes is essential to make the model more realistic since UAVs will consume fuel, which is flyable time, during fling on the way between the nodes [2]. Allowing UAVs to pass from one node to another node allows a single UAV to scan multiple nodes that need scanning. Passing feature can decrease the needed UAV number during the operation and eliminate the unnecessary fuel consumption at traveling between the nodes [2]. The optimal solutions for operations can be gathered by minimizing the travel time between nodes considering reaching the maximum number of the nodes [3].

After identifying the problem and highlighting several important aspects of the model, several assumptions and ignores had to be made to reduce the real-life problem into a solvable MIP model. Most of these assumptions are considered because of the need to create a table for DSS, which means the MIP model needs to be calculatable with different values multiple times. Each assumption is given below:

- Grid map (Figure 11) of an area assumed to have a nodes-based grids (Figure 10).

Figure 11. Grid Map

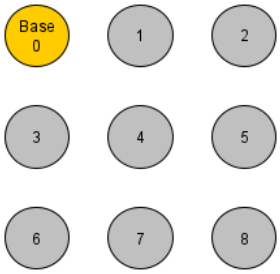

Figure 10. Grid Map Representation as Nodes

- Node map has a shape of a square. Every map has the same number of rows and columns. Figure 11 is a suitable example since it is a three-by-three square map. Square assumption was taken into consideration to decrease the complexity of the model since the MIP model wanted to generate multiple sizes of nodes and even the maps with rectangle shape needed additional parameters. Scalability was the primary concern.
- There is only one base location in every model, and it is located on the top left corner of the map. It is the first node on every map. This assumption is considered to reduce the complexity of the model.

- There is no mathematical implementation of the wind effect, the velocity of the UAV, and the possibility of malfunction of the equipment in the model.
- Every UAV is considered the same, and capacity time is the only parametric variable of UAVs.
- There are direct paths between every node, as given in Figure 12. Since the node map has the shape of a square, each road has a length. Represented as u_{ij}. These lengths are calculated by using analytical geometry. For example, considering Figure 12, the road length between node zero and node one is equal to one. The length of the road between node zero to node six is equal to two. Also, the road length between node zero to node four is equal to the square root of two. Lengths are not the exact time dependency of the roads given in the model. Cost of the Roads,

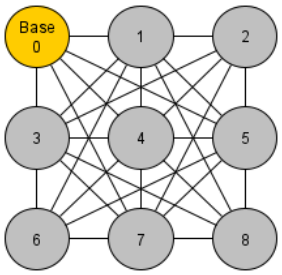

Figure 12. Node Map with Roads

  c_{ij}, which is a parameter and will be mentioned during the explanation of the mathematics of the model, is the actual time dependency of the roads.
- The flight distance over a node is ignored. It can be represented as the dn distance from Figure 13, which is considered zero. By ignoring dn, each node is considered a point in the road map rather than an area. This assumption aims to reduce additional computation from the model and keep assumption six as simple as possible.

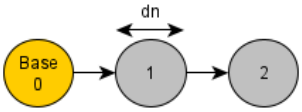

Figure 13. Flight Distance of a Node

- Each node needs a different scanning time to complete the operation in real life. To integrate different scanning time feature into the model, every node's necessary time is computed using a uniform distribution. Bounds of uniform distribution generated as parameters and chosen as one to three hours.
- Partial coverage of the nodes did not implement into the model to reduce complexity. Each UAV must travel a field if it has enough time left in its capacity to finish scanning that node and returning to the base. For example, a UAV travels from base node to node four; if it has enough capacity to reach node four, complete scan of node four and return to the base node considering the node map in Figure 12. Additionally, UAV can travel from node four to node one if it has enough time left after scanning node four. Then it can return to base from node one. Ignoring

partial coverage enables generating fly paths general to the time and before the flight, like passive mapping. Otherwise, the model had to be implemented so that time had to be steps rather than numeric computations. During each step, the model had to check the spend time on the nodes and decide on the paths during the flight, which is active mapping.

