Efficient Tabu Search Algorithm for the Cyclic Inspection Problem

Wojciech Bożejko[1(✉)], Radosław Grymin[1], Jarosław Pempera[1], and Mieczysław Wodecki[2]

[1] Department of Control Systems and Mechatronics, Wrocław University of Science and Technology, Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland
{wojciech.bozejko,radoslaw.grymin,jaroslaw.pempera}@pwr.edu.pl
[2] Department of Telecommunications and Teleinformatics, Wrocław University of Science and Technology, Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland
mieczyslaw.wodecki@pwr.edu.pl

Abstract. Unmanned aerial vehicles (UAVs) are a powerful tool for remote monitoring of objects requiring early anomaly detection, emergency intervention, or measurement data collection. We consider the problem of determining the optimal path of a UAV performing remote inspection of objects. The UAV inspects objects repeatedly (infinitely many times) every specified period of time. We propose an effective heuristic algorithm based on the tabu search method. In its construction, we used some properties of the problem under consideration.

1 Introduction

COVID-19 pandemic in 2020 introduced many limitations in people worldwide's movement and work. The need for new technologies that allow performing people's labor without exposing them to the virus emerged. Many businesses switched to remote work. However, there are still such branches of business, including security and supervision of facilities (consisting of periodic inspections), which require a human presence. Intelligent inspection systems may replace human in doing such work. A popular and increasingly used solution used in inspection is the use of unmanned aerial vehicles (UAVs). Due to the finite battery capacity and long charging time, they require efficient path planning algorithms.

We should note that the battery charging station's location can be virtually any due to relatively low technical requirements. As a rule, only access to electricity (including solar energy) and a small inductive charging area are required. For this reason, systems of this type can be installed in virtually any field conditions. They can be installed permanently (e.g. inspection of forest facilities) or ad hoc (e.g., an inspection of disasters). However, from the point of view of designing inspection systems, we can determine the optimal route, including the inspection object, and then adjust the charging station's location to the route.

A particular case of the problem discussed in this paper is Close Enough Traveling Salesman Problem (CETSP), currently the object of research by many

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
H. Sanjurjo González et al. (Eds.): SOCO 2021, AISC 1401, pp. 751–757, 2022.
https://doi.org/10.1007/978-3-030-87869-6_71

scientists. CETSP differs from the problem under consideration in that there is a fixed location (battery charging station) in the flight path at which each UAV flight begins and ends.

CETSP was explored by Tekdas and Isler, 2009 [15]. They solved the problem of collecting information from wireless sensor network by many mobile robots (m-CETSP, m-Close Enough Traveling Salesman Problem).

Coutinho et al. 2016 [4] presented exact methods based on second order cone programming (SOCP) and the Branch and Bound (B&B) method, Yang et al. 2018 [16] presented a hybrid algorithm based on the Particle Swarm Optimization (PSO) and the Genetic Algorithm (GA). Mennell 2009 [11] and Mennell et al. 2011 [10] proposed a heuristic based on the intersections of regions called Steiner zones. Yuan et al. [17] created an efficient evolutionary algorithm that was able to find the shortest path for all instances despite the long execution time.

Some specific cases of the CETSP problem were solved with polynomial approximation schemes by Arkin and Hassin 1993 [1], Mata and Mitchell 1995 [9] and Dumitrescu and Mitchell 2003 [5], Carrabs et al. 2017 [3]. Behdani and Smith, 2014 presented strong lower and upper boundaries for CETSP based on mixed integer programming (MIP).

Behdani and Smith, 2014 [2] proposed an upper and a lower bound based on the partitioning scheme of the TSPN problem. Taking advantage of the fact that the optimal solution of the TSPN problem can be presented as a finite series of points representing regions, a natural approach to obtain an acceptable path is to approximate the solution space with a discrete set of points. This solution allows obtaining an upper bound of the length of the optimal TSPN path. Behdani and Smith's approach is based on dividing a continuous solution space into smaller sets and identifying those that may contain a point representing the optimal TSPN path. The review [12] describes the most promising applications for aerial drones and insights into modeling approaches.

