Covering segments on a line with drones

Sergey Bereg*[1], José-Miguel Díaz-Báñez†[2], Alina Kasiuk‡[2], Miguel-Angel Pérez-Cutiño§[2], and Fabio Rodríguez¶[2]

[1]Department of Computer Science, University of Texas at Dallas, USA
[2]Department of Applied Mathematics, University of Seville, SPAIN

Abstract

Covering a set of segments in a plane with vehicles of limited autonomy is a problem of practical interest. The limited battery endurance imposes periodical visits to a static base station. Typically, two optimization problems are considered: minimize the number of tours, and minimize the total traveled distance. In a general setting, the problems are NP-hard and in this letter, we study the one-dimensional version. For covering segments on a line, we design efficient solutions for both optimization problems. First, we design a Greedy algorithm that is optimal for the first task, and for both tasks when only one segment is considered. Being n and m the number of segments and tours of an optimal solution, respectively, our algorithm runs in $O(m+n)$ time. For the second criterion, our solution is based on Dynamic Programming and runs in $O(n^2) + O(nm)$ time.

1 Introduction

Trajectory optimization through linear segments is of practical interest in the robotics community. Road network patrolling, anomaly detection in solar power plants, power lines inspection and other similar infrastructures with unmanned vehicles are studied in various pieces of research [2, 1, 6]. In this work, we will use the term drone, though this research may be applied to any agent with limited autonomy. The use of drones or Unmanned Aerial Vehicles (UAVs), commonly called drones, has been proposed for the efficient maintenance of infrastructures, in order to reduce potential risks and costs for the distribution companies [7, 3]. The battery life limit of these small-size robots severely restricts the duration of the mission, as it becomes impossible to complete the overall coverage with a single tour. Therefore, considering each tour should start and end at a base station, the problem of minimizing the total cost of travel is hard in general and some heuristics have been considered in the literature [4, 5].

Two optimization problems can be formulated when the objects to be covered are line segments. Given a set of line segments S with any distribution in the plane, a depot or a base location O from where the robots can be launched and recharged and, a real number L, we consider:

- $MinTours$-problem: Finding the minimum number of tours $t_1, \cdots, t_p$ covering S, that is,

$$S \subset \bigcup_{i=1}^{p} t_i.$$

- $MinDistance$-problem: Compute a set of tours that covers S with minimum total length.

A tour t is considered to be *valid* if it starts and ends at O, and the length of t is at most L. The length of a tour is the sum of the Euclidean distance between its consecutive vertices; the length of a set of tours is the sum of the lengths of each of its elements. The NP-hardness of $MinTours$- and $MinDistance$-problems in the plane can be proved by a reduction from the Traveling Salesman Problem (TSP). However, in this paper we show that the one-dimensional case related to both problems, where segments are located through a line, can be solved in polynomial time. We consider several scenarios, designing efficient algorithms capable of finding the optimal solution. The paper is structured as follows: Section 2 formally defines the considered problems; Section 3 describes the optimal solution to the problem of finding the minimum number of tours; and Section 4 focuses on finding the set of tours covering S with minimum total length. In this version, we omit several proofs due to space restrictions.

2 Problem formulation

Let $S = \{s_1, \cdots, s_n\}$ be a set of disjoint segments arranged on a line, O be a point on the plane corresponding to a base station, and L be the maximum distance that a drone with constant battery life can travel. The problem is to find a set of drone routes (*tours*) with lengths no greater than L starting and ending at O so that they jointly traverse all segments

*Email: besp@utdallas.edu
†Email: dbanez@us.es
‡Email: akasiuk@us.es
§Email: migpercut@alum.us.es
¶Email: frodriguex@us.es

in S with minimum total cost. Two objective functions are considered: the number of tours or the total traveled distance (sum of the lengths of the tours).

Since segments in S lie on a line, we define the problems using the following notations. Let $s_i = [a_i, b_i], i = 1, 2, \ldots, n$ be n disjoint intervals on the line $y = 0$ such that $a_1 < a_2 < \cdots < a_n$; a_1, b_n are the edges of S. Let $O = (0, -h)$ be the base station and $L > 0$ be the maximum length of a tour using the full battery. Formally, the problem is to compute a set of tours $T = \{t_1, t_2, \ldots, t_m\}$ covering S so that:

- $1DMinTours$-problem: the number of tours $m = |T|$ is minimized.
- $1DMinDistance$-problem: $\sum_{i=1}^{m} l_i$ is minimized, where l_i is the length of t_i.

