Surveillance Planning with Bézier Curves

Jan Faigl and Petr Váňa

***Abstract*—This paper concerns surveillance planning for an Unmanned Aerial Vehicle (UAV) that is requested to periodically take snapshots of areas of interest by visiting a given set of waypoint locations in the shortest time possible. The studied problem can be considered as a variant of the combinatorial traveling salesman problem in which trajectories between the waypoints respect the kinematic constraints of the UAV. Contrary to the existing formulation for curvature-constrained vehicles known as the Dubins traveling salesman problem, the herein addressed problem is motivated by planning for multi-rotor UAVs which are not limited by the minimal required forward velocity and minimal turning radius as the Dubins vehicle, but rather by the maximal speed and acceleration. Moreover, the waypoints to be visited can be at different altitudes, and the addressed problem is to find a fast and smooth trajectory in 3D space from which all the areas of interest can be captured. The proposed solution is based on unsupervised learning in which the requested 3D smooth trajectory is determined as a sequence of Bézier curves in a finite number of learning epochs. The reported results support feasibility of the proposed solution which has also been experimentally verified with a real UAV.**

***Index Terms*—Motion and Path Planning, Nonholonomic Motion Planning, Aerial Systems: Applications**

I. Introduction

In this paper, we study surveillance planning for an Unmanned Aerial Vehicle (UAV) that is requested to periodically take snapshots of objects of interest by visiting a given set of sensing locations at different altitudes such that, the total required time to visit all the locations is minimal. The addressed problem can be considered as a variant of the combinatorial the Traveling Salesman Problem (TSP) in which connections between the waypoints respect the kinematic constraints of the UAV. Contrary to the existing formulation for curvature-constrained vehicles known as the Dubins Traveling Salesman Problem (DTSP) [1], the herein addressed problem is motivated by planning for multi-rotor UAVs which are not limited by a minimal turning radius as the Dubins vehicle (or any other requirements on the minimal forward velocity to keep the vehicle in the air such as fixed-wing aircrafts), but rather by the maximal speed and acceleration.

Similarly to the DTSP with Neighborhoods (DTSPN) [2], [3], also in the addressed problem, it is allowed to visit a neighborhood of the particular sensing location, and thus save the travel time by avoiding a precise visitation of the prescribed sensing locations. This generalization is arising from the surveillance missions where a downward looking camera is utilized to take a snapshot of the objects of interest, and relatively small objects can be captured with the desired level of details within the camera field of view. Hence it is sufficient to take a snapshot within a specific sensing distance δ from the particular sensing location. Therefore, the prescribed sensing locations have δ-neighborhood, and it is sufficient that the final trajectory is passing the δ-neighborhood of each sensing location, e.g., see trajectory in Fig. 1.

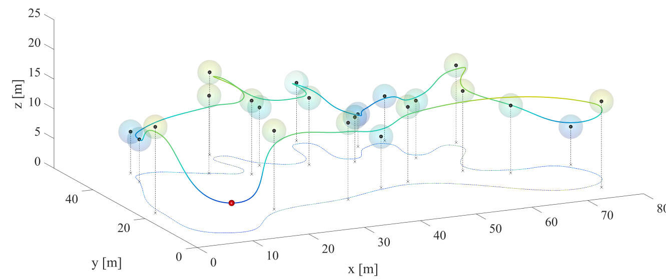


Fig. 1. An example of the 3D smooth trajectory visiting δ-neighborhood of the given set of sensing locations. The trajectory is found by the proposed unsupervised learning based algorithm as a sequence of Bézier curves minimizing the TTE of the whole trajectory.

In [4], it has been shown that Dubins tour visiting a sequence of waypoints can be advantageous for multi-rotor UAVs to achieve a precise navigation to the desired locations rather than navigation along straight line segments. However, a multi-rotor UAV can accelerate on straight lines and can make turns at small turning radius with lower speed. Therefore, it might be more suitable to abandon curvature-constrained trajectory based on Dubins maneuvers and use a different smooth trajectory parameterization that can take an advantage of the trajectory curvature with the maximal vehicle velocity and acceleration limits. Then, a vehicle velocity profile can be computed and the expected time to travel the trajectory, further denoted as the Travel Time Estimation (TTE), can be determined and minimized during the surveillance planning.