1.1 Problem Description and Properties

A set of vertices is given $V = \{1, \ldots, n\}$ in a two-dimensional space with coordinates (x_i, y_i), $i \in V$. Each vertex of i lies at the center of a region R_i which is a circle of radius $r_i > 0$. It is assumed that the drone reaching any point in the area means that the object has been properly inspected and $(x_i, y_i) \neq (x_j, y_j)$, $\forall i, j \in V$, $i \neq j$, i.e. no vertices are duplicated.

Solving the Cyclic Inspection Problem consists in determining the coordinates of (x_i, y_i) and the circle $\sigma = (\sigma_1, \ldots, \sigma_n)$, $\sigma_i \in V$, representing the order in which the vertices are visited in such a way that the path passing through the points (x_i, y_i) creates the Hamilton cycle of the shortest length.

For a given circle σ and coordinates $X = [x_i]$, $Y = [y_i]$ $i \in V$ the length of path can be determined from the expression 1.

$$L(\sigma, X, Y) = \sum_{i=1}^{n-1} d(p_{\sigma_i}, p_{\sigma_{i+1}}) + d(p_{\sigma_n}, p_{\sigma_1}), \tag{1}$$

where $d(p_i, p_j)$ is the Euclidean distance between the points $p_i = (x_i, y_i)$ and $p_j = (x_j, y_j)$.

Finally, we want to find circle σ^* and coordinates X^*, Y^* such that $L(\sigma^*, X^*, Y^*)$ is the smallest possible.

Theorem 1. *Let R_a be an area completely contained within an area R_b then the inspection point of area R_a will be a valid point in area R_b.*

From property 1 we conclude that all areas that contain other areas can be omitted. In this way, we decrease the number of points at which the objects will be inspected.

Theorem 2. *For each path that is a solution to an inspection problem, we can always assign inspection points such that they all lie on the edges of the visibility circles.*

For proof, it suffices to note that any path to a point inside the circle of visibility must first enter through some point on the periphery of that circle.

Using the property 2, the inspection point of the object i can be unambiguously described with the angle α_i between the straight line parallel to the OX axis and the straight line connecting the inspection object location and the inspection point.

For the area R_i and the angle α_i, the coordinates (x_i, y_i) of the inspection point are given by the equations:

$$x_i(\alpha_i) = X_i + r_i \cos(\alpha_i), \tag{2}$$

$$y_i(\alpha_i) = Y_i + r_i \sin(\alpha_i). \tag{3}$$

Finally, the path length for fixed sequence σ and $\alpha = (\alpha_1, \ldots, \alpha_n)$ is

$$L(\sigma, \alpha) = \sum_{i=1}^{n-1} d(p_{\sigma_i}(\alpha_{\sigma_i}), p_{\sigma_{i+1}}(\alpha_{\sigma_{i+1}})) + d(p_{\sigma_n}(\alpha_{\sigma_n}), p_{\sigma_1}(\alpha_{\sigma_1})), \tag{4}$$

where $p_i(\alpha_i) = (x_i(\alpha_i), y_i(\alpha_i))$.

Figure 1 shows the circles of visibility for the three objects, the angles that determine the inspection points and the flight path of the UAV.

For a given circle σ function $L(\sigma, \alpha)$ is a real-valued function of a n number of real-valued inputs. We can determine the values of arguments minimizing $L(\sigma, \alpha)$ by several methods. In our opinion, the most effective algorithm to solve such a problem is the one proposed by Potts [13]. Another reason for using this algorithm is the widespread availability of software libraries containing the implementation of this algorithm or algorithms based on it.