For simplicity, we assume that the tours of T are given ordered, that is, from left to right or from right to left. In addition, we consider other important notations and definitions. For an interval $s_i = [a_i, b_i]$, we term a_i, b_i as the left and right point respectively of s_i. This concept is extended for any tour t: the left point of t is the leftmost point of t that lies on the line defined by the intervals covered by t; the right point of t is defined analogously. If a point $x \in s_i$ for some $s_i \in S$, then we assume the relaxation of $x \in S$. This is important to define subsets of S as $S_{p,q} = \{x : x \in S, p \leq x \leq q\}$; then $T_{p,q}$ is the set of tours covering $S_{p,q}$. If p, q are the left and right points respectively of the tour t_i, then we define the portion of S covered by t_i as $S^i_{p,q}$, and $S - t_i$ as the part of S not covered by t_i. Finally, we consider that t_i is a maximal tour if $l_i = L$, we term m as the minimal number of tours to cover S, and T^* is the optimal set of tours covering S. See Figure 1 for a visual explanation of some of the aforementioned definitions.

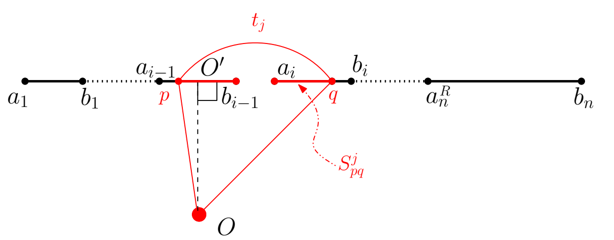


Figure 1: An example of a set S of intervals. A tour t_j (in red) is the path $OpqO$. The portion of S covered by this tour is $S^j_{pq} = [p, b_{i-1}] \cup [a_i, q]$.

3 Minimizing the number of tours

In this section, we show that the 1DMinTours-problem can be solved using the following greedy approach. For a tour $t = OpqO$, let $S - t$ be the closure of part of S not covered by t.

Greedy Strategy (GS): *Let f be the farthest point from O in S. If S can be covered by one tour, perform a minimal length tour t covering S, else perform a maximal tour t covering f and update $S := S - t$.*

In the following, we prove that GS retrieves an optimal solution (it is easy to see that the optimal solution is not necessarily unique).

Theorem 1 *GS computes an optimal solution for minimizing the number of tours.*

Proof. Proof by induction on m, the minimum number of tours.

Base Case: If all segments of S can be covered with one tour, the greedy algorithm computes only one tour using the farthest point from O.

Inductive Step: Suppose that the minimum number of tours covering a set S is at least two, i.e. $m \geq 2$. Let $f \in S$ be the farthest point from O and let t_f be the maximal tour covering f. Assume w.l.o.g. that $f = b_n$ (the proof is analogous if $f = a_1$). Let $T^* = \{t_1, t_2, \ldots, t_m\}$, t_m be an optimal solution such that the tour t_m reaches f. Let S^* be the set of points in S covered by tours $t_1, t_2, \ldots, t_{m-1}$. Let $S^{'}$ be the set of points in S not covered by tour t_f. Since t_f is maximal, $S' \subseteq S^*$. Then S' can be covered by $m-1$ tours (for example, $t_1, t_2, \ldots, t_{m-1}$). By the induction hypothesis, the greedy algorithm covers S' by at most $m-1$ tours. Therefore the greedy algorithm computes at most m covering tours for S. Since m is the minimum number of tours covering a set S, the number of tours computed by the greedy algorithm is exactly m. □

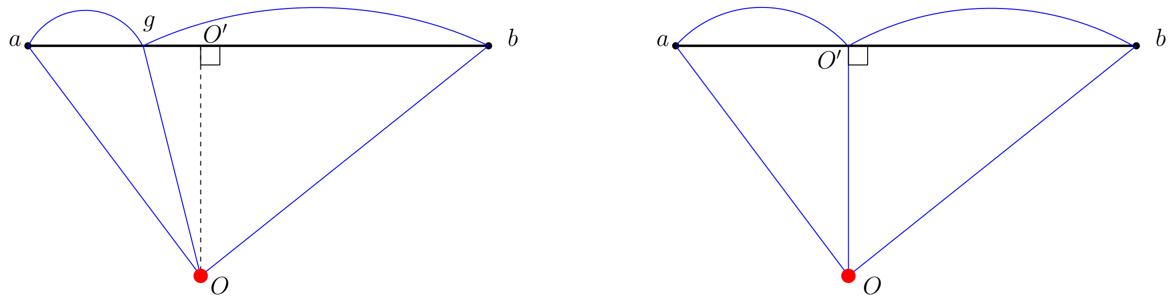
Theorem 2 *The 1DMinTours-problem can be solved in $O(m \log n)$ or $O(m + n)$, where n is the number of segments and m is the minimal number of tours.*