Even though the DTSP and DTSPN have been addressed by many approaches including approximation algorithms such as [5], [6], [7], heuristics [1], [8], [9], [10], [11], and also evolutionary techniques [12], [13], [14], to the best of our knowledge, the TSP-like formulation of the surveillance planning with 3D smooth trajectory parametrization has not yet been directly addressed. A possible 3D surveillance planning can follow a decoupled approach like the Alternating Algorithm (AA) [1] or Local Iterative Optimization (LIO) [8] in which the sequence of visits to the waypoint locations is determined as the Euclidean TSP (ETSP), and then a smooth trajectory connecting the locations is computed. Hence various types of curves such as B-Splines [15], polynomial functions [16] or Bézier curves [17] can be utilized for generation of continuous

Manuscript received: September, 10, 2017; Revised November, 29, 2017; Accepted December, 22, 2017.

This paper was recommended for publication by Editor Nancy Amato upon evaluation of the Associate Editor and Reviewers' comments. The presented work has been supported by the Czech Science Foundation (GAČR) under research project No. 16-24206S.

Authors are with the Czech Technical University, Faculty of Electrical Engineering, Technicka 2, 166 27, Prague, Czech Republic `{faiglj|vanapet1}@fel.cvut.cz`

Digital Object Identifier (DOI): see top of this page.

and smooth trajectory [18]. However, such a sequence can be of poor quality because it is determined independently to the constraints of the UAV and the δ-neighborhood.

Therefore, we propose to leverage on advancements on the recent unsupervised learning of the Self-Organizing Map (SOM) in a solution of the DTSP [10] that has been extended to the DTSPN in [11], and we propose to directly employ a smooth curvature and the TTE optimization during the solution of the sequencing part of the addressed surveillance problem. A determination of 3D smooth trajectories during the optimization is likely to be more demanding than a closed-form solution of the optimal Dubins maneuvers [19]. On the other hand, the results presented in [11] show that a solution of problems with tens of waypoints is found in hundreds of milliseconds, which further motivate us to consider the unsupervised learning with a more general trajectory parameterization than Dubins maneuvers. The selected trajectory parameterization is based on Bézier curves because it is specified by only four control points and it allows to find smooth trajectories not only in 2D but also in 3D scenarios. Therefore, the contribution of the presented work is the novel unsupervised learning procedure for planning 3D smooth trajectories parameterized as a sequence of Bézier curves visiting δ-neighborhood of the given sensing locations placed at different altitudes.

The remainder of the paper is organized as follows. The addressed problem is formally introduced in the next section together with the necessary background on the Bézier curve parametrization and computation of the velocity profile and thus the TTE. The proposed SOM-based solution is described in Section III. Results on a comparison of the proposed approach with the selected DTSP approaches in 2D missions, an influence of the increasing δ to the solution quality, and evaluation of altitudes differences in 3D missions are presented in Section IV together with the report on the experimental deployment of the proposed method with a real UAV. Concluding remarks are dedicated to Section V.

II. Problem Statement

In the herein addressed surveillance planning, we consider a smooth trajectory to capture a given set of n objects of interest $\mathcal{O}$ by traveling along the trajectory that satisfies the limitations of the used UAV. For each $o_i \in \mathcal{O}$, we consider a sensing location $\boldsymbol{s}_i \in \mathbb{R}^3$ from which a snapshot of o_i can be captured. Moreover, o_i can be captured within the δ distance from $\boldsymbol{s}_i$, and thus for each $o_i \in \mathcal{O}$, the surveillance trajectory has to contain a waypoint location $\boldsymbol{p}_i$ such that $\|(\boldsymbol{p}_i, \boldsymbol{s}_i)\| \leq \delta$. The problem is to determine a fastest possible trajectory to visit the δ-neighborhood of all the sensing locations, which consists of determining the sequence of visits to the neighborhoods, i.e., a variant of the TSP, but also the trajectory optimization.