- Typically, all aerial vehicles with the plane's aerodynamics cannot have a perfect point in their routes. Each point in the route contains a curve because of the movement in time. However, this model ignores the effect of all physical limitations of the aerodynamics of planes to the routes.
- In real-life operations, time spent on refueling or managing must be added to the total operation time. However, since the model is not using time steps and calculates routes using passive mapping, which is explained in assumption nine, time spent on the base is ignored.
- To reduce the complexity of the model step-based time model was ignored. Because of that, computing the flight time of UAVs parallel is impossible. The main reason for that is step-based time model can compute the entire operation considering the flights of each UAV simultaneously and returns the optimal operation time as the time between the first takeoff and the last landing, without concerning what happened during the operation. However, the model calculated in this paper ignores the step-based time model, converts the model to the passive route mapping model. In the passive route mapping model, each route had to be calculated before the flight. To find the optimized operation time, the total time spends on the routes for every UAV minimized. This means that the output of the model is the minimized flight time combination of all flights. More clearly, the output is the time UAVs spend time in the air, not the entire operation (Figure 14). Since this model cannot give the parallelism, which real-time has, combined flight times optimized, possible improvement and advice will be given under the future work part.

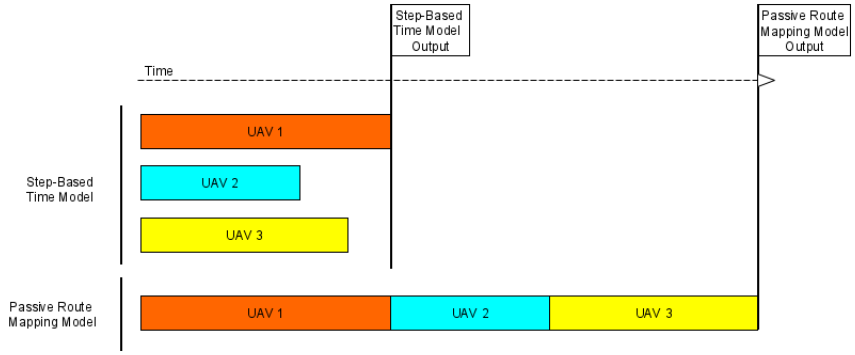

Figure 14. Operation Duration Time Representation Comparison for Step-Based Time Model and Passive Route Mapping Model

The binary decision variable x_{rij} is defined to indicate if the UAV $r, r \in \{1, 2, \ldots\ldots, p\}$ flies through the path (i, j) in an optimal solution. Predefined n value indicates the number of nodes in the model, including the base node. Cost of the Roads c_{ij}, which is mentioned in assumption six, is the required travel time to pass path (i, j). It is calculated by the multiplication of length of roads, u_{ij} and time

requirement of a unit road, w. The capacity of the UAVs, Q, is the available flight time of a single UAV. Time demand of the Nodes, d_i, indicates the required time to complete the scan on node i. The MIP model of the MT-CCVRP can be written as:

Minimize

$$\sum_{r=1}^{p}\sum_{i=0}^{n}\sum_{j=0\ i\neq j}^{n} c_{ij}x_{rij} \tag{1}$$

Subject to

$$\sum_{r=1}^{p}\sum_{i=0\ i\neq j}^{n} x_{rij} = 1, \qquad \forall j\epsilon\{1,\dots,n\}, \tag{2}$$

$$\sum_{j=1}^{n} x_{r0j} = 1, \qquad \forall r\epsilon\{1,\dots,p\}, \tag{3}$$

$$\sum_{i=0\ i\neq j}^{n} x_{rij} = \sum_{i=0}^{n} x_{rji}\,, \qquad \forall j\epsilon\{0,\dots,n\}, r\epsilon\{1,\dots,p\}, \tag{4}$$

$$\sum_{i=0}^{n}\sum_{j=1\ i\neq j}^{n} \left(d_j + c_{rij}\right)x_{rij} \leq Q, \qquad \forall r\epsilon\{1,\dots,p\}, \tag{5}$$

$$\sum_{r=1}^{p}\sum_{i=0}^{n}\sum_{j=0 i\neq j}^{n} x_{rij} \leq |S| - 1 + p \qquad \forall S \in \{1,\dots n\} \tag{6}$$

$$x_{rij} + x_{rji} \leq 2 \qquad \forall r\epsilon\{1,\dots,p\}, i\epsilon\{0,\dots,n\}, j\epsilon\{0,\dots,n\}, \tag{7}$$

$$x_{rij}\epsilon\{0,1\}, \qquad \forall r\epsilon\{1,\dots,p\}, i\epsilon\{0,\dots,n\}, j\epsilon\{0,\dots,n\}, i \neq j \tag{8}$$