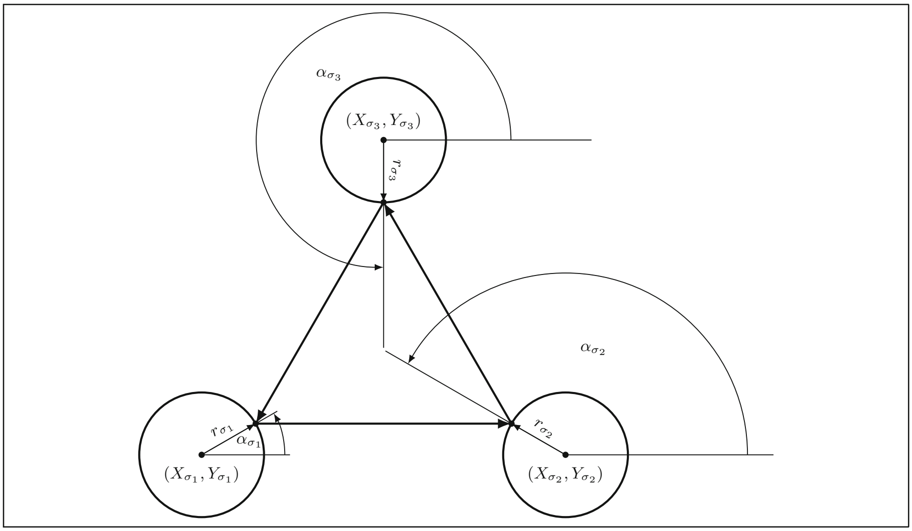

Fig. 1. An example of inspecting three objects

2 Tabu Search Algorithm

In the previous section, we showed how we can effectively determine the UAV flight path for a given sequence of inspection objects. In the current section, we propose an algorithm for determining the sequence of inspection objects for which the path with the shortest length exists. From among many combinatorial optimization methods, we chose the tabu search method proposed by Fred Glover [6–8]). The choice of this method was motivated by two facts: (i) algorithms based on this method have experimentally proven high efficiency for many optimization problems and (ii) in the tabu search algorithm, it is possible to effectively use the results of the objective function calculations performed for one solution when calculating the objective value for other solutions.

Let $L(\sigma)$ denote the length of the shortest path determined by the algorithm discussed in the previous section for the circle σ. In each iteration of the algorithm, its neighborhood $\mathcal{N}(\sigma)$ is determined for the current solution σ. The solutions of the neighborhood are generated by reordering some subsequence of σ. The length of such sequences is relatively small so the neighbors are similar to σ.

The neighborhood $\mathcal{N}(x_i)$ is searched for finding the best solution σ', which becomes the current solution on the next iteration. Certain features of the last movement performed are stored in the tabu list. Proper interpretation of these features makes it impossible to return to solutions already considered.

When the best solution from the neighborhood is forbidden by the taboo list, but the value of the objective function for this solution is better (lower) than for the best solution found up to this moment during the searching process, we choose such solution as a new current solution for the next iteration. Of course, the best solution found so far is being updated. The algorithm ends its execution after a predetermined number of iterations.

The most popular neighborhood types used in local search algorithms for routing problems are based on twist and 2-Opt types of move. The *twist* move used in our algorithm is described by the (a, b) pair. The twist modification described by the (a, b) pair consists in reversing the order of items into position from a to b. For example, for $\sigma = (1, 2, 3, 4, 5, 6, 7)$ and we obtain neighbor $(1, 2, 6, 5, 4, 3, 7)$ for *twist* (3,6). The tabu list remembered $LT = 7$ last twists, which, as research has shown, effectively enables the return to the previous solutions.

3 Computational Experiments

In order to evaluate the efficiency of the algorithm proposed in the previous section (hereinafter abbreviated as TS) we designed and performed computational test. The solution generated by our algorithm was compared with solution generated by MIP-based algorithm (mixed integer programming) proposed by Behdani and Smith [2]. The benchmark set contains 2 groups of instances CETSP-6 and CETSP=12, each group contains 10 instances with the same number of inspection objects. The first set includes $n = 7$ inspection objects with visibility circles with radius $r_i = 0.25$, while the second set includes 13 objects with visibility circles with radius $r_i = 0.5$. Starting permutation was chosen as natural one $(1, 2, \ldots, n)$.