The considered trajectory parameterization is based on the Bézier curve that is defined by four control points. The end locations of the curve are directly defined by two control points, and the departure and terminal tangents of the curve are defined by the two additional points. This allows to connect a sequence of Bézier curves into a smooth path, and thus we consider n Bézier curves connected as the final trajectory, i.e., one curve for one object of interest. The expanded form of the utilized parametrization of the Bézier curve $\mathbf{X}(\tau)$ [20] can be expressed as

$$\mathbf{X}(\tau) = \mathbf{B}_0(1-\tau)^3 + 3\mathbf{B}_1\tau(1-\tau)^2 + 3\mathbf{B}_2\tau^2(1-\tau) + \mathbf{B}_3\tau^3, \tag{1}$$

where $0 \leq \tau \leq 1$ and $\mathbf{B_k}$ stands for the k-th control point.

Since the final trajectory $\mathcal{X}$ is closed, it consists of n Bézier curves $\mathbf{X}_i$ and because it has to be smooth, two consecutive curves $\mathbf{X}_i$ and $\mathbf{X}_j$ with the control points ($\mathbf{B}_0^i$, $\mathbf{B}_1^i$, $\mathbf{B}_2^i$, $\mathbf{B}_3^i$) and ($\mathbf{B}_0^j$, $\mathbf{B}_1^j$, $\mathbf{B}_2^j$, $\mathbf{B}_3^j$) have to be connected at the same end point. Thus, the last control point $\mathbf{B}_3^i$ and the first control point $\mathbf{B}_0^j$ need to be identical to keep the trajectory continuous

$$\mathbf{B}_3^i = \mathbf{B}_0^j. \tag{2}$$

In addition, the Bézier curves have to point to the same direction, and therefore, the tangents can be defined as

$$\mathbf{t}_a^i = \mathbf{B}_1^i - \mathbf{B}_0^i, \quad \mathbf{t}_b^i = \mathbf{B}_3^i - \mathbf{B}_2^i \tag{3}$$

with the length of the particular tangent vector

$$l_a^i = \left\|\mathbf{t}_a^i\right\|, \quad l_b^i = \left\|\mathbf{t}_b^i\right\|. \tag{4}$$

This requirement can be satisfied by the condition ensuring the trajectory is smooth

$$l_a^j\,\mathbf{t}_b^i = l_b^i\,\mathbf{t}_a^j. \tag{5}$$

It is assumed, the multi-rotor UAV can, in general, travel an arbitrary path. Moreover, we aim to determine not the shortest path as in the DTSP with a constant forward velocity of the vehicle, but we rather aim to determine the fastest trajectory for the particular UAV limited only by its maximal velocity and acceleration. Therefore instead of minimizing the length of the path, the *Travel Time Estimation* (TTE) of the trajectory $\mathcal{T}(\mathcal{X})$ is considered in the proposed problem formulation.

Finally, we assume the UAV is requested to start its mission at the given initial location $\boldsymbol{s}_1$ (further called depot) and it is requested to return at the same location after finishing the patrolling tour, and thus visitation of $\boldsymbol{s}_1$ is considered without the δ-neighborhood, i.e., $\delta = 0$ for the depot $\boldsymbol{s}_1$.

Having the introduced preliminaries, the surveillance planning problem can be formulated as follows. For the given set of n sensing locations $S = \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n\}$, $\boldsymbol{s}_i \in \mathbb{R}^3$, one $\boldsymbol{s}_i$ for each $o_i \in \mathcal{O}$, the sensing distance δ, and the depot location $\boldsymbol{s}_1$, determine a smooth trajectory $\mathcal{X}$ as a sequence $\Sigma = (\sigma_1, \ldots, \sigma_n)$ of Bézier curves $\mathbf{X}_i$, $1 \leq i \leq n$, i.e., $\mathcal{X} = (\mathbf{X}_{\sigma_1}, \ldots, \mathbf{X}_{\sigma_n})$, and $1 \leq \sigma_i \leq n$, such that the TTE of the trajectory $\mathcal{T}(\mathcal{X})$ is minimal, $\mathbf{X}_{\sigma_1}$ starts at the depot $\boldsymbol{s}_1$, $\mathbf{X}_{\sigma_n}$ terminates at $\boldsymbol{s}_1$, and for each $\boldsymbol{s}_{\sigma_i}$, there is a point $\boldsymbol{p}_{\sigma_i}$ of the trajectory $\mathbf{X}_{\sigma_i}$ within the δ distance from $\boldsymbol{s}_{\sigma_i}$, $\left\|(\boldsymbol{p}_{\sigma_i}, \boldsymbol{s}_{\sigma_i})\right\| \leq \delta$.