$$\sum_{i=1}^{n} d_i \leq p(Q - 2\max(c)) \tag{9}$$

$$n \geq p \tag{10}$$

$$d_0 = 0 \tag{11}$$

The objective function (1) minimizes the total travel cost of the UAVs. The model constrain (2) is ensuring that each node is visited by exactly one UAV. The constraint (3) guarantees that each UAV can leave the depot. The constraint (4) guarantees that every flying UAV returns to the base. In the constraint (5), the capacity constraints are stated, ensuring that summation of the demands of nodes and demands of the roads is less than equal to UAVs capacity. Constraint (6) ensures that there is no discontinuity at the flight paths. It enforces the model to eliminate disconnected routes. In the constraint (7), the availability of returning path is satisfied. It enables UAVs to return to the base by using the direct route to reach the node. Also, it enables single node flights for UAVs. The constraint (8) is the

decision variables. It enables the flight of a specified UAV on a specific road. The constraint (9) ensures that all available UAVs' available capacity can satisfy the needed total demand of the nodes. The constraint (10) ensures there are more nodes than UAVs. Equation (11) means that there is no time demand for the base node.

To solve this MIP model, a solver software was required. However, the DSS has more than one parameter and requires calculations for each combination. The solver environment requires automation to solve processes since changing values and recalculating one by one is not a practical solution. So the Python programming language was decided to give flexibility and automation to the environment and PuLP libraries of Python language used to implement the MIP model. To achieve higher computational power, a CBC solver is used with a multithreaded configuration. To allow continuity of computations, the solver limited with ten minutes to solve each model. Each changeable parameter is given below in Table 1 with their values.

Table 1. Parameter Values

Parameters	Values Used
n	9, 16, 25, 36, 49, 64, 81, 100
p	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 25, 30, 35, 40, 45, 50
Q	180, 240, 300, 360, 420, 480, 540, 600
w	5, 10, 15, 20, 25, 30

Each parameter has eight, 18, eight, and six values, respectively. As a result, the python program calculated a total of 6912 MIP models and returned solution status and time. The program was run using Python 3.6 on a macOS with 2.9 GHz 6-Core Intel Core i9, 32GB 2400 MHz DDR4 RAM. 6912 MIP model solved in approximately 12 hours. However, the node demand requirements estimated as uniform distribution and were different for every generation, which can cause non-solved results for solvable designs. To overcome non-solved results, the python program executed a total of five-time and results are averaged from the five different results of the exact model.

3. Computational Results

This paper considers two examples as computational results to understand the outputs of the models and their usage for the DSS. For each example, values of the parameters and their resulted flight times are given in a table. Additionally, to compute the operation time, flight time is divided by the UAV value. Operation times will be the primary consideration factor of the search and rescue operations, and flight time will be the main factor for aerial scanning operations.

Example 1:

Table 2 contains the result for the number of UAVs from one to eight. Higher UAV numbers are not possible because of constraint number ten.

Table 2. Results of Example One

p	n	Q	w	Flight Time (min)	Operation Time (min)
1	9	420	5	- not coverable -	- not coverable -
2	9	420	5	- not coverable -	- not coverable -
3	9	420	5	64	21.33
4	9	420	5	77	19.25
5	9	420	5	90	18
6	9	420	5	106	17.66
7	9	420	5	126	18
8	9	420	5	146	18.25

Additionally, graphical solutions of $p = \{4,6,7\}$ are given below.