The experiments were performed on a machine belonging to Wrocław Centre for Networking and Supercomputing with Intel Xeon E5-2670 v3 2.3GHz processor (24 physical cores), 64 GB of RAM, CentOS 6.10. Tabu search algorithm was implemented in Python language (version of interpreter 3.7.4) (set size of tabu list to 7,). Incorporated modified Powell's method [13,14] from SciPy library (SciPy version 1.5.2). For each instance and each algorithm $A \in \{MIP, TS\}$ were collected: $L(A)$ – the length of path generated by algorithm A, and CPU – running time (in seconds), and evaluated the percentage relative distance to the best solution $PRD(A) = \frac{L(A)-L^*}{L^*} \cdot 100\ \%$, where $L^* = \min\{L(IA), L(TS)\}$.

Table 1 contains the results of running compared algorithms. From analyze the quality of solutions generated by both algorithm one may conclude that the TS algorithm found the best solution for all instances for the first group of instances (see results marked in bold). The PRD value for MIP is in the range 1.9–4.9% for this group (average 2.5%). In practice, this means that thanks to better planning of the UAV path, you can save nearly 2.5% of energy. In the case of the second group of instances, the quality of solutions generated by both algorithms is similar, with a slight advantage of the TS algorithm. The mean PRD value of the TS algorithm is 0.66 while the MIP algorithm is 0.85. The advantage of the TS algorithm over the MIP algorithm is also manifested in the runtime. The TS algorithm generates better solutions in over 20 times shorter time.

Table 1. PRD and CPU time for MIP and TS algorithms

Instance	MIP – Behdani and Smith			TS		
	$L(MIP)$	$PRD[\%]$	$CPU[s]$	$L(TS)$	$PRD[\%]$	$CPU[s]$
$r_i = 0.25$, $n = 7$						
CETSP-6-1	34.2768	2.5	0.8	**33.7750**	0.0	0.04
CETSP-6-2	27.8568	2.3	0.8	**27.5119**	0.0	0.06
CETSP-6-3	25.6439	2.6	1.6	**25.3383**	0.0	0.05
CETSP-6-4	37.0838	1.9	0.5	**36.7176**	0.0	0.05
CETSP-6-5	22.9133	4.8	1.2	**21.7241**	0.0	0.07
CETSP-6-6	28.1387	3.4	0.8	**27.5811**	0.0	0.07
CETSP-6-7	35.0102	3.1	0.7	**34.3591**	0.0	0.05
CETSP-6-8	24.514	4.6	1.5	**23.7608**	0.0	0.05
CETSP-6-9	29.0243	4.3	0.8	**28.1325**	0.0	0.05
CETSP-6-10	34.9533	2.8	0.9	**34.5152**	0.0	0.05
Average		2.5	0.8		0.0	0.04
$r_i = 0.5$, $n = 13$						
CETSP-12-1	**34.113**	0.0	17.6	34.479	1.1	0.3
CETSP-12-2	**45.235**	0.0	4.3	45.796	1.2	0.3
CETSP-12-3	**33.204**	0.0	6.8	33.721	1.6	0.3
CETSP-12-4	33.253	0.2	16.5	**33.196**	0.0	0.3
CETSP-12-5	**38.077**	0.0	10.4	38.991	2.4	0.2
CETSP-12-6	36.163	1.0	2.9	**35.813**	0.0	0.2
CETSP-12-7	35.092	2.8	3.5	**34.152**	0.0	0.2
CETSP-12-8	**38.410**	0.0	3.4	38.509	0.3	0.2
CETSP-12-9	32.572	3.0	11.1	**31.632**	0.0	0.4
CETSP-12-10	40.716	1.5	4.4	**40.122**	0.0	0.2
Average		0.85	8.09		0.66	0.26