Problem 2.1 (Surveillance Mission Planning Problem):

$$\begin{aligned}
&\textit{minimize}_{\ \Sigma,\mathcal{X},P} \quad \mathcal{T}(\mathcal{X}) = \sum_{i=1}^{n} \mathcal{T}(\mathbf{X}_{\sigma_i})\\
&\textit{subject to}\\
&\quad \Sigma = (\sigma_1, \ldots, \sigma_n),\ 1 \leq \sigma_i \leq n, \sigma_i \neq \sigma_j \text{ for } i \neq j;\\
&\quad \mathcal{X} = (\mathbf{X}_{\sigma_1}, \ldots, \mathbf{X}_{\sigma_n}) \text{ is a smooth connection of } n\\
&\quad \text{Bézier curves where } \mathbf{X}_{\sigma_1} \text{ starts at } \boldsymbol{s}_1,\ \mathbf{X}_{\sigma_n} \text{ ends at}\\
&\quad \boldsymbol{s}_1, \text{ and each } \mathbf{X}_{\sigma_i} \text{ has a point } \boldsymbol{p}_{\sigma_i} \in \mathbf{X}_{\sigma_i} \text{ such that}\\
&\quad P = (\boldsymbol{p}_{\sigma_1}, \ldots, \boldsymbol{p}_{\sigma_n}),\ \ \left\|(\boldsymbol{p}_{\sigma_i}, \boldsymbol{s}_{\sigma_i})\right\| \leq \delta \text{ for } \boldsymbol{s}_{\sigma_i} \in S.
\end{aligned} \tag{6}$$

A. Travel Time Estimation (TTE)

The Travel Time Estimation (TTE) can be determined from the velocity profile along a parameterized trajectory such as the Bézier curve (1). A simplified model of the velocity profile is considered, and the vertical and horizontal movements of the vehicle are individually limited by the maximal velocity and acceleration. Although the motion of the multi-rotor UAV is generally coupled, we are allowed to consider movements independent because of the utilized Model Predictive Controller (MPC) [21] employed for the trajectory following. Decoupling the axes and motion generation for each axis separately [22] allows efficient computation of the MPC [21]. Therefore, the employed implementation of [21] provides real-time computation of the required thrust of individual rotors while it still guarantees the vehicle follows the planned trajectory for the vertical and horizontal accelerations under the respective limits together with the requested horizontal and vertical velocities. We denote v_{vert} to the maximal vertical speed, a_{vert} to the corresponding maximal magnitude of the vehicle acceleration and similarly v_{horiz} and a_{horiz} for the horizontal speed and acceleration, respectively. The TTE is the minimal expected time to travel the given trajectory, and thus the maximal velocity profile along the trajectory is determined with respect to the path curvature and the acceleration limits.