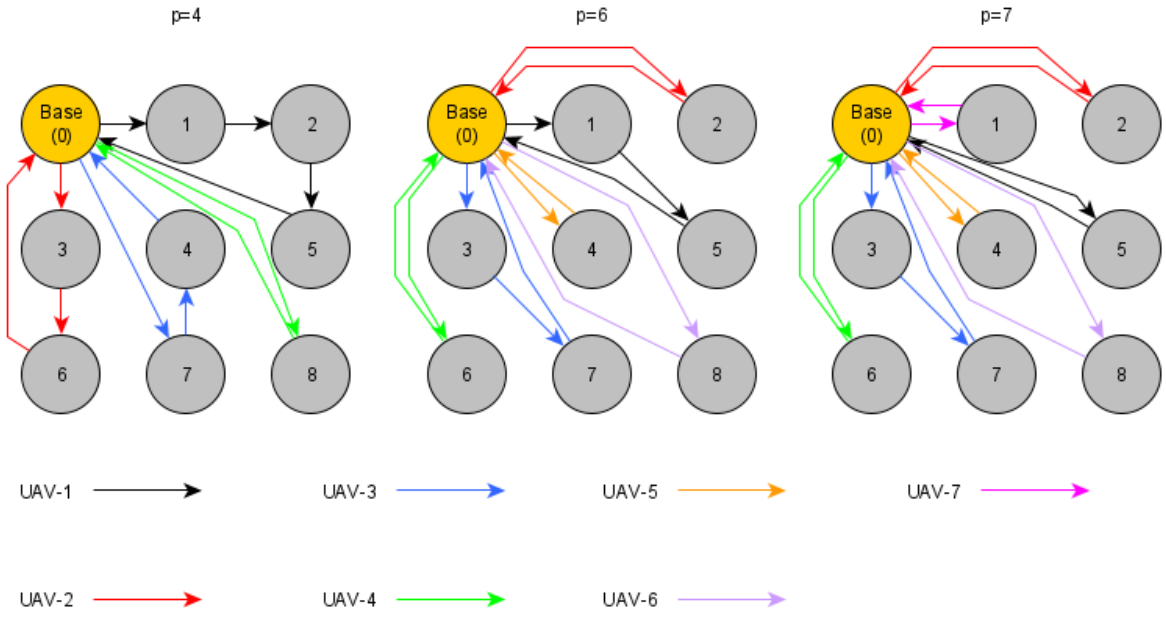

Figure 15. Graph of Result Example One

For search and rescue operations, using six UAVs for the given area in Table 2 is the optimal decision. Since minimum operation time is at that option, operators can decide on the number of operating UAV as six and complete the scan of the approximately 17.66 minutes.

However, for aerial scanning, the best option can depend on a time criterion. If the optimal case is to scan under a hundred minutes of flight time, then five UAV options can be the best one.

Additionally, operators can learn the minimum required UAV or the minimum number of sub-tours to scan the area.

Example 2:

Table 3 contains the result for the number of UAVs from one to 35. Higher UAV numbers are not possible because of constraint number ten.

Table 3. Results of Example Two

p	n	Q	w	Flight Time (min)	Operation Time (min)
1	36	540	5	- not coverable -	- not coverable -
2	36	540	5	- not coverable -	- not coverable -
3	36	540	5	- not coverable -	- not coverable -
4	36	540	5	- not coverable -	- not coverable -
5	36	540	5	- not coverable -	- not coverable -
6	36	540	5	- not coverable -	- not coverable -
7	36	540	5	- not coverable -	- not coverable -
8	36	540	5	- not coverable -	- not coverable -
9	36	540	5	306	34
10	36	540	5	332	33.20
15	36	540	5	482	32.13
20	36	540	5	669	33.45
25	36	540	5	886	35.44
30	36	540	5	1127	37.56
35	36	540	5	1414	40.40

For search and rescue operations, using 15 UAVs for the given area is the optimal decision. Since minimum operation time is at that option. Also, operators can get information about the effect of UAV numbers on the operation. As seen in Table 3, 10 UAVs search the operation area for approximately 33.2 minutes; however, 15 UAVs search the entire area for approximately 32.13 minutes. Operators most likely prefer to carry on with 10 UAVs rather than 15 because of insufficient change in the operation time.

For aerial scanning, if the optimal case is to decrease flight time as much as possible, then using nine UAVs is the best solution since fewer UAVs are insufficient to complete the scan. Additionally, operators can decide on increasing the fuel inside the UAVs by using DSS. Fuel addition can allow scanning areas with fewer UAVs in fewer flight times.

Table 4. Results of Example Two with Fuel Increase

p	n	Q	w	Flight Time (min)	Operation Time (min)
1	36	600	5	- not coverable -	- not coverable -
2	36	600	5	- not coverable -	- not coverable -
3	36	600	5	- not coverable -	- not coverable -
4	36	600	5	- not coverable -	- not coverable -
5	36	600	5	- not coverable -	- not coverable -
6	36	600	5	- not coverable -	- not coverable -
7	36	600	5	- not coverable -	- not coverable -
8	36	600	5	282	35.25
9	36	600	5	306	34
10	36	600	5	332	33.20
15	36	600	5	482	32.13
20	36	600	5	669	33.45
25	36	600	5	886	35.44
30	36	600	5	1127	37.56
35	36	600	5	1414	40.40

As can be seen in Table 4 area become scannable with eight UAVs after increasing the fuel. However, the flight and operation times of the other numbers of UAVs did not change, which indicates that UAVs are returning the base without depleting their fuel completely.

