Energy-Efficient Drone Coverage Path Planning using Genetic Algorithm

Rutuja Shivgan and Ziqian Dong
Department of Electrical and Computer Engineering
College of Engineering and Computing Sciences
New York Institute of Technology, New York, NY 10023
Email: {rshivgan, ziqian.dong}@nyit.edu

***Abstract*—Unmanned Aerial Vehicles (UAVs) have been increasingly used in environmental sensing and surveying applications. Coverage path planning to survey an area while following a set of waypoints is required to complete a task. Due to the battery capacity, the UAV flight time is often limited. In this paper, we formulate the UAV path planning problem as a traveling salesman problem in order to optimize UAV energy. We propose a genetic algorithm to solve the optimization problem i.e. to minimize the energy consumption for the UAV to complete a task. We also consider reducing the number of turns to allow the UAV to optimize the flight path and to minimize its energy consumption. We compare the energy consumption of the proposed genetic algorithm to the greedy algorithm with different number of waypoints. Results show that our proposed algorithm consumes 2-5 times less energy than that of the greedy algorithm by reducing the number of turns while covering all the waypoints.**

***Index Terms*—Unmanned Aerial Vehicles, Energy model, Path planning, traveling salesman problem, Optimization, Genetic algorithm**

I. Introduction

Unmanned Aerial Vehicles (UAVs) are widely used for surveying, inspection of power plants, search and rescue (SAR) operations, precision agriculture and various other applications [1]. These applications require specific sensors, such as video cameras, temperature sensors, proximity sensors, ground sensors and etc. to collect information. Wireless sensor networks (WSNs) have been adopted for real-time data collection that can be executed through a swarm of UAVs [2].

Coverage path planning is the determination of the path that a UAV must follow in order to visit every set point in a given area. Many researches have been conducted on UAV path planning while avoiding obstacles such as those inspecting buildings and transmission lines. As UAVs have limited battery power, their flight time is often limited by the battery capacity. The objective of this paper is to minimize the energy consumption of a UAV to cover a surveying area. This problem can be formulated as a Traveling Salesman Problem (TSP), which is an NP-Hard problem. While most of the research has focused on finding the shortest distance coverage, they often do not consider the energy consumption of a drone which depends on the speed, distance, acceleration, deceleration and the number of turns it takes. In this paper, we provide a more accurate model that considers energy consumption during acceleration, deceleration and turning to find the optimized energy-efficient path. We use UAV and drone interchangeably in the remainder of this paper.

978-1-7281-4846-5/20/$31.00 ©2020 IEEE

Many researches have focused on finding the optimized drone path planning to carry out the tasks. Nascimento [3] studied the best drone tour for data collection in WSN which considered drone flight time and the time required for data collection from WSN nodes. The author proposed brute force algorithms with two heuristics which gave a near optimal solution in small WSN with an improved performance in large WSN area when compared to other local search approaches. Galceron and Carrerras [4] introduced various methods such as cellular, grid-based, and graph-based approaches for drone coverage path planning. These methods segment the images of a given coverage area to find sub-optimal coverage path for a drone to follow. Carmelo Di Franco [5] proposed energy-aware coverage path planning of UAVs. The authors carried out a set of experiments at different operating conditions like acceleration, constant speed and deceleration to derive an energy model for a single UAV. Maza et al. [6] presented an algorithm to ensure the full coverage of a desired area. It divides a given area into grids and is designed to have UAVs to fly through the center of each grid. Each UAV has to follow a zig-zag pattern using sweep direction that helped in minimizing the number of turns, however, the algorithm does not consider obstacle avoidance.

Gupta [7] used genetic algorithms to solve the TSP. This approach can be used for various applications like vehicle routing problems, task scheduling, etc. Kiraly [8] presented a multi-chromosome genetic algorithm to find the optimized route plan with minimal route cost for each salesmen. The author considered constraints like maximum tour length per salesmen and the time window of truck loading time to solve multiple traveling salesman problem. Feo [9] gave a brief literature review of Greedy Randomized Adaptive Search Procedure (GRASP) which consists of a construction phase in which a solution was found via adaptive greedy function and a local search phase for a more efficient solution. However, the rendered results were local optimal.