The profile for the vertical velocity is directly computed from the altitude differences along the curve, and thus the first and second derivatives along the $z-$axis are utilized, and the magnitude of the vertical velocity is limited by v_{vert} and a_{vert}. On the other hand, two acceleration components that affect the vehicle simultaneously can be identified in the horizontal plane: the tangent a_{tan} and radial a_{rad} accelerations. The velocity change is caused by the tangent acceleration a_{tan}. The radial acceleration a_{rad} is defined by the path curvature, and it does not directly affect the vehicle speed. The tangent and radial accelerations are always perpendicular and the combined value cannot exceed a_{horiz} such that $a_{tan}^2 + a_{rad}^2 \leq a_{horiz}^2$. The path curvature defines a_{rad} as $a_{rad} = v^2\,\kappa_h$, where κ_h is the horizontal curvature of the trajectory determined as

$$\kappa_h = \frac{\|x'y'' - y'x''\|}{(x'^2 + y'^2)^{\frac{3}{2}}}, \tag{7}$$

and for Bézier curves it can be expressed in a closed-form. The cuvature also defines the maximal possible horizontal velocity v_{pos} of the vehicle along the trajectory as

$$v_{pos} = \min\left(v_{horiz}, \sqrt{\frac{a_{horiz}}{\kappa_h}}\right). \tag{8}$$

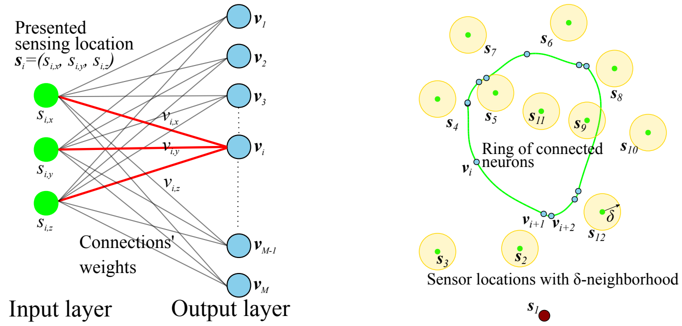
Then, the maximal possible tangent acceleration a_{tan} can be expressed as

$$\begin{array}{rcl} a_{tan}^2 & = & a_{horiz}^2 - a_{rad}^2 \\ & = & a_{horiz}^2 - v_{pos}^4 \kappa_h^2 \end{array}, \tag{9}$$

where the right side is always positive because of (8).

The velocity profile, and thus the TTE, for a sequence of curves can be determined numerically in six steps:

1) Sample the curves into finite uniformly sampled points and determine the corresponding values of the curvature (7) computed for the first and second derivatives expressed from (1).
2) Set the initial and final vehicle velocity to zero as it is assumed the vehicle starts the mission with the zero velocity from the initial location (depot) and return to it, e.g., for replacing battery.
3) Compute derivatives along the z-axis and limit the velocity according to v_{vert} and a_{vert}.
4) Determine v_{pos} for each sampled point (8).
5) Iterate over the samples forward and limit the velocity by the maximum possible tangent acceleration (9), i.e., adjust the travel time between the respective samples.
6) Iterate over the samples backward and limit the velocity by the maximum possible tangent acceleration (9).



(a) SOM structure (b) Ring of neurons

Fig. 2. SOM structure for the 3D surveillance planning and visualization of the ring during the evolution. The green disks represent sensing locations each with δ-neighborhood visualized as the yellow disk. The requested initial and terminal location without the neighborhood is called depot, and it is shown as the dark red disk $\boldsymbol{s}_1$. The neurons are blue disks organized in the output layer into a ring of neurons. The neuron's weights are waypoint locations in the input space. The neurons are further associated with parameters for Bézier curves, and thus the ring of neurons forms the desired smooth trajectory.

III. Surveillance Planning with Bézier Curves

The proposed planning approach is motivated by recent advancements on Self-Organizing Map (SOM) based solution of the DTSP [10] and DTSPN [11], which originates in growing SOM for the TSPN [23]. The used SOM is a two-layered neural network where the input layer serves for presenting the sensing locations $\boldsymbol{s}_i \in S$ and the output layer is an array of neurons $\mathcal{N} = \{\boldsymbol{\nu}_1, \ldots, \boldsymbol{\nu}_M\}$, where M is the current number of neurons in the network.

In the proposed approach, the neurons are the waypoint locations and a ring of neurons connected by the Bézier curves is the requested surveillance trajectory, which evolves in the input space $\mathbb{R}^3$ during the learning. A structure of the network and an example of Bézier curves connecting the neurons are visualized in Fig. 2. The learning is an iterative procedure that works in learning epochs, and the main principle of the learning is as follows.