4. Usage of Decision Support System

This paper shares a DSS that uses four control parameters. Operators can limit those four control parameters with known values and observe results as combinations of unknown parameters. Also, operators can set specific limits to drop unnecessary results. For example, in Table 2 operator limited the node number, available flight time, and weight of the roads and observed the flight and operation time depending on the combinations of UAV numbers. Table 3 and Table 4 operator observed the flight and operation time depending on the combinations of UAV numbers and available flight time-limited to 540 and 600. In Table 5, operators can observe the flight and operation time depending on combinations of available flight time and node number limited to nine to 25. In addition, more control parameters can be added to MIP models, allowing operators to control more parameters in DSS. An application can be created for DSS to perform the operations given as examples and provide ease of use to the operators in the field. Furthermore, application do not have to share only the flight time results of the MIP models. Additionally, in step-based time models, applications can share the possible routes of the UAVs depending on the search area, etc.

Table 5. DSS Results of p=5 and n=[9,25]

p	n	Q	w	Flight Time (min)	Operation Time (min)
5	9	180	5	- not coverable -	- not coverable -
5	9	240	5	101.33	20.27
5	9	300	5	99.40	19.88
5	9	360	5	91.60	18.32
5	9	420	5	90	18
5	9	480	5	90	18
5	9	540	5	90	18
5	9	600	5	90	18
5	16	180	5	- not coverable -	- not coverable -
5	16	240	5	- not coverable -	- not coverable -
5	16	300	5	- not coverable -	- not coverable -
5	16	360	5	- not coverable -	- not coverable -
5	16	420	5	126	25.20
5	16	480	5	126	25.20
5	16	540	5	126	25.20
5	16	600	5	126	25.20
5	25	180	5	- not coverable -	- not coverable -
5	25	240	5	- not coverable -	- not coverable -
5	25	300	5	- not coverable -	- not coverable -
5	25	360	5	- not coverable -	- not coverable -
5	25	420	5	- not coverable -	- not coverable -
5	25	480	5	- not coverable -	- not coverable -
5	25	540	5	- not coverable -	- not coverable -
5	25	600	5	170	34

5. Future Work

Future work heading is included in this report to identify the possible improvement of the model expressed. There are two suggestions to the model that are relevant to the node time demand consumption and structure. Firstly, partitioning the time demand of a node between several UAVs can decrease the operation time significantly. The model expressed in this paper ignored partitioning the time possibility as specified in assumption number nine. Adding partitioning the time feature makes the model more realistic. Secondly, the structure of the model can be changed in two ways. The first way is to keep current models' idea by adding separate time computations for each UAV. Each UAV starts at

time zero, and the result can be taken as the operation time of a UAV, which has the longest flight time. The second way is to change models' structure. The solution might be more similar to a simulation than MIP. Every one-minute can be taken as a step, and UAVs can be distributed inside the model again at each step. For example, at the 40^{th} step, which also means 40^{th} minute, a UAV can check the status of the map and decide to join a search on a high time demanding node rather than returning base. Additionally, other UAVs can join the high time demanding node, so operation time can be reduced significantly, even if it increases the total flight time.

6. References

[1] Rivera, Juan Carlos, H. Murat Afsar, and Christian Prins. "A multistart iterated local search for the multitrip cumulative capacitated vehicle routing problem." *Computational Optimization and Applications* 61.1 (2015): 159-187.

[2] Borcinova, Zuzana. "Two models of the capacitated vehicle routing problem." *Croatian Operational Research Review* (2017): 463-469.

[3] Rivera, Juan Carlos, H. Murat Afsar, and Christian Prins. "Mathematical formulations and exact algorithm for the multitrip cumulative capacitated single-vehicle routing problem." *European Journal of Operational Research* 249.1 (2016): 93-104.

[4] Baker, Barrie M., and M. A. Ayechew. "A genetic algorithm for the vehicle routing problem." *Computers & Operations Research* 30.5 (2003): 787-800.

[5] Lysgaard, Jens, and Sanne Wøhlk. "A branch-and-cut-and-price algorithm for the cumulative capacitated vehicle routing problem." *European Journal of Operational Research* 236.3 (2014): 800-810.

[6] Ngueveu, Sandra Ulrich, Christian Prins, and Roberto Wolfler Calvo. "An effective memetic algorithm for the cumulative capacitated vehicle routing problem." *Computers & Operations Research* 37.11 (2010): 1877-1885.