For multiple drone networks, Modares et al. [1] proposed an energy-efficient coverage path planning approach, where a

Authorized licensed use limited to: Ericsson. Downloaded on September 24,2024 at 20:02:51 UTC from IEEE Xplore. Restrictions apply.

task is allocated to each drone based on their battery capacity to minimize energy consumption for each drone. Bellingham [10] studied the co-ordination of a swarm of drones in which each UAV has to perform tasks such as visiting a set of waypoints while avoiding no fly zones and collision avoidance between UAVs. Li [11] proposed path planning of multiple UAVs to inspect an area where topography of plateaus and mountains fluctuates. The algorithm was divided into two phases. The first phase finds the global optimum route with the minimum number of turns using a single drone. The second phase divides the search area according to UAVs initial position and assigns tasks to the set of UAVs. The considered constraints include drone angle restriction, terrain restrictions, flight altitude limits, and battery life. Ant colony optimization (ACO) algorithm has also been explored to solve the multiple traveling salesman problem in [12], where a Max-min ant system was developed to minimize the path length for each salesman.

The contribution of this paper includes: we formulated the drone path planning problem as a traveling salesman problem with an objective to optimize the drone energy consumption while completing a task of traversing all required waypoints and return to its initial position. We considered minimizing energy consumption by reducing the number of turns which will be a key factor of this project. We proposed a genetic algorithm to solve the optimization problem, implemented the solvers in MATLAB and compared the results with greedy algorithm. We showed the proposed algorithm can significantly reduce the energy consumption than that of the greedy algorithm, especially for an area with a large number of waypoints.

The remainder of the paper is organized as follows. Section II introduces the energy consumption model of a UAV. Section III presents the optimization problem formulation of UAV energy consumption with path planning. Section IV presents the proposed genetic algorithm and its implementation. Section V compares the performance of the proposed genetic algorithm with greedy algorithm. Section VI concludes the paper.

II. Energy Consumption Model

In this section, we present the drone energy-consumption model for a UAV that needs to traverse a set of waypoints in a surveyed area to perform data collection tasks. The UAV is a quadrotor with quad-core 64-bit, 2.56 GHz processor and 3300 mAH Li-Po battery. We used the energy model proposed by Lige Ding [13] to formulate the energy consumption problem. As the energy consumption of a drone depends on the drone speed, the energy consumption model needs to consider the different flight stages including acceleration, deceleration, hovering, and turning. The consumed power is calculated by multiplying the supply voltage and the current, which is measured by an energy measurement module. The experiments conducted by Lige Ding provided a better understanding of the energy performance of a drone [13].

A. Effect of velocity

The UAV was commanded to fly in a straight line at different constant speeds. The power consumption measured at 2 m/s, 4 m/s, 6 m/s and 8 m/s were 242W, 245W, 246W and 268W, respectively [13].

B. Effect of acceleration and deceleration

Power consumption during acceleration and deceleration was recorded and can be used to calculate energy consumption when UAV accelerates and decelerates. Similar readings of power consumption were observed during the acceleration and deceleration phases as observed in effect of velocity. As velocity increases, power consumption during acceleration increases and vice versa.

C. Energy consumption model for turning phase

In this experiment, the power consumption of the UAV was recorded when it rotated at four different angles 45°, 90°, 135°, 180°. It was assumed that an angular speed of ω_{turn} (2.07 rad/sec) would require P_{turn} (260 W) for the turn [13]. The energy consumption at turning angle $\Delta\theta$ can be calculated by

$$E_{turn} = P_{turn}\frac{\Delta\theta}{\omega_{turn}} \tag{1}$$

where E_{turn} denotes Energy consumption during turn, P_{turn} denotes Power consumption during turn, $\Delta\theta$ denotes turning angle, ω_{turn} denotes angular velocity during turn.

D. Energy consumption model for flying straight

When a UAV flies from a starting point to a target point, it goes through three phases: acceleration phase, uniform speed phase and deceleration phase. We can calculate energy consumption at different speeds and distance $E(v, d)$.

$$E(v,d) = \int_0^{t_1} P_{acc}dt + \int_{t_1}^{t_2} P(v)dt + \int_{t_2}^{t_3} P_{dec}dt \tag{2}$$

Where $Pacc$ denotes power consumption during acceleration, $Pdec$ denotes Power consumption during deceleration, $P(v)$ denotes Power consumption at uniform velocity v, v denotes velocity, d denotes distance traveled, and t_1, t_2 and t_3 denote the time duration of the acceleration phase, constant speed flight phase, and deceleration phase, respectively.

When a UAV travels for a short distance, it is not able to achieve a uniform speed because it directly goes from accelerating phase to decelerating phase. Eq. 2 can be used to evaluate the optimal drone speed. When the distance is too short, drone will take its whole time on accelerating and decelerating and when the distance between two way points is large enough, drones can go through all flight phases and reach an optimal speed. Once the path length is ideal, the drone can remain at the optimal speed.

Authorized licensed use limited to: Ericsson. Downloaded on September 24,2024 at 20:02:51 UTC from IEEE Xplore. Restrictions apply.

III. Energy-Efficient Path Planning Problem Formulation

We define the energy-efficient path planning problem in this section. Let V denote the set of waypoints a UAV needs to visit. The UAV is required to return to their initial waypoint after completion of a task. We consider a graph $G(V, E)$, where V denotes the set of n waypoints and E denotes the set of edges. Let e_{ij} be the energy consumption between waypoint i and j which can be calculated from Eq. 2. Binary decision variable $x_{ij} \epsilon \{0, 1\}$ is used to denote that path between waypoint i and j is selected or not.

$$x_{i,j} = \begin{cases} 1, & \text{if drone travels from } i \text{ to } j, \\ 0, & \text{the edge is not selected.} \end{cases} \tag{3}$$

Consider (α_i, β_i) are the coordinates of waypoint i and (α_j, β_j) are the coordinates of waypoint j. Let d_{ij} be the distance between waypoint i and j. The distance between two waypoints is calculated as follows:

$$d_{ij} = \sqrt{(\alpha_i - \alpha_j)^2 + (\beta_i - \beta_j)^2} \tag{4}$$

We formulate the energy-efficient path planning problem as follows:

$$\min \quad \sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} e_{ij} x_{ij} \tag{5}$$

$$s.t \sum_{i=1, i \neq j}^{n} x_{1j} = \sum_{j=2}^{n} x_{j1} = 1 \tag{6}$$

$$\sum_{i=1, i \neq j}^{n} x_{ij} = 1 \quad \forall\ j \epsilon\ V \tag{7}$$

$$\mu_i - \mu_j + p x_{ij} \leq p - 1 \quad \forall\ 2 = i \neq j \leq n \tag{8}$$

Eq. 6 represents the constraint that the drone will depart from an initial waypoint and return to the initial waypoint. Eq. 7 indicates the constraint that each waypoint is preceded by and precedes exactly one another waypoint except the initial waypoint. Eq. 8 ensures that subtours are eliminated. It is called the Miller-Tucker-Zemlin (MTZ) subtour elimination constraint. μ_i is the order in which waypoint i is visited and μ_j is the order in which waypoint j is visited. p denotes the maximum number of waypoints that can be visited by any drone.

Our problem formulation is a Traveling Salesman Problem, which is similar to Vehicle Routing Problem and is considered NP-Hard. Our goal is to minimize the energy consumption while reducing the number of turns. To solve our optimization problem, we used genetic algorithm which is introduced in the next section.

IV. Genetic Algorithm

Genetic algorithm (GA) is a heuristic algorithm which is based on genetic and random selection. It was introduced by John Holland in 1960 to find the optimum solution for NP-hard problem depending on bio-inspired operators like mutation and crossover [7].

The following terms are used in describing the genetic algorithm:

A. Initial Population

Population is the number of possible solutions for a given problem. The term population is similar to human being population, in our case we consider all possible traveling routes for the drone as population. Chromosome is another term which represents single solution for our problem. Gene is the one element position for a chromosome. The size of the population should be large enough to optimize our solution, which was found after multiple trials. The initial population is created by a random function in MATLAB.

B. Fitness value

Fitness value is defined as the optimized solution for our problem. It takes the value of the better fitted chromosome among all the chromosomes compared at each iteration. For each iteration, new chromosome is created and the fitness value for this chromosome, f, is calculated based on the energy consumed along this path, where

$$f = min \sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} e_{ij} x_{ij}$$

C. Selection

Different types of selection mechanisms can be adopted in the selection of a parent. For Roulette wheel selection, the probability of selecting an individual parent depends on the fitness value. In tournament selection, k-random individual solutions from population are considered where the best solution out of these k solutions is selected. This will be considered as parent 1 and the same procedure is followed to select other parent [7]. In this paper, we used tournament selection with k = 5. From 5 random solutions, we chose the best solution to be the parent. Since our objective is to minimize energy consumption, selection of chromosomes depends on whether it presents a smaller fitness value, which means a more energy-efficient path.

D. Crossover

Crossover is the term that represents the production of the next generation which may have better fitness solution. We used two-point crossover in which two points are randomly selected and applied to a pair of chromosomes where genes are exchanged with each other to generate a new chromosome.

Authorized licensed use limited to: Ericsson. Downloaded on September 24,2024 at 20:02:51 UTC from IEEE Xplore. Restrictions apply.

E. Mutation

Mutation is the term used to represent the new possible solution derived from crossover solution by flipping bit or swapping the order of genes. In flipping mutation, the order of 2 or more random genes will be reversed. In this paper, we used swapping mutation where two or more random genes are selected from a chromosome and then swapped [7].

F. Termination and result

Once all the iterations are completed, the process will be terminated and the solution for our problem is generated.

Table I shows the list of variables used in the algorithms and their descriptions. Here, α, β denotes x and y co-ordinates of the waypoint. n denotes the number of waypoints. pop denotes the population which is the set of all possible paths a drone can follow. $popSize$ denotes the population size which is the number of all possible paths. $emat$ is the energy matrix which is calculated based on the energy model in Section II. It includes the values of energy consumption by drone while travelling in straight line, accelerating, decelerating and during turn. $dmat$ is the distance matrix that records distances from one waypoint to all other waypoints. $minEnergy$ is the minimum energy consumed by the drone while traveling all the waypoints. $globalMin$ is the global minimum value which is used to store the minimum energy after each iteration. $tmpPop$ is used for storing temporary solution. $newPop$ represents variable used for saving new generated solution after crossover and mutation.

Energy matrix, x and y co-ordinates of waypoints and population size are inputs for GA. First we calculate distance matrix given the x and y co-ordinates of all waypoints. The results are stored in $dmat$. Then the total energy consumption and total distance traveled are calculated for each iteration. In this step, all possible solutions, e.g. the distance and energy from waypoint 1 to n is calculated. The minimum total energy is stored in $minEnergy$. If the calculated $minEnergy$ is less than $globalMin$, $globalMin$ will be updated with this value and the process will be terminated. If it is not the optimum result, it will use crossover and mutation methods. The selection of parents is based on tournaments selection method where the best solution is selected from $k = 5$ random solutions. A pair of solutions obtained from the tournament selection method is selected and the two points will be randomly selected for crossover. The new solution is generated after exchanging the genes and stored in $tmpPop$. To apply mutation, we swap the order of genes in the solution. The newly generated solution will be stored in $newPop$. After this, pop is updated with new solutions. Again the energy consumption during those solutions (paths) is calculated. If it is less than $globalMin$, this solution will be the the global minimum and the process will be terminated. The pseudo code for genetic algorithm is shown in Algorithm 1.

Algorithm 1 Genetic Algorithm

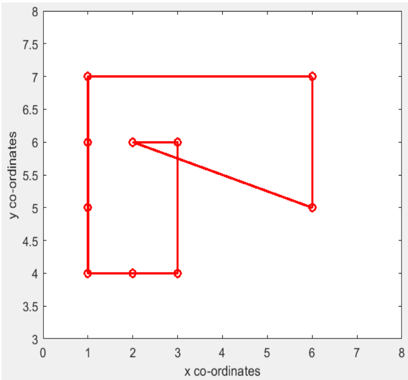
```
Input: Energy matrix, x and y co-ordinates, Population size
for i = 1 : n do
    for j = 1 : n do
        d_ij = sqrt((α_i − α_j)^2 + (β_i − β_j)^2)
    end for
end for
Initialize Population
for iteration = 1 : numIter do
    for p = 1 : popSize do
        d = dmat(pop(p, n), pop(p, 1));
        e = emat(pop(p, n), pop(p, 1));
        for k = 2 : n do
            d = d + dmat(pop(p, k − 1), pop(p, k));
            e = e + emat(pop(p, k − 1), pop(p, k));
        end for
        totalDist(p) = d;
        totalEnergy(p) = e;
    end for
    if minEnergy < globalMin then
        result = minEnergy;
    else
        calculate total energy again using crossover and mutation
    end if
    for p = 5 : 5 : popsize do
        Use Tournament selection to find parents
        Apply crossover to find new solution
        save it in tmpPop
        for k = 1 : size(tmpPop) do
            select previously generated solution from tmpPop
            apply swapping mutation
            if solution is feasible then
                save it in variable newPop
            end if
        end for
        update the solution in pop
    end for
end for
```

Greedy algorithm is a heuristic algorithm which finds local optimum solution at every iteration. It does not always reach a global optimal solution but gives locally optimal solution in less time. For greedy algorithm, it follows the same process to calculate the minimum energy consumption as genetic algorithm but the main difference is that greedy algorithm does not use crossover and mutation process so it does not always reach the optimum results. The pseudo code for greedy algorithm is shown in Algorithm 2.

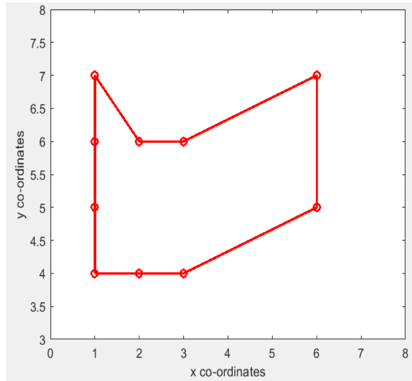
V. Results and Analysis

We compared the genetic algorithm and the greedy algorithm in solving a network of N waypoints a drone has to traverse, where $N = \{10, 25, 50, 100\}$. We tested the network in MATLAB under the following three scenarios: 1) using genetic algorithm by optimizing the energy consumption, 2) using genetic algorithm by optimizing only the distance traveled, and 3) greedy algorithm by optimizing energy consumption.

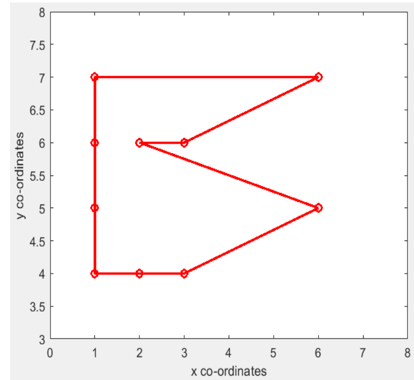
Authorized licensed use limited to: Ericsson. Downloaded on September 24,2024 at 20:02:51 UTC from IEEE Xplore. Restrictions apply.



(a) Genetic algorithm with energy optimization.



(b) Genetic algorithm with distance optimization.



(c) Greedy algorithm with energy optimization.

Fig. 1: Path trajectory of UAV with 10 waypoints using a) Genetic algorithm with energy optimization, b) Genetic algorithm with distance optimization, and c) Greedy algorithm with energy optimization.

TABLE I: Terminology

Variable	Description
α, β	x and y co-ordinates
n	Number of waypoints
pop	Population
$popSize$	Population Size
$emat$	Energy Matrix
$dmat$	Distance Matrix
$minEnergy$	Minimum Energy
$globalmin$	Global minimum
$tmpPop$	Temporary population
$newPop$	New population

Algorithm 2 Greedy Algorithm

Input: Energy matrix, x and y co-ordinates, Population size
for $i = 1 : n$ **do**
  for $j = 1 : n$ **do**
    $d_{ij} = \sqrt{(\alpha_i - \alpha_j)^2 + (\beta_i - \beta_j)^2}$
  end for
end for
Initialize population
for $iteration = 1 : numIter$ **do**
  for $p = 1 : popSize$ **do**
    $d = dmat(pop(p, n), pop(p, 1));$
    $e = emat(pop(p, n), pop(p, 1));$
    for $k = 2 : n$ **do**
      $d = d + dmat(pop(p, k-1), pop(p, k));$
      $e = e + emat(pop(p, k-1), pop(p, k));$
    end for
    totalDist(p) = d;
    totalEnergy(p) = e;
  end for
  if $minEnergy < globalMin$ **then**
    $result = minEnergy$
  end if
end for

TABLE II: Average energy consumption with genetic algorithms and greedy algorithm for 10 waypoints.

Algorithm	GA (Energy optimization)	GA (Distance optimization)	Greedy (Energy optimization)
Energy consumption (kJ)	1.8766	2.2019	2.076
Distance traveled (m)	19.1231	15.7388	20.3983
Computation time (sec)	13	14	4

Fig. 1 shows the path selected by each algorithm for 10 waypoints. As the number of turns depends on the turning angle, the proposed genetic algorithm aims at reducing the number of turns and thus minimizes its energy consumption. Fig. 1a shows the path selection for genetic algorithm with energy consumption optimization. We can see that the distance between two waypoints in a chosen path may be larger, however, the energy consumption of choosing this path is less. So the drone will choose that path instead. Fig.1b shows the path selection by the UAV when using genetic algorithm by optimizing the distance traveled. Table II presents the average energy consumption, distance traveled and computation time for the genetic algorithms and greedy algorithm. The energy consumption by drone using genetic algorithm with energy optimization and with distance optimization is 1.8766 kJ and 2.2019 kJ, respectively. We observed that the calculated energy consumption by genetic algorithm with distance optimization is 0.33 kJ higher than the genetic algorithm with energy optimization. For greedy algorithm, the average energy consumption of the drone is 2.076 kJ. Fig. 1c shows the path selection by the UAV using greedy algorithm. Greedy algorithm fails to find the optimized path for energy optimization as calculated energy consumption by greedy algorithm is 0.2 kJ higher than the genetic algorithm with energy optimization.

We also compared the computation time for the genetic algorithm with energy optimization, genetic algorithm with distance optimization, and greedy algorithm. The genetic algorithm with energy optimization takes about 13 secs to

Authorized licensed use limited to: Ericsson. Downloaded on September 24,2024 at 20:02:51 UTC from IEEE Xplore. Restrictions apply.

TABLE III: Mean energy consumption of genetic algorithms, greedy algorithm and their SD for different waypoints.

No. of waypoints	GA (Energy optimization)		GA (Distance optimization)		Greedy algorithm (Energy optimization)	
	Mean	**± Std. deviation**	**Mean**	**± Std. deviation**	**Mean**	**± Std. deviation**
10	1.8766 kJ	0	2.2019 kJ	0	2.076 kJ	0.1994
25	26.3036 kJ	2.6533	28.86 kJ	8.9716	57.7268 kJ	16.4832
50	58.7393 kJ	7.9109	67.3361 kJ	16.5077	160.2181 kJ	80.5338
100	826.7902 kJ	44.8034	1021.307 kJ	188.712	4362.8 kJ	1353.15

complete, the genetic algorithm with distance optimization takes 14 secs, while greedy algorithm with energy optimization takes only 4 secs for path finding. We further compared these three algorithms by increasing the number of waypoints. For $N = \{25, 50, 100\}$, we generated a random symmetric energy matrix of 25×25, 50×50 and 100×100, respectively. We evaluated the total energy consumed by the UAV to complete traversing all the waypoints. We carried out 100 experiments and compared the average energy consumption for path planning of drone for each algorithm.

Table III shows the average energy consumption by the proposed genetic algorithm with energy optimization, the genetic algorithm that optimizes distance traveled, the greedy algorithm and their standard deviations (SD) when $N = \{10, 25, 50, 100\}$. For the proposed genetic algorithm with energy optimization, energy consumption for $N = \{10, 25, 50, 100\}$ are 1.8766 ± 0 kJ, 26.3036 ± 2.6533, 58.7393 ± 7.9109 kJ and 826.7902 ± 44.8034 kJ, respectively. Similarly, for $N = \{10, 25, 50, 100\}$, the energy consumption for GA with travelled distance optimization are 2.2019 ± 0 kJ, 28.86 ± 8.9716, 67.3361 ± 16.5077 kJ and 1021.307 ± 188.712 kJ, respectively. For greedy algorithm, the energy consumption are 2.076 ± 0.1994 kJ, 57.7268 ± 16.4832, 160.2181 ± 80.5338 kJ and 4362.8 ± 1353.15 kJ for $N = \{10, 25, 50, 100\}$, respectively. It is clearly shown that the proposed genetic algorithm with energy optimization has the smallest standard deviation and converges to minimum energy consumption compared to genetic algorithm with distance optimization and greedy algorithm. The proposed genetic algorithm with energy optimization is shown to be able to significantly reduce the drone energy consumption about 2-5 times as compared to greedy algorithm as the number of waypoints increases. The benefit of the proposed algorithm is more prominent in larger network coverage path planning than greedy algorithm.

VI. Conclusions

In this paper, we presented a model of drone path planning considering the total energy consumption of completing a task. The energy model is based on drone acceleration, deceleration, hovering and turning. We formulated the energy-efficient path planning problem as a traveling salesman problem. Genetic algorithm was proposed to minimize the energy consumption of a drone by reducing the number of turns. Our results show that genetic algorithm with energy optimization uses on average 2-5 times less energy comparing to that of the greedy algorithm and the energy saving is more prominent as the number of waypoints increases. As future work, we plan to improve the the genetic algorithm to reduce its computation complexity and extend the study to multiple-drone path planning problem using different optimization algorithms.

Acknowledgement

This material is based upon work supported by the National Science Foundation under Grant No. 1841558.

References

[1] J. Modares, F. Ghanei, N. Mastronarde, and K. Dantu, "Ub-anc planner: Energy efficient coverage path planning with multiple drones," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 6182–6189.

[2] M. Erdelj, E. Natalizio, K. R. Chowdhury, and I. F. Akyildiz, "Help from the sky: Leveraging uavs for disaster management," *IEEE Pervasive Computing*, vol. 16, no. 1, pp. 24–32, 2017.

[3] R. I. da Silva and M. A. Nascimento, "On best drone tour plans for data collection in wireless sensor network," in *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, ser. SAC '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 703–708. [Online]. Available: https://doi.org/10.1145/2851613.2851854

[4] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robotics and Autonomous systems*, vol. 61, no. 12, pp. 1258–1276, 2013.

[5] C. Di Franco and G. Buttazzo, "Energy-aware coverage path planning of uavs," in *2015 IEEE International Conference on Autonomous Robot Systems and Competitions*. IEEE, 2015, pp. 111–117.

[6] I. Maza and A. Ollero, "Multiple uav cooperative searching operation using polygon area decomposition and efficient coverage algorithms," in *Distributed Autonomous Robotic Systems 6*. Springer, 2007, pp. 221–230.

[7] S. Gupta and P. Panwar, "Solving travelling salesman problem using genetic algorithm," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 3, pp. 376–380, 01 2013.

[8] A. Király and J. Abonyi, "Redesign of the supply of mobile mechanics based on a novel genetic optimization algorithm using google maps api," *Engineering Applications of Artificial Intelligence*, vol. 38, 11 2014.

[9] T. Feo and M. Resende, "Greedy randomized adaptive search procedures," *Journal of Global Optimization*, vol. 6, pp. 109–133, 03 1995.

[10] J. Bellingham, M. Tillerson, A. Richards, and J. P. How, *Multi-Task Allocation and Path Planning for Cooperating UAVs*. Boston, MA: Springer US, 2003, pp. 23–41. [Online]. Available: https://doi.org/10.1007/978-1-4757-3758-5_2

[11] J. Li, X. Li, and L. Yu, "Multi-uav cooperative coverage path planning in plateau and mountain environment," in *2018 33rd Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, May 2018, pp. 820–824.

[12] Weimin Liu, Sujian Li, Fanggeng Zhao, and Aiyun Zheng, "An ant colony optimization algorithm for the multiple traveling salesmen problem," in *2009 4th IEEE Conference on Industrial Electronics and Applications*, May 2009, pp. 1533–1537.

[13] L. Ding, D. Zhao, H. Ma, H. Wang, and L. Liu, "Energy-efficient min-max planning of heterogeneous tasks with multiple uavs," in *2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE, 2018, pp. 339–346.

Authorized licensed use limited to: Ericsson. Downloaded on September 24,2024 at 20:02:51 UTC from IEEE Xplore. Restrictions apply.