A single learning epoch is an adaptation of the network towards all sensing locations, which are presented to the network in a random order to avoid local optima [24]. In the used growing SOM, the best matching neuron to the presented location $\boldsymbol{s}$ is determined as the closest point $\boldsymbol{p}_s$

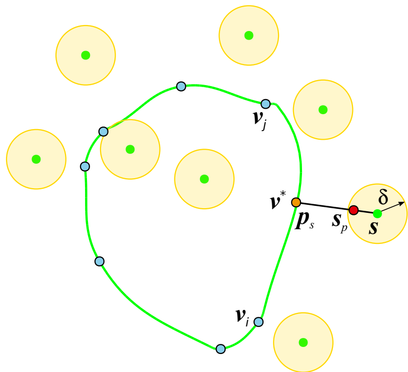


Fig. 3. Principle of determining the winner neuron to the presented sensing location $\boldsymbol{s}$. The winner neuron $\boldsymbol{\nu}^*$ is created at the location $\boldsymbol{p}_s$ which is the closest point of the sequence of Bézier curves connecting the neurons. An intersection of the straight line segment $(\boldsymbol{p}_s, \boldsymbol{s})$ with the disk (or ball in $\mathbb{R}^3$) shaped δ-neighborhood of $\boldsymbol{s}$ is used to determine the alternate location $\boldsymbol{s}_p$ towards which the network is adapted instead of $\boldsymbol{s}$ to save the travel time. The point $\boldsymbol{s}_p$ is inside the δ-neighborhood of $\boldsymbol{s}$ to ensure the waypoint location (the winner neuron) would be at the distance to $\boldsymbol{s}$ shorter than δ.

on the trajectory represented by the ring. Then, the winner neuron $\boldsymbol{\nu}^*$ is created at the location corresponding to the point $\boldsymbol{p}_s$. However, instead of an adaptation towards $\boldsymbol{s}$, the δ-neighborhood is utilized to save the travel time and a point $\boldsymbol{s}_p$ inside the neighborhood is determined, e.g., a point on the segment $(\boldsymbol{p}_s, \boldsymbol{s})$ close to the perimeter δ. Notice, if $\boldsymbol{p}_s$ is already inside δ-neighborhood, only the respective winner neuron $\boldsymbol{\nu}^*$ is created and the network is not adapted. In a case of $\delta = 0$, the winner neuron is adapted towards $\boldsymbol{s}$, i.e., $\boldsymbol{s}_p = \boldsymbol{s}$. The proposed selection of the winner neuron is visualized in Fig. 3.

The adaptation follows the standard SOM learning [25] and it can be considered as a movement of the neurons towards $\boldsymbol{s}_p$ and a new location of each adapted neuron $\boldsymbol{\nu}$ becomes $\boldsymbol{\nu}'$

$$\boldsymbol{\nu}' = \boldsymbol{\nu} + \mu f(\sigma, d)(\boldsymbol{s}_p - \boldsymbol{\nu}), \tag{10}$$

where μ is the learning rate, $f(\sigma, d)$ is the neighbouring function, σ is the learning gain, and d is the distance of $\boldsymbol{\nu}$ from the winner neuron $\boldsymbol{\nu}^*$ in the number of neurons in the ring. The neighbouring function defines active neurons around $\boldsymbol{\nu}^*$ that are adapted and it has the standard form utilized in SOM [25] adjusted for the TSP [24]

$$f(\sigma, d) = \begin{cases} e^{\frac{-d^2}{\sigma^2}} & \text{for } d < 0.2M \\ 0 & \text{otherwise} \end{cases} . \tag{11}$$

The learning gain is decreased after each learning epoch according to the cooling schedule $\sigma = (1-\alpha)\sigma$, where α is the gain decreasing rate. Besides, a ring regeneration is performed after each learning epoch to remove all non-winner neurons and keep the total number of neurons less than $2n$ during the learning.

