Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

A descent method for the Dubins traveling salesman problem with neighborhoods*

Zheng CHEN†, Chen-hao SUN, Xue-ming SHAO†‡, Wen-jie ZHAO

State Key Laboratory of Fluid Power and Mechatronic Systems,
School of Aeronautics and Astronautics, Zhejiang University, Hangzhou 310027, China

†E-mail: z_chen@zju.edu.cn; mecsxm@zju.edu.cn

Received Jan. 21, 2020; Revision accepted Apr. 28, 2020; Crosschecked June 11, 2020; Published online Aug. 13, 2020

Abstract: In this study, we focus mainly on the problem of finding the minimum-length path through a set of circular regions by a fixed-wing unmanned aerial vehicle. Such a problem is referred to as the Dubins traveling salesman problem with neighborhoods (DTSPN). Algorithms developed in the literature for solving DTSPN either are computationally demanding or generate low-quality solutions. To achieve a better trade-off between solution quality and computational cost, an efficient gradient-free descent method is designed. The core idea of the descent method is to decompose DTSPN into a series of subproblems, each of which consists of finding the minimum-length path of a Dubins vehicle from a configuration to another configuration via an intermediate circular region. By analyzing the geometric properties of the subproblems, we use a bisection method to solve the subproblems. As a result, the descent method can efficiently address DTSPN by successively solving a series of subproblems. Finally, several numerical experiments are carried out to demonstrate the descent method in comparison with several existing algorithms.

Key words: Dubins vehicle; Descent method; Dubins traveling salesman problem
https://doi.org/10.1631/FITEE.2000041 **CLC number:** TP242.6; V279

1 Introduction

When performing surveillance missions, a fixed-wing unmanned aerial vehicle (UAV) usually flies in an altitude hold mode with a constant cruise speed. In such an altitude hold mode, the kinematic fixed-wing UAV can be modeled as a Dubins vehicle which moves only forward at a constant speed with a minimum turning radius (Dubins, 1957). For this reason, the minimum-time surveillance of multiple ground targets by a fixed-wing UAV has been dubbed as the Dubins traveling salesman problem (DTSP) in the literature. Each DTSP consists of finding a shortest path of a Dubins vehicle visiting each target exactly once and finally returning to the initial target.

Since DTSP has been proven to be NP-hard (Ny et al., 2012), if the number of targets is large, it is difficult to solve in situ by exact algorithms. Therefore, many heuristic algorithms have been developed in three main categories, i.e., the decoupled approach (Savla et al., 2005; Ma and Castanon, 2006; Isaiah and Shima, 2015), transformation method (Obermeyer et al., 2010; Isaacs et al., 2011; Ny et al., 2012; Isaacs and Hespanha, 2013), and evolutionary approach (Machareret et al., 2012; Zhang et al., 2014).

While the transformation method and evolutionary approach perform well in finding high-quality solutions to DTSP, their computational burdens are not acceptable for practical scenarios when the control decisions have to be made in situ, if not exactly,

‡ Corresponding author
* Project supported by the National Natural Science Foundation of China (Nos. 61903331 and 61703366) and the Fundamental Research Funds for the Central Universities, China (No. 2019FZA4024)
ORCID: Zheng CHEN, https://orcid.org/0000-0002-3573-1546

at least efficiently. The decoupled approach solves DTSP in a hierarchical way. At a high level, the sequence of targets is ordered. Then, at a low level, by taking into account the kinematic constraint of a Dubins vehicle, the solution path is planned according to the sequence ordered at a high level.

The sequence of targets is usually ordered by the solution to a degenerate variant of DTSP, namely the Euclidean traveling salesman problem (ETSP). ETSP is different from DTSP because the minimum turning radius in ETSP is not bounded. Up to present, several well-developed algorithms (Lawler et al., 1985; Arora, 1998) have been available to solve ETSP with a complexity of $n \log n$, where n is the number of targets. Thus, using the solution to ETSP to order the sequence of targets is probably the most popular way.

Note that the shortest Dubins path between any two configurations (a configuration consists of a two-dimensional (2D) point and a heading orientation angle) belongs to a finite set of six candidate paths (Dubins, 1957). Thus, once the sequence of targets is ordered in advance, it is enough to find a specific heading orientation angle at each target so that the concatenated Dubins path is the shortest. In fact, several approximation-based algorithms have been developed to optimize the heading orientation angle at each target. In Savla et al. (2005), an alternating algorithm (AA) was developed to approximate the heading orientation angles. A single vehicle algorithm (SVA) was developed in Rathinam et al. (2007) to design the heading angle at each target. Based on the principle of receding horizon, a look-ahead algorithm (LAA), which is a natural extension of SVA, was developed in Ma and Castanon (2006). Based on the solution to the three-point Dubins problem (Chen and Shima, 2019), an iterative algorithm was designed in Sadeghi and Smith (2016) to optimize the heading angles.

Note that to complete a surveillance mission of multiple ground targets, a UAV may just need to reach a region above each target instead of arriving to a point exactly above each target. For instance, if a surveillance mission is to take a snapshot of each target, a UAV needs just to reach a point in a circular region above each target. This variant of DTSP is dubbed a DTSP with neighborhoods (DTSPN).