4 Minimizing the total distance

4.1 One segment

First, note that GS is not optimal for minimizing the total distance, even for restricted scenarios where only one segment is considered (i.e. $n = 1$). This is the case of Figure 2, where the solution provided by GS (shown in (a)) is worst than the solution (shown in (b)). We extend GS to optimally solve the 1DMinDistance-problem for only one segment.

Greedy Strategy with Projection Point (GSP): *Go to the untraveled point of S farthest from O, and perform a maximal tour while the projection point O' is not reached. If possible, cover the last part with one tour; otherwise, select the two tours containing the projection point.*



(a) Greedy solution.

(b) Coming to the base on the projection point.

Figure 2: GS is not optimal to cover segment $[a, b]$ when the optimal set of tours for minimizing the total distance includes the projection point O'.

Theorem 3 *GSP is optimal for the 1DMinDistance-problem with only one segment.*

Proof. The optimal solution for minimizing the total distance in one segment has the particularity that the tours that do not include O' have to be of maximum length. Otherwise, for a tour of non-maximum length, we can change its returning point for a point closer to O that is also valid but with a lower distance to the base. On the other hand, in the optimal solution (this is unique), the projection point O' can be covered by one or two tours (since points in S are covered at a maximum of two times). GSP is optimal since it uses the aforementioned characterizations to build the solution. First, GSP extracts the maximum-length tours that do not contain O'. Finally, it checks if the rest of the segment can be covered with one tour (optimal), or if we need two. In the second case, the optimal partition of the segment uses the projection point, as this is the closest distance from O to S. $\square$

Theorem 4 *1DMinDistance-problem for one segment can be solved in $O(m)$ where m is the number of tours of the optimal solution.*

For two o more segments, the greedy approach does not solve the problem. The reason is that each gap between two segments poses a decision problem: covering it with a tour of maximum length (when it is possible) or finishing the tour at the end of a segment and start a new tour from the next one.

4.2 Segments to one side

In this letter, we only show how to solve the scenario where all segments are on one side of the projection point. The general case can be solved by using an extension of this case. Formally, we call 1DMD-one-side-problem to the 1DMinDistance-problem, with an additional restriction: either $0 \leq a_1$ or $a_n \leq 0$. Without loss of generality, we consider the case where $0 \leq a_1$.

Let us built a discrete set using the following approach: For every b_i, we consider the set of points C_i defined by the jumps of the greedy solution starting at b_i and continuing until a gap is reached, or all the segments are covered (Figure 3); each C_i contains at most m points. Let $C = \bigcup C_i, i \in [1 \dots n]$, be the set of candidate points defined with this strategy that contains, at most, nm points.

Lemma 5 *The right point q of any tour t_j in the optimal solution T^* satisfies $q \in C$.*

Proof. Assume $t_m \in T^*$ as the last tour with the rightmost point not in C. Let q be the right point of t_m and $s_i = [a_i, b_i]$ the segment where q lies; then $q \in (a_i, b_i)$. Hence, the tour t_{m+1} with leftmost point q, has a length lower than L because the rightmost point of t_{m+1} is in C. As $a_i < q$, we can increase the length of t_{m+1}, hence reducing the total distance. This contradicts that T^* is optimal. $\square$

Lemma 6 *The left point p of any tour t_j in the optimal solution T^* satisfies that p is the left point of some interval of S, or t_j is maximal.*

Proof. Assume $t_i \in T^*$ as a non-maximum length tour with the left point $p \in (a_i, b_i]$ for some interval in S. Then, we can increase t_i by moving p left towards, reducing the distance from the base station to p. This contradicts the optimality of T^*. Hence tours of non-maximal length lie only at the left point of some interval of S. $\square$