4 Conclusion

We designed the new approach for solving the discrete-continues problem of cyclic inspection. Thanks to our original fast path planning method for UAVs and use of TS algorithm, we can find comparable or even better solutions than solutions generated by algorithms known from literature in much shorter time. The further stage of research on the proposed hybrid approach is to examine the impact of the methods of formulating a low-level continuous optimization problem on the computation time. In the next steps, it would also be necessary to prove the properties of the objective function for a given permutation (whether it is a convex optimization, examine the number of extremes, whether there are many global optimas, etc.).

Acknowledgments. The paper was supported by the National Science Centre of Poland, grant OPUS no. 2017/25/B/ST7/02181, and computational grant No. 96 of Wrocław Centre for Networking and Supercomputing.

References

1. Arkin, E.M., Hassin, R.: Approximation algorithms for the geometric covering salesman problem. Discrete Appl. Math. **55**(3), 197–218 (1994)
2. Behdani, B., Smith, J.C.: An integer-programming-based approach to the close-enough traveling salesman problem. INFORMS J. Comput. **26**(3), 415–432 (2014)
3. Carrabs, F., Cerrone, C., Cerulli, R., Gaudioso, M.: A novel discretization scheme for the close enough traveling salesman problem. Comput. Oper. Res. **78**, 163–171 (2017)
4. Coutinho, W.P., Nascimento, R.Q.d., Pessoa, A.A., Subramanian, A.: A branch-and-bound algorithm for the close-enough traveling salesman problem. INFORMS J. Comput. **28**(4), 752–765 (2016)
5. Dumitrescu, A., Mitchell, J.S.: Approximation algorithms for tsp with neighborhoods in the plane. J. Algorithms **48**(1), 135–159 (2003)
6. Glover, F.: Future paths for integer programming and links to artificial intelligence. Comput. Oper. Res. **13**(5), 533–549 (1986)
7. Glover, F.: Tabu search–part I. ORSA J. Comput. **1**(3), 190–206 (1989)
8. Glover, F.: Tabu search–part II. ORSA J. Comput. **2**(1), 4–32 (1990)
9. Mata, C.S., Mitchell, J.S.: Approximation algorithms for geometric tour and network design problems. In: Symposium on Computational Geometry, pp. 360–369. Citeseer (1995)
10. Mennell, W., Golden, B., Wasil, E.: A Steiner-zone heuristic for solving the close-enough traveling salesman problem. In: 2th INFORMS Computing Society Conference: Operations Research, Computing, and Homeland Defense (2011)
11. Mennell, W.K.: Heuristics for solving three routing problems: close-enough traveling salesman problem, close-enough vehicle routing problem, sequence-dependent team orienteering problem. Ph.D. thesis (2009)
12. Otto, A., Agatz, N., Campbell, J., Golden, B., Pesch, E.: Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: a survey. Networks **72**(4), 411–458 (2018)
13. Powell, M.J.: An efficient method for finding the minimum of a function of several variables without calculating derivatives. Comput. J. **7**(2), 155–162 (1964)
14. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: Numerical Recipes (any edition). Cambridge University Press, Cambridge (2007)
15. Tekdas, O., Isler, V., Lim, J.H., Terzis, A.: Using mobile robots to harvest data from sensor fields. IEEE Wirel. Commun. **16**(1), 22–28 (2009)
16. Yang, Z., et al.: A double-loop hybrid algorithm for the traveling salesman problem with arbitrary neighbourhoods. Eur. J. Oper. Res. **265**(1), 65–80 (2018)
17. Yuan, B., Orlowska, M., Sadiq, S.: On the optimal robot routing problem in wireless sensor networks. IEEE Trans. Knowl. Data Eng. **19**(9), 1252–1261 (2007)