Several algorithms have been developed for DTSPN. Xin et al. (2014) developed two hybrid encoding-based differential evolution algorithms by adopting a complete encoding scheme and a partial encoding scheme, separately. Obermeyer (2009) used a genetic algorithm to address the DTSPN with polygonal target regions. Obermeyer et al. (2010) developed two sampling-based roadmap methods for a visual reconnaissance UAV, and transformed DTSPN to a variant of the generalized traveling salesman problem (TSP). Yu and Hung (2012) studied the DTSPN with overlapping target regions where the sequence of targets is ordered by the solution to ETSP, and proposed an alternating iterative algorithm to shorten the solution path. Faigl and Váňa (2018) studied a variant of the DTSPN with the vehicle speed not being constant, and determined the solution to the variant DTSPN as a sequence of Bézier curves. Based on convex optimization (Goaoc et al., 2013), Tuqan et al. (2019) developed a simplified path planning algorithm. Although significant improvements on efficient algorithms have been made for solving DTSPN, there still exists a gap between algorithms and their real-time applications. In fact, these algorithms are too computationally expensive to be applied to real-time scenarios (Váňa and Faigl, 2015). Unlike the aforementioned algorithms for DTSPN, the geometric properties for the solution to DTSPN were studied so that a decoupled algorithm for DTSPN was developed by Váňa and Faigl (2015).

In this study, to achieve a better trade-off between solution quality and computational cost, a descent algorithm (DA) is designed to solve DTSPN. This DA simultaneously optimizes the order of sequence of target regions and the configuration in each region. The core idea of DA is to decompose DTSPN into a series of subproblems, each of which is to find the shortest Dubins path from a fixed configuration to another fixed configuration via an intermediate circular region. Through successively solving the subproblems, the length of the solution to DTSPN generated by DA decreases and converges to a cluster point. The efficiency of DA is influenced mainly by the time needed to solve each subproblem. Using the geometric properties of the subproblem established in Váňa and Faigl (2015) and Bhargav et al. (2019), a real-valued function is established, whose root gives the solution to the subproblems. As a result, a simple bisection method can be directly employed to solve the subproblems. This allows DA to

solve DTSPN efficiently, which will be illustrated by numerical simulations.

2 Problem formulation

The state of a Dubins vehicle can be represented as a configuration $[(x,y),\theta]\in \mathrm{SE}(2)$, where $(x,y)\in \mathbb{R}^2$ denotes the position in the 2D plane and $\theta\in \mathbb{S}^1$ is the heading orientation angle of the vehicle with respect to a reference direction. Without loss of generality, assume that the speed is one. Denote $\rho>0$ as the minimum turning radius of the Dubins vehicle. Then, the kinematics of the Dubins vehicle is expressed as

$$\begin{cases}\dot{x}(t)=\cos\theta(t),\\ \dot{y}(t)=\sin\theta(t),\\ \dot{\theta}(t)=\dfrac{u(t)}{\rho},\end{cases} \tag{1}$$

where $t\in\mathbb{R}_+$ is the time and $u\in[-1,1]$ is the control parameter, representing the lateral acceleration of the Dubins vehicle.

For simplification, denote $\mathcal{C}^r(\boldsymbol{z})\subset\mathbb{R}^2$ as a circular region centered at $\boldsymbol{z}\in\mathbb{R}^2$ with a radius $r>0$. Then, we aim to address the following DTSPN problem:

Problem 1 (DTSPN) Let $n\geq 3$ be the number of circular regions and $r>0$ the radius of each circular region. Given a sequence of target points $(\boldsymbol{z}_1,\boldsymbol{z}_2,\ldots,\boldsymbol{z}_n)$, then DTSPN consists of steering system (1) from a point $\boldsymbol{z}$ in $\mathcal{C}^r(\boldsymbol{z}_1)$, visiting at least one point in each region of $\mathcal{C}^r(\boldsymbol{z}_i)$, and returning to the point $\boldsymbol{z}\in\mathcal{C}^r(\boldsymbol{z}_1)$, so the resulting path is the shortest among all the reasonably controlled paths.

The following assumption is considered in the study:

Assumption 1 The distance between any two circular regions is at least four times the minimum turning radius ρ.

Unlike the variant of DTSPN in Váňa and Faigl (2015), the formulation of DTSPN in Problem 1 does not fix the sequence of target regions, and thus Problem 1 is more difficult to solve. In the next section, a gradient-free DA will be designed for the DTSPN defined in Problem 1.

3 Algorithm for DTSPN

Given any two configurations $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$ in SE(2), denote

$$F:\mathrm{SE}(2)\times\mathrm{SE}(2)\to\mathbb{R},\ (\boldsymbol{y}_1,\boldsymbol{y}_2)\mapsto F(\boldsymbol{y}_1,\boldsymbol{y}_2) \tag{2}$$

as the length of the shortest Dubins path from $\boldsymbol{y}_1$ to $\boldsymbol{y}_2$.

Let

$$\Sigma=(\sigma_1,\sigma_2,\ldots,\sigma_n)$$

be a sequence of integers from 1 to n. Then, it is clear that the DTSPN defined in Problem 1 is equivalent to the following combinatorial and finite optimization problem:

$$\begin{aligned}&\min_{\boldsymbol{y}_{\sigma_i},\Sigma}\left\{\sum_{i=1}^{n}F(\boldsymbol{y}_{\sigma_i},\boldsymbol{y}_{\sigma_{i+1}})\right\}\\ &\text{s.t. }\boldsymbol{y}_{\sigma_i}\in\mathcal{C}^r(\boldsymbol{z}_{\sigma_i})\cup[0,2\pi],\ i=1,2,\ldots,n.\end{aligned} \tag{3}$$

Converting the infinite optimization problem of Problem 1 to problem (3) allows to design an efficient gradient-free DA to solve DTSPN, as presented in Algorithm 1.