As a consequence of Lemma 6, it is straightforward to notice that the left point of a tour in the optimal solution is the left point of a segment, or is in C. Using this fact and Lemma 5, we design a polynomial algorithm based on dynamic programming. Our algorithm iterates over the sorted points of C, in ascending order. For every point $c_k \in C$, we compute the maximum-length tour starting on it, and its associated left point c'_k. We know that either $c'_k \in C$, or is the left point of some interval of S. Let j_k (j'_k) be the index of the

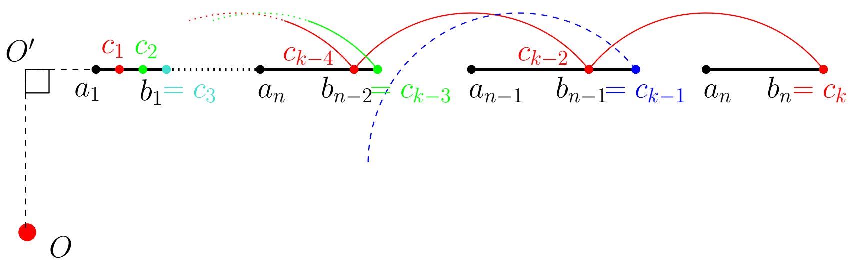


Figure 3: The one side case. Construction of the candidate set C.

segment where c_k (c'_k) is located, and be $\Sigma^*(c_k)$ the optimal cost for S_{a_1,c_k}. The formula for any c_k is as follows:

$$\Sigma^*(c_k) = \begin{cases} \operatorname{len}(a_1, c_k) & \text{if } a_1 = c'_k \\ \min\limits_{j'_k \leq j \leq j_k} \{\operatorname{len}(a_j, c_k) + \Sigma^*(b_{j-1})\} & \text{if } c'_k \notin C \\ \min\{L + \Sigma^*(c'_k), \min\limits_{j'_k < j \leq j_k} \{\operatorname{len}(a_j, c_k) + \Sigma^*(b_{j-1})\}\}, & \text{otherwise}, \end{cases} \tag{1}$$

where $\operatorname{len}(a_j, c_k)$ is the length of the tour that defines the interval S_{a_j,c_k}; and a_j (b_j) is the left (right) point of any segment contained within the maximum-length tour starting at c_k. A maximum of $n-1$ values of a_j needs to be checked for every c_k; one for every gap. We term the algorithm based on the formula 1 as DPOS (Dynamic Programming on One Side). As a consequence of Lemmas 5 and 6, we have:

Theorem 7 *DPOS is optimal for the 1DMD-one-side-problem.*

Theorem 8 *The 1DMD-one-side problem can be solved in $O(n^2) + O(nm)$, where n is the number of segments and m is the number of tours in the optimal solution.*

Acknowledgments

This work is partially supported by grants PID2020-114154RB-I00 and TED2021-129182B-I00 funded by MCIN/AEI/10.13039/501100011033 and the European Union NextGenerationEU/PRTR.

References

[1] Saurav Agarwal and Srinivas Akella. Line coverage with multiple robots. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3248–3254. IEEE, 2020.

[2] Sergey Bereg, Luis Evaristo Caraballo, and José Miguel Díaz-Báñez. Efficient inspection of underground galleries using k robots with limited energy. In *Iberian Robotics conference*, pages 706–717. Springer, 2017.

[3] Alvaro Calvo, Giuseppe Silano, and Jesús Capitán. Mission planning and execution in heterogeneous teams of aerial robots supporting power line inspection operations. In *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1644–1649. IEEE, 2022.

[4] Tucker Farrell, Kidus Guye, Rebecca Mitchell, and Guangdong Zhu. A non-intrusive optical approach to characterize heliostats in utility-scale power tower plants: Flight path generation/optimization of unmanned aerial systems. *Solar Energy*, 225:784–801, 2021.

[5] Mahdi Jemmali, Ali Kashif Bashir, Wadii Boulila, Loai Kayed B Melhim, Rutvij H Jhaveri, and Jawad Ahmad. An efficient optimization of battery-drone-based transportation systems for monitoring solar power plant. *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[6] František Nekovář, Jan Faigl, and Martin Saska. Multi-tour set traveling salesman problem in planning power transmission line inspection. *IEEE Robotics and Automation Letters*, 6(4):6196–6203, 2021.

[7] Joon-Young Park, Seok-Tae Kim, Jae-Kyung Lee, Ji-Wan Ham, and Ki-Yong Oh. Method of operating a gis-based autopilot drone to inspect ultrahigh voltage power lines and its field tests. *Journal of Field Robotics*, 37(3):345–361, 2020.