The adaptation is terminated if all winner neurons become negligibly close to their respective $\boldsymbol{s}_p$, and thus inside the δ-neighborhood of each $\boldsymbol{s}$. Then, the sequence of Bézier curves connecting the winner neurons represents a feasible solution to the surveillance planning problem. If the network convergence is not fast enough, the learning can be explicitly terminated after i_{max} learning epochs and a feasible solution can be constructed from traversing the ring and using $\boldsymbol{s}_p$ associated with the winner neurons. [1] For the used learning rate $\mu = 0.5$, the network converges [27] and for the initial learning gain $\sigma = 12.41n + 0.6$ and the gain decreasing rate $\alpha = 0.1$ [24], the network is stabilized in less than 100 learning epochs.

Although the learning procedure follows SOM for the TSPN [23] where neurons are waypoint locations adapted towards the sensing locations, the connection of neurons by Bézier curves to minimize the TTE of the whole trajectory is the most important part of the proposed surveillance planning. The smooth connection of the sequence of Bézier curves has to satisfy (5) and because it is assumed the multi-rotor UAV can follow any 3D path, we propose to support evaluation of (5) by the vehicle position (x, y, z) expressed as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = v \begin{bmatrix} \cos\theta\cos\psi \\ \sin\theta\cos\psi \\ \sin\psi \end{bmatrix}, \tag{12}$$

where θ is the turning angle and ψ is the climb/dive angle of the trajectory at the point (x, y, z) of the path.

The waypoint locations (the neurons) and the particular angles θ and ψ are not sufficient to specify the control points of the Bézier curve connecting two neighboring neurons $\boldsymbol{\nu}_{i-1}$ and $\boldsymbol{\nu}_i$. Therefore, each neuron is further associated with the tangent vectors and each Bézier curve is defined by two tangent vectors separately. Thus, each neuron $\boldsymbol{\nu}_i$ corresponds to the waypoint location $\boldsymbol{\nu}_i \in \mathbb{R}^3$. Besides, it is also associated with angles θ_i and ψ_i, and with the lengths of two tangent vectors l_a^i and l_b^i. Each neuron $\boldsymbol{\nu}_i$ is incident with two Bézier curves, and therefore, the tangent vector for the Bézier curve $\mathbf{X}_{i-1}$ that terminates at $\boldsymbol{\nu}_i$ is

$$\mathbf{t}_b^{i-1} = l_b^i \begin{bmatrix} \cos(\theta_i)\cos(\psi_i) \\ \sin(\theta_i)\cos(\psi_i) \\ \sin(\psi_i) \end{bmatrix} \tag{13}$$

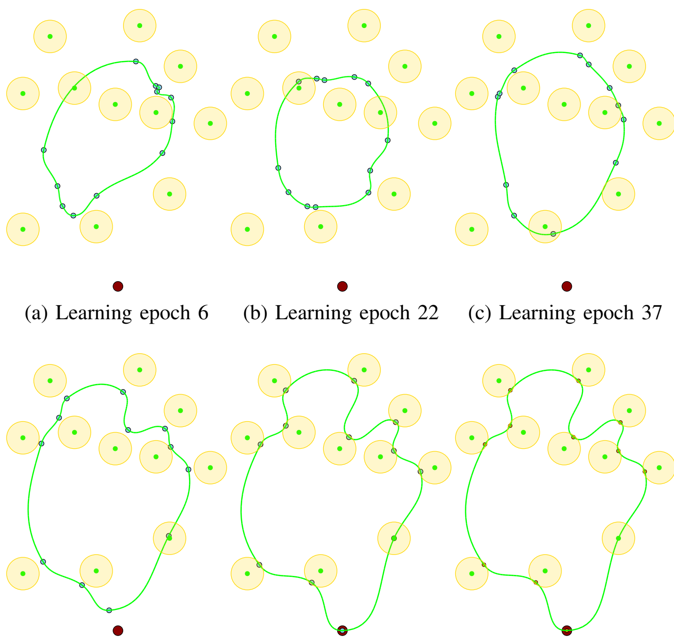
and the tangent vector $\mathbf{t}