In steps 1 and 2, we assign the initial parameters for DTSPN and the while loop. With these initial parameters, the sequence of targets is ordered by the solution to ETSP in step 3. The sequence $(L_k)_{k\in\mathbb{N}}$ denotes the length of the solution path generated by each step in the while loop. In steps 6–9, we adjust the configurations in the circular regions of even numbers while fixing the configurations in the circular regions of odd numbers. Then, the configurations in the circular regions of odd numbers are optimized in steps 10–13 with the configurations in the circular regions of even numbers being fixed. As the configuration in each circular region is optimized by steps 6–13, the sequence obtained in step 3 may not be optimal any more. Thus, in steps 14–17 the sequence of targets is updated. The UPDATE sequence in step 14 can use an efficient algorithm presented in Tran (2019). The output in step 18 gives the sequence of configurations that lies in the circular regions. Successively concatenating these configurations by the Dubins path will give rise to the solution path of DTSPN. Note that not only the configurations in circular regions are adjusted but also the sequence of configurations is reordered in the while loop.

Algorithm 1 Descent algorithm (DA)

1: **Input:** set $r>0$ and assign a value to $\boldsymbol{z}_i$
// r denotes the radius and $\boldsymbol{z}_i$ the center of the i^{th}
// circular region
2: **Initialization:** Let $k=1$ and $L_0=\infty$, and assign a configuration in $\mathcal{C}^r(\boldsymbol{z}_i)\cup[0,2\pi]$ to $\boldsymbol{y}_i^k$ and a small positive value to ε
3: $(\sigma_1,\sigma_2,\ldots,\sigma_n)=\text{ETSP}(\boldsymbol{y}_1,\boldsymbol{y}_2,\ldots,\boldsymbol{y}_n)$
// using the solution to ETSP to order the
// sequence
4: $L_1=\sum_{j=1}^{n}F(\boldsymbol{y}_{\sigma_j}^k,\boldsymbol{y}_{\sigma_{j+1}}^k)$
// the length of the Dubins path consecutively
// connects $\boldsymbol{y}_1^k,\boldsymbol{y}_2^k,\ldots,\boldsymbol{y}_n^k$
5: **while** $L_k-L_{k-1}>\varepsilon$ **do**
// while the solution is improving, this loop
// continues
6: **for** $j=1:\text{int}(n/2)$ **do**
// int$(\cdot)$ presents the integer part
7: $i=2j-1$
8: $\boldsymbol{y}_{\sigma_{i+1}}^{k+1}=\underset{\boldsymbol{y}\in\mathcal{C}^r(\boldsymbol{z}_{\sigma_{i+1}})\cup\mathbb{S}}{\text{argmin}}\{F[\boldsymbol{y}_{\sigma_i}^k,\boldsymbol{y}]+F[\boldsymbol{y},\boldsymbol{y}_{\sigma_{i+2}}^k]\}$
9: **end for**
10: **for** $j=1:\text{int}(n/2)$ **do**
11: $i=2j$
12: $\boldsymbol{y}_{\sigma_{i+1}}^{k+1}=\underset{\boldsymbol{y}\in\mathcal{C}^r(\boldsymbol{z}_{\sigma_{i+1}})\cup\mathbb{S}}{\text{argmin}}\{F[\boldsymbol{y}_{\sigma_i}^{k+1},\boldsymbol{y}]+F[\boldsymbol{y},\boldsymbol{y}_{\sigma_{i+2}}^{k+1}]\}$
13: **end for**
14: $(\hat{\sigma}_1,\hat{\sigma}_2,\ldots,\hat{\sigma}_n)=\text{UPDATE}(\boldsymbol{y}_1,\boldsymbol{y}_2,\ldots,\boldsymbol{y}_n)$
// randomly generate a sequence
15: **if** $\sum_{j=1}^{n}F(\boldsymbol{y}_{\sigma_j},\boldsymbol{y}_{\sigma_{j+1}})>\sum_{j=1}^{n}F(\boldsymbol{y}_{\hat{\sigma}_j},\boldsymbol{y}_{\hat{\sigma}_{j+1}})$ **then**
// if the new sequence induces a shorter
// path
16: $(\sigma_1,\sigma_2,\ldots,\sigma_n)=(\hat{\sigma}_1,\hat{\sigma}_2,\ldots,\hat{\sigma}_n)$
// update the order of sequence
17: **end if**
18: $k=k+1$ and $L_k=\sum_{j=1}^{n}F(\boldsymbol{y}_{\sigma_j},\boldsymbol{y}_{\sigma_{j+1}})$
// update the values of k and L_k
19: **end while**
20: **Output:** $(\boldsymbol{y}_{\sigma_1},\boldsymbol{y}_{\sigma_2},\ldots,\boldsymbol{y}_{\sigma_n})=(\boldsymbol{y}_{\sigma_1}^k,\boldsymbol{y}_{\sigma_2}^k,\ldots,\boldsymbol{y}_{\sigma_n}^k)$
// the optimal sequence and configurations in target
// regions

It can be seen that the computational burdens of DA come mainly from steps 8 and 12 in Algorithm 1. In fact, steps 8 and 12 require to solve the following subproblem:

$$\mathcal{P}:\ \boldsymbol{y}=\underset{\boldsymbol{\eta}\in\mathcal{C}^r(\boldsymbol{z})\cup[0,2\pi]}{\arg\min}\big\{F(\boldsymbol{y}_a,\boldsymbol{\eta})+F(\boldsymbol{\eta},\boldsymbol{y}_b)\big\},\quad(4)$$

where $\boldsymbol{y}_a$ and $\boldsymbol{y}_b$ are two configurations and $\mathcal{C}^r(\boldsymbol{z})$ is a circular region centered at $\boldsymbol{z}\in\mathbb{R}^2$ with a radius $r>0$. Therefore, once the subproblem $\mathcal{P}$ can be efficiently solved, then DA will be efficient.