[7] Azi, Nabila, Michel Gendreau, and Jean-Yves Potvin. "An exact algorithm for a single-vehicle routing problem with time windows and multiple routes." *European journal of operational research* 178.3 (2007): 755-766.

[8] Tirkolaee, Erfan Babaee, et al. "An improved ant colony optimization for the multi-trip Capacitated Arc Routing Problem." *Computers & Electrical Engineering* 77 (2019): 457-470.

7. Appendix

```
# Hurol Volkan Kasikaralar
# EM599 - Final Project - mt-CCVRP
#
# Final edit: 27/06/2021 - Refactor

import math
import random
import pandas as pd
import numpy as np
from pulp import *


def create_node_map(n):
    """
        Creates the node map matrix assuming map shape is square
    :param n: Number of total nodes
    :return: Node map matrix
    """
    # Assumption: map shape is square
    n_side = int(math.sqrt(n))
    nmap = np.zeros((n_side, n_side))
    x = 0
    for i in range(n_side):
        for j in range(n_side):
            nmap[i][j] = x
            x += 1
    return nmap


def create_arc_weights(node_map, n, w_unit):
    """
        Creates a matrix that contains the weight of the roads between each node
    :param node_map: Node map matrix
    :param n: Number of total nodes
```

```
    :param w_unit: Weight of a unit road
    :return: Road weight matrix
    """
    wmap = np.zeros((n, n))
    for i in range(n):
        node_f = np.where(node_map == i)
        for j in range(n):
            node_s = np.where(node_map == j)
            if i == j:
                wmap[i][j] = 0
            else:
                wmap[i][j] = round(math.sqrt((node_f[0] - node_s[0]) ** 2 + (node_f[1] - node_s[1]) ** 2)
* w_unit)
    return wmap


def create_node_demand(weight_map, n, R, Q, demand_limit):
    """
        Creates the time demand of each node by using the uniform distribution for given limits and
check whether
         drones can cover the grid or not.
    :param weight_map: Road weight matrix
    :param n: Number of total nodes
    :param R: Number of available drones
    :param Q: Capacity of a single drone
    :param demand_limit: Limits of uniform distribution
    :return: Cover success (Boolean), Time demand of nodes as matrix, Time demand of nodes as
array
    """
    n_side = int(math.sqrt(n))
    dmap = np.zeros((n_side, n_side))

    while True:
        for i in range(n_side):
            for j in range(n_side):
                if i == 0 and j == 0:
```

```
                dmap[i][j] = 0
            else:
                dmap[i][j] = int(random.uniform(demand_limit[0], demand_limit[1]))
        if sum(sum(dmap)) <= R * np.floor(Q - 2 * np.max(weight_map)):  # Checks the coverage
            return True, dmap, dmap.flatten()
        else:
            return False, 0, 0


def create_multi_weight_map(weight_map, R):
    """
        Creates multiple weight matrices to decrease complexity of the calculations
    :param weight_map: Time demand of nodes as array
    :param R: Number of available drones
    :return: Time demand of nodes as multiple arrays
    """
    multi_weight_map = weight_map
    for i in range(R - 1):
        if i == 0:
            multi_weight_map = np.vstack((multi_weight_map[None], weight_map[None]))
        else:
            multi_weight_map = np.vstack((multi_weight_map, weight_map[None]))
    return multi_weight_map


def create_dataset(n, R, Q, w_unit, demand_limit):
    """
        Creates the MIP and solves it
    :param n: Number of total nodes
    :param R: Number of available drones
    :param Q: Capacity of a single drone
    :param w_unit: Weight of a unit road
    :param demand_limit: Limits of uniform distribution
    :return: Result of the MIP, Status of MIP solution
    """
```

```
    node_map = create_node_map(n)
    weight_map = create_arc_weights(node_map, n, w_unit)
    status, demand_map, demand = create_node_demand(weight_map, n, R, Q, demand_limit)
    if not status:
        return -1, "Not Cover"
    multi_weight_map = create_multi_weight_map(weight_map, R)

    node_list = list(range(0, n))
    node_list_demand = list(range(1, n))
    car_list = list(range(1, R + 1))

    costs = makeDict([car_list, node_list, node_list], multi_weight_map, 0)

    # Initializing the Solver