In the next section, the convergence of DA will be established. By proposing an efficient method to solve the subproblem $\mathcal{P}$ in Eq. (4), the efficiency of DA will be guaranteed.

4 Convergence and efficiency

In this section, the convergence and efficiency of the DA in Algorithm 1 will be established.

4.1 Convergence of the descent algorithm

We shall show the following lemma and corollary that any sequence $(L_k)_{k\in\mathbb{N}}$ generated by DA converges to a cluster point in a finite number of while-loop iterations:

Lemma 1 Any sequence $(L_k)_{k\in\mathbb{N}}$ generated by DA is monotonically non-increasing, i.e.,

$$L_{k+1}\le L_k$$

for every $k\in\mathbb{N}$.

Proof Let $[\boldsymbol{y}_{\sigma_1}^k,\boldsymbol{y}_{\sigma_2}^k,\ldots,\boldsymbol{y}_{\sigma_n}^k]_{k\in\mathbb{N}}$ be the sequence of configurations generated by DA at the end of the while loop in Algorithm 1. Fixing an arbitrary $k\in\mathbb{N}$, we have

$$\begin{aligned}L_k&=\sum_{j=1}^{n-1}F[\boldsymbol{y}_{\sigma_j}^k,\boldsymbol{y}_{\sigma_{j+1}}^k]\\&\overset{(a)}{\ge}\sum_{\substack{i=1:\text{int}(n/2),\\ j=2i-1}}\Big\{F[\boldsymbol{y}_{\sigma_j}^k,\boldsymbol{y}_{\sigma_{j+1}}^{k+1}]+F[\boldsymbol{y}_{\sigma_{j+1}}^{k+1},\boldsymbol{y}_{\sigma_{j+2}}^k]\Big\}\\&\overset{(b)}{\ge}\sum_{j=1}^{n-1}F[\boldsymbol{y}_{\sigma_j}^{k+1},\boldsymbol{y}_{\sigma_{j+1}}^{k+1}]\\&=L_{k+1},\end{aligned}\quad(5)$$

where inequalities (a) and (b) are obtained by steps 8 and 12 in Algorithm 1, respectively. Since k takes arbitrary values in $\mathbb{N}$, it follows that inequality (5) holds for any $k\in\mathbb{N}$.

Thanks to the monotonically non-increasing property established in Lemma 1, any solution to DTSPN approximated by existing algorithms, such as AA (Savla et al., 2005), SVA (Rathinam et al., 2007), and LAA (Ma and Castanon, 2006), can be improved further by DA. By the following corollary, it is shown that the solution generated by the DA in Algorithm 1 converges to a cluster point of DTSPN in a finite number of iterations:

Corollary 1 Let $(L_k)_{k\in\mathbb{N}}$ be a sequence generated by the DA in Algorithm 1. Given any $\varepsilon > 0$, there exists a finite positive integer $N \in \mathbb{N}$ such that

$$|L_N - L_{N+1}| \leq \varepsilon. \tag{6}$$

Proof Note that the sequence $(L_i)_{k\in\mathbb{N}}$ generated by DA is monotonically non-increasing according to Lemma 1. Then, considering the fact that the sequence $(L_i)_{k\in\mathbb{N}}$ is bounded, the sequence $(L_i)_{k\in\mathbb{N}}$ converges to a cluster point according to the monotone convergence theorem in Yeh (2006).

Therefore, DA is deterministically convergent to a cluster point according to Lemma 1 and Corollary 1. Thus, another issue arises; i.e., how efficient DA is when solving DTSPN. In Section 4.2, the efficiency of DA will be ensured by proposing an efficient method to solve the subproblem $\mathcal{P}$, which is required in steps 8 and 12 in Algorithm 1.

4.2 Efficiency of the descent algorithm

It is clear that solving the subproblem $\mathcal{P}$ in Eq. (4) is equivalent to finding the shortest Dubins path from a fixed initial configuration $\boldsymbol{y}_a$ to a fixed final configuration $\boldsymbol{y}_b$ via an intermediate circular region $\mathcal{C}^r(\boldsymbol{z})$. In this subsection, an efficient method for solving the subproblem $\mathcal{P}$ is presented.

Denote by C a circular arc and S a straight line segment. According to Dubins (1957), the solution paths between two configurations are smooth concatenations of C and S, and their patterns belong to two families:

$$\text{CSC} = \{\text{RSR}, \text{RSL}, \text{LSR}, \text{LSL}\},$$
$$\text{CCC} = \{\text{RLR}, \text{LRL}\},$$

where R (resp. L) denotes that the circular arc has a right (resp. left) turning direction. For example, if the pattern of a solution path is CSC, the solution path is a circular arc followed by a straight line segment and is ended by another circular arc, as shown in Fig. 1. By Assumption 1, the solution path will be the type of CSC. In such a case, given any two configurations $\boldsymbol{y}_a$ and $\boldsymbol{y}_b$ and a circular region $\mathcal{C}^r(\boldsymbol{z})$, there are two cases for the relationship between the circular region $\mathcal{C}^r(\boldsymbol{z})$ and the shortest Dubins path from $\boldsymbol{y}_a$ to $\boldsymbol{y}_b$. Fig. 1 shows two cases; Fig. 1a shows that the shortest Dubins path from $\boldsymbol{y}_a$ to $\boldsymbol{y}_b$ intersects with the intermediate circle, and Fig. 1b shows that the shortest Dubins path from $\boldsymbol{y}_a$ to $\boldsymbol{y}_b$ does not intersect with the intermediate circle. Depending on these two cases, the solution to subproblem $\mathcal{P}$ will be treated separately in the two subsections.

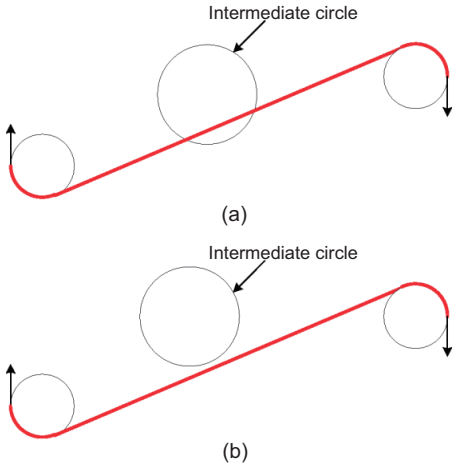

Fig. 1 Two cases for the relationship between the shortest Dubins path and the intermediate circular region: (a) solution path intersecting with the intermediate circle; (b) solution path not intersecting with the intermediate circle

4.2.1 Solution to $\mathcal{P}$ for the intersection case

When the shortest Dubins path from $\boldsymbol{y}_a$ to $\boldsymbol{y}_b$ intersects with the intermediate circular region $\mathcal{C}^r(\boldsymbol{z})$, the solution to subproblem $\mathcal{P}$ is in the overlap between the intermediate circular region and straight line of the Dubins path (Fig. 1a). In this case, any point on the overlapping segment can be considered as the solution to the subproblem $\mathcal{P}$. For simplification, we choose the middle point of the overlapping segment as the solution to $\mathcal{P}$. It is clear that the heading orientation angle is aligned to the straight line segment. Because the shortest Dubins path between any two configurations can be obtained in a constant time by checking at most four candidate paths in CSC, the solution to $\mathcal{P}$ for the intersection case in Fig. 1a can be analytically found by simple geometric analysis.

4.2.2 Solution to $\mathcal{P}$ for the non-intersection case

When the shortest Dubins path from $\boldsymbol{y}_a$ to $\boldsymbol{y}_b$ does not intersect with the intermediate circular region $\mathcal{C}^r(\boldsymbol{z})$, then the solution path of $\mathcal{P}$ has only one common tangent point within the circular region (Váňa and Faigl, 2015; Bhargav et al., 2019). By Assumption 1, it is known that the solution paths of $\mathcal{P}$

before and after the tangent point both take a form of CSC (Fig. 2).

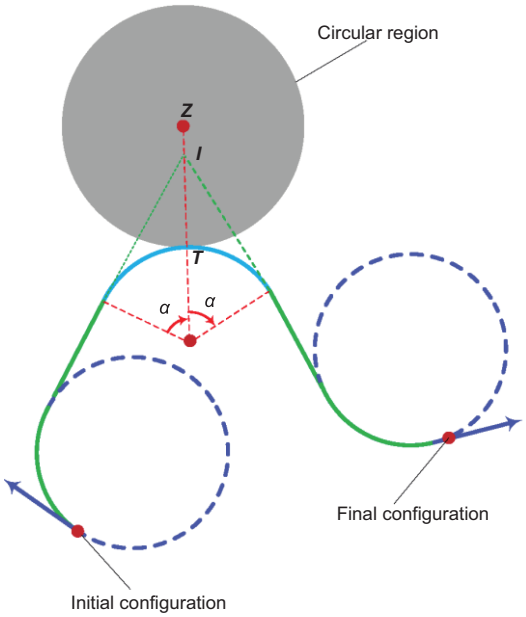

Fig. 2 Geometry of the solution path of the subproblem

For simplification, denote $C_1S_2C_3$ and $C_4S_5C_6$ as the solution paths before and after the tangent point, respectively. It has been shown in Váňa and Faigl (2015) and Bhargav et al. (2019) that the arc lengths of C_3 and C_4 are the same. Furthermore, it has been established in Bhargav et al. (2019) that the intersecting point (i.e., point $\boldsymbol{I}$ in Fig. 2) of S_2 and S_5 is collinear with the center $\boldsymbol{z}$ of the circular region and the tangent point (i.e., point $\boldsymbol{T}$ in Fig. 2). These geometric properties will be used to find the solution to $\mathcal{P}$.

Given any $\theta \in \mathbb{S}$, we have that the point

$$\boldsymbol{T}(\theta) \triangleq \boldsymbol{z} + [r\cos(\theta - \pi/2), r\sin(\theta - \pi/2)] \tag{7}$$

is tangent to the intermediate circular region $\mathcal{C}^r(\boldsymbol{z})$. Denote by $\bar{C}_1\bar{S}_2\bar{C}_3$ and $\bar{C}_4\bar{S}_5\bar{C}_6$ the shortest Dubins paths from $\boldsymbol{y}_a$ to $[\boldsymbol{T}(\theta), \theta]$ and from $[\boldsymbol{T}(\theta), \theta]$ to $\boldsymbol{y}_b$, respectively. Let $\boldsymbol{I}(\theta)$ be the intersecting point of $\bar{S}_2$ and $\bar{S}_5$. Define function

$$F(\theta) \triangleq [\cos\theta, \sin\theta][\boldsymbol{I}(\theta) - \boldsymbol{T}(\theta)]^{\mathrm{T}} \tag{8}$$

as the inner product of the unit vector of $[\cos\theta, \sin\theta]$ and vector $\boldsymbol{I}(\theta) - \boldsymbol{T}(\theta)$. Then, as shown in Fig. 3, we have

$$\begin{cases} F(\theta) > 0, & \theta > \theta^*, \\ F(\theta) = 0, & \theta = \theta^*, \\ F(\theta) < 0, & \theta < \theta^*, \end{cases} \tag{9}$$

where $\theta^* \in \mathbb{S}$ denotes the heading angle at the tangent point along the solution path of subproblem $\mathcal{P}$.

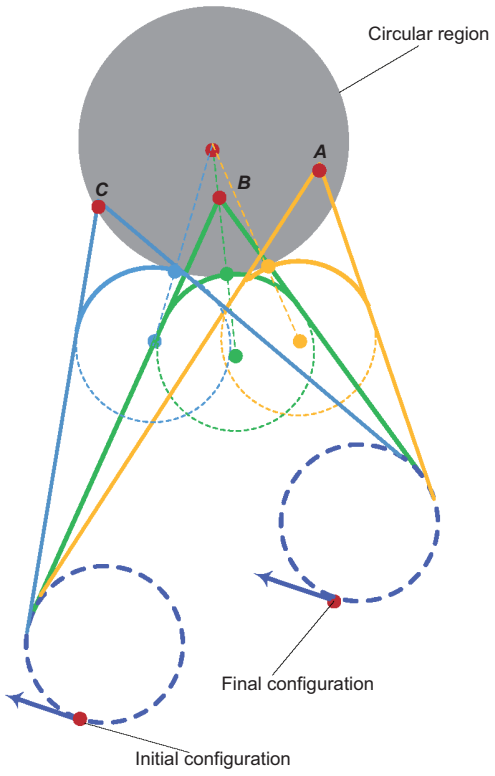

Fig. 3 Geometry of the Dubins paths

The result established in Eq. (9) allows to use a simple bisection method to accurately and efficiently find the optimal heading angle θ^* at the tangent point. Note that the shortest Dubins paths from $\boldsymbol{y}_a$ to $(\boldsymbol{T}(\theta^*), \theta^*)$ and from $(\boldsymbol{T}(\theta^*), \theta^*)$ to $\boldsymbol{y}_b$ can be efficiently obtained by checking four candidate paths in CSC. Thus, we can compute the solution to $\mathcal{P}$ efficiently. By embedding this efficient bisection method into Algorithm 1, the efficiency of DA can be significantly improved in comparison with the algorithm proposed by Váňa and Faigl (2015), where the subproblem P is solved by an approximation method.

5 Numerical simulations

In this section, a large number of numerical simulations are presented to demonstrate the performance of DA (Algorithm 1) in comparison with those of several existing algorithms, such as AA, SVA, and LAA. As shown in Váňa and Faigl (2015), evolutionary algorithms are computationally demanding when applied to address DTSPN. Thus, we shall not compare the performance of DA with those of evolutionary algorithms.

5.1 Bisection method for solving P

According to Algorithm 1, the computational cost of DA is determined mainly by the time needed to solve each subproblem $\mathcal{P}$. The bisection method developed in Section 4.2 can solve each subproblem $\mathcal{P}$ efficiently and accurately. In this subsection, the time to solve subproblem $\mathcal{P}$ by the bisection method is obtained through a large number of simulations. Four examples selected from the simulated subproblems are presented in Fig. 4.

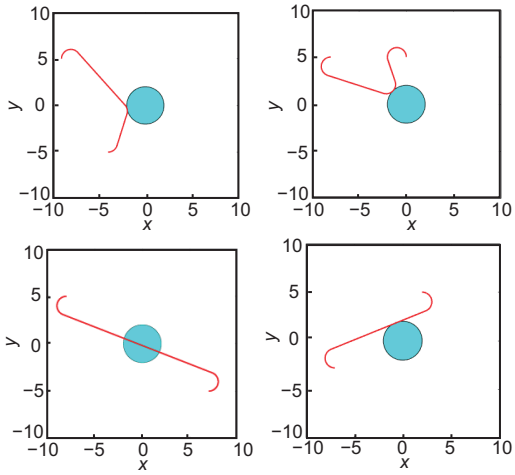

Fig. 4 Solution paths of subproblem P for different geometries

Numerical simulation results show that, using Matlab, solving subproblem $\mathcal{P}$ in Eq. (4) averagely requires around $t_p \triangleq 2.49 \times 10^{-5}$ s on a desktop with Intel® Core™ i7-8550U CPU@1.80 GHz. Denote by $n > 0$ the number of targets. Then, the while loop in Algorithm 1 requires around $n \cdot t_p$. Thus, if the number of targets is 30, a while loop in Algorithm 1 requires just around 7.47×10^{-4} s on the same desktop. In the next subsection, the performance of DA will be presented.

5.2 Descent algorithm for DTSPN

We use a uniform distribution to randomly generate 30 circular regions (the radius of each region is 4, i.e., $r = 4$) within the square $[0, 100] \times [0, 100]$ (distance unit is meter). Considering the center of each region as a target, we use the algorithm in Tran (2019) to order the sequence (Fig. 5a), which actually gives the solution to ETSP.

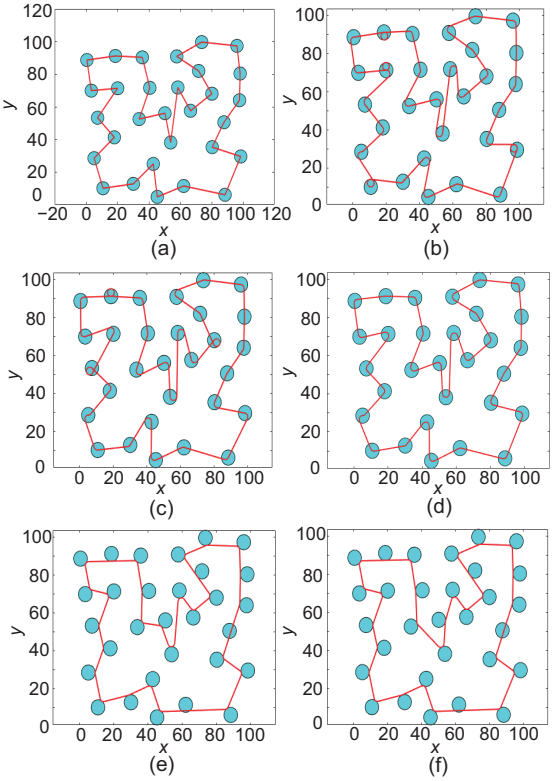

Fig. 5 Paths of DTSPN generated by ETSP (a), AA (b), SVA (c), LAA (d), DA (e), and DA with a reordered sequence (f)

From the ordered sequence of ETSP, AA (Savla et al., 2005), SVA (Rathinam et al., 2007), LAA (Ma and Castanon, 2006), and the developed DA are tested on the randomly generated DTSPN. Note that DA requires to guess a configuration in each target region. In this simulation, we choose the solution to LAA as the initial guess of DA. The lengths of the solution paths generated by these algorithms are presented in Table 1, from which we can see that the performance of DA maintains its superiority in comparison with those of AA, SVA, and LAA. We can see that DA reordering the sequence of target regions can generate a slightly better solution than that not reordering the sequence.

Table 1 Lengths of DTSPN's solution paths generated by AA, SVA, LAA, and DA

Algorithm	Length
AA	665.45
SVA	621.12
LAA	590.76
DA*	458.2
DA#	454.99

* indicates that the order of sequence is not optimized.
# indicates that the order of sequence is optimized

Note that reordering the sequence in Algorithm 1 involves randomly inserting one target region into a sequence which requires to solve a subproblem $\mathcal{P}$. Thanks to the bisection method that can efficiently and accurately solve the subproblem $\mathcal{P}$ in Eq. (4), reordering the sequence in Algorithm 1 does not distinctly influence the total computational cost.

The solution paths generated by AA, SVA, LAA, and DA are presented in Fig. 5. Fig. 5 clearly shows that the paths generated by AA, SVA, and LAA must pass through the center of each region. Unlike the three existing algorithms, the visiting point of DA in each region is optimized instead of being fixed, which leads to a better-quality solution (Table 1).

By a large number of simulations with 30 regions randomly generated by uniform distribution, the DA coded in Matlab is tested, showing that it averagely takes 0.014 s on a desktop with Intel® Core™ i7-8550U CPU@1.80 GHz to generate the solution to DTSPN with 30 target regions. In fact, the computational time can be further improved if DA is coded by C++ or other more efficient programming languages. In conclusion, the development of DA can solve DTSPN in an efficient way.

6 Conclusions

Aiming at addressing DTSPN in an efficient way, we have proposed a gradient-free descent algorithm. The core idea of the algorithm is to decompose DTSPN into a series of subproblems, each of which consists of finding the shortest path from a configuration to another configuration via an intermediate circular region. Considering that the computational cost of the descent algorithm is determined mainly by the time to solve each subproblem, a simple bisection method has been established to efficiently and accurately address the subproblems. Numerical simulation results showed that the subproblems can be solved within several milliseconds. This allows the descent method to address DTSPN in an efficient way even if the number of targets is up to 30. In addition, a larger number of simulations have been conducted to show that any approximation solution obtained by existing methods, such as AA, SVA, and LAA, can be further improved by the proposed algorithm.

Contributors

Zheng CHEN designed the research. Zheng CHEN and Chen-hao SUN simulated the numerical examples. Zheng CHEN and Xue-ming SHAO drafted the manuscript. Wen-jie ZHAO helped organize the manuscript. Chen-hao SUN revised the manuscript. Zheng CHEN and Xue-ming SHAO revised and finalized the paper.

Compliance with ethics guidelines

Zheng CHEN, Chen-hao SUN, Xue-ming SHAO, and Wen-jie ZHAO declear that they have no conflict of interest.

References

Arora S, 1998. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. *J ACM*, 45(5):753-782. https://doi.org/10.1145/290179.290180

Bhargav J, Chen Z, Shima T, 2019. Shortest bounded-curvature paths via circumferential envelope of a circle. IFAC Conf.

Chen Z, Shima T, 2019. Shortest Dubins paths through three points. *Automatica*, 105:368-375. https://doi.org/10.1016/j.automatica.2019.04.007

Dubins LE, 1957. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *Am J Math*, 79(3):497-516. https://doi.org/10.2307/2372560

Faigl J, Váňa P, 2018. Surveillance planning with Bézier curves. *IEEE Rob Autom Lett*, 3(2):750-757. https://doi.org/10.1109/LRA.2018.2789844

Goaoc X, Kim HS, Lazard S, 2013. Bounded-curvature shortest paths through a sequence of points using convex optimization. *SIAM J Comput*, 42(2):662-684. https://doi.org/10.1137/100816079

Isaacs JT, Hespanha JP, 2013. Dubins traveling salesman problem with neighborhoods: a graph-based approach. *Algorithms*, 6(1):84-99. https://doi.org/10.3390/a6010084

Isaacs JT, Klein DJ, Hespanha JP, 2011. Algorithms for the traveling salesman problem with neighborhoods involving a Dubins vehicle. Proc American Control Conf, p.1704-1709. https://doi.org/10.1109/ACC.2011.5991501

Isaiah P, Shima T, 2015. Motion planning algorithms for the Dubins travelling salesperson problem. *Automatica*, 53:247-255. https://doi.org/10.1016/j.automatica.2014.12.041

Lawler EL, Lenstra JK, Kan AHGR, et al., 1985. The Traveling Salesman Problem: a Guided Tour of Combinatorial Optimization. Wiley, New York, USA.

Ma X, Castanon DA, 2006. Receding horizon planning for Dubins traveling salesman problems. Proc 45th IEEE Conf on Decision and Control, p.5453-5458. https://doi.org/10.1109/CDC.2006.376928

Macharet DG, Alves Neto A, da Camara Neto VF, et al., 2012. An evolutionary approach for the Dubins' traveling salesman problem with neighborhoods. Proc 14th Annual Conf on Genetic and Evolutionary Computation, p.377-384. https://doi.org/10.1145/2330163.2330218

Ny JL, Feron E, Frazzoli E, 2012. On the Dubins traveling salesman problem. *IEEE Trans Autom Contr*, 57(1):265-270. https://doi.org/10.1109/TAC.2011.2166311

Obermeyer KJ, 2009. Path planning for a UAV performing reconnaissance of static ground targets in terrain. Proc AIAA Guidance, Navigation, and Control Conf, p.10-13. https://doi.org/10.2514/6.2009-5888

Obermeyer KJ, Oberlin P, Darbha S, 2010. Sampling-based roadmap methods for a visual reconnaissance UAV. Proc AIAA Guidance, Navigation, and Control Conf, Article 21. https://doi.org/10.2514/6.2010-7568

Rathinam S, Sengupta R, Darbha S, 2007. A resource allocation algorithm for multivehicle systems with nonholonomic constraints. *IEEE Trans Autom Sci Eng*, 4(1):98-104. https://doi.org/10.1109/TASE.2006.872110

Sadeghi A, Smith SL, 2016. On efficient computation of shortest Dubins paths through three consecutive points. Proc IEEE 55th Conf on Decision and Control, p.6010-6015. https://doi.org/10.1109/CDC.2016.7799192

Savla K, Frazzoli E, Bullo F, 2005. On the point-to-point and traveling salesperson problems for Dubins' vehicle. Proc American Control Conf, p.786-791. https://doi.org/10.1109/ACC.2005.1470055

Tran DC, 2019. Tsp.zip. www.mathworks.com/matlabcentral/fileexchange/46629-tsp-zip

Tuqan M, Daher N, Shammas E, 2019. A simplified path planning algorithm for surveillance missions of unmanned aerial vehicles. Proc IEEE/ASME Int Conf on Advanced Intelligent Mechatronics, p.1341-1346. https://doi.org/10.1109/AIM.2019.8868551

Váňa P, Faigl J, 2015. On the Dubins traveling salesman problem with neighborhoods. Proc IEEE/RSJ Int Conf on Intelligent Robots and Systems, p.4029-4034. https://doi.org/10.1109/IROS.2015.7353945

Xin B, Chen J, Xu D, et al., 2014. Hybrid encoding based differential evolution algorithms for Dubins traveling salesman problem with neighborhood. *Contr Theory Appl*, 31(7):941-954. https://doi.org/10.7641/CTA.2014.40280

Yeh J, 2006. Real Analysis: Theory of Measure and Integration. World Scientific, Hackensack, New Jersey, USA.

Yu X, Hung JY, 2012. Optimal path planning for an autonomous robot-trailer system. Proc 38th Annual Conf on IEEE Industrial Electronics Society, p.2762-2767. https://doi.org/10.1109/IECON.2012.6389140

Zhang X, Chen J, Xin B, et al., 2014. A memetic algorithm for path planning of curvature-constrained UAVs performing surveillance of multiple ground targets. *Chin J Aeron*, 27(3):622-633. https://doi.org/10.1016/j.cja.2014.04.024

Zheng CHEN received his PhD degree in applied mathematics from the Université Paris-Saclay in 2016, and his BS and MS degrees in aerospace engineering from the Northwestern Polytechnical University in 2010 and 2013, respectively. He is currently a researcher at the School of Aeronautics and Astronautics at Zhejiang University. His current research interests include guidance and control in aerospace engineering.