    model = LpProblem("mt-CCVRP", LpMinimize)

    x = LpVariable.dicts("Route", (car_list, node_list, node_list), 0, 1, cat="Binary")  # variable "x"

    routes = list(
        filter(None, [((r, i, j) if i != j else None) for r in car_list for i in node_list for j in node_list]))
    model += lpSum(
        [x[r][i][j] * costs[r][i][j] for (r, i, j) in routes]), "Objective Function"  # Objective Function (1)

    for j in node_list_demand:
        parameter = list(filter(None, [((r, i, j) if i != j else None) for r in car_list for i in node_list]))
        model += lpSum(
            [x[r][i][j] for (r, i, j) in parameter]) == 1, "(2) Node %s Visit Constraint" % j  # Constraint (2)

    for r in car_list:
        model += lpSum(
            [x[r][0][j] for j in node_list_demand]) == 1, "(3) Vehicle %s Base Leave Constraint" % r  #
Constraint (3)

    for r in car_list:
        for j in node_list:
            parameter = list(filter(None, [((r, i, j) if i != j else None) for i in node_list]))
```

```
            parameter_2 = list(filter(None, [((r, j, i) if i != j else None) for i in node_list]))
            model += lpSum([x[r][i][j] for (r, i, j) in parameter]) == lpSum(
                [x[r][j][i] for (r, j, i) in parameter_2]), "(4) Vehicle %s Base Return From %s Constraint"
% (
                    r, j)  # Constraint (4)

    for r in car_list:
        parameter = list(filter(None, [((r, i, j) if i != j else None) for i in node_list for j in
node_list_demand]))
        model += lpSum([x[r][i][j] * (demand[j] + costs[r][i][j]) for (r, i, j) in
                    parameter]) <= Q, "(5) Capacity of Vehicle %s Constraint" % r  # Constraint (5)

    parameter = list(filter(None, [((r, i, j) if i != j else None) for r in car_list for i in node_list_demand
for j in
                            node_list_demand]))

    model += lpSum([x[r][i][j] for (r, i, j) in
                parameter]) <= n - 1 + R, "(6) No Disconnect at the Routes Constraint"  # Constraint (6)

    for r in car_list:
        for i in node_list:
            for j in node_list:
                model += lpSum(
                    [x[r][i][j] + x[r][j][i]]) <= 2, "(7) For Vehicle %s Not Use Same Road %s %s Constraint"
% (
                        r, i, j)  # Constraint (7)

    # MIP Solvers
    status = model.solve(pulp.PULP_CBC_CMD(msg=False, timeLimit=600, gapRel=0, threads=8))
    # status = model.solve(CPLEX_CMD(timeLimit=600, gapRel=0))
    # for var in model.variables():
    #     print(f"{var.name}: {var.value()}")
    # for name, constraint in model.constraints.items():
    #     print(f"{name}: {constraint.value()}")

    if model.status == 1:
```

```
        return model.objective.value(), LpStatus[model.status]
    else:
        return -1, LpStatus[model.status]


# Decision Parameters
n_list = [9, 16, 25, 36, 49, 64, 81, 100]  # number of node
R_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 25, 30, 35, 40, 45, 50]  # number of drones
Q_list = [180, 240, 300, 360, 420, 480, 540, 600]  # capacity of the drones
w_unit_list = [5, 10, 15, 20, 25, 30]  # unit weight of the arc
demand_limits = [60, 180]

count_size = len(n_list) * len(R_list) * len(Q_list) * len(w_unit_list)
turn = 0

# Multiple MIP Calculator
for node in n_list:
    df = pd.DataFrame(columns=['node_number', 'drone_number', 'capacity_of_drones', 'arc_weight',
'result', 'type'])
    for drone in R_list:
        for capacity in Q_list:
            for weight_unit in w_unit_list:
                value, output_type = create_dataset(node, drone, capacity, weight_unit, demand_limits)
                row = {'node_number': node, 'drone_number': drone, 'capacity_of_drones': capacity,
                       'arc_weight': weight_unit, 'result': value, 'type': output_type}
                df = df.append(row, ignore_index=True)
                turn += 1
                print(
                    'Iteration {}% \nNode     {} \nDrone     {} \nCapacity  {} \nWeight    {} \nValue     {}
\nType     {}'.format(
                        round(turn * 100 / count_size, 2), node, drone, capacity, weight_unit, value,
output_type))
                print('-------------------------------------------------')
    df.to_csv('Output/output_{}.csv'.format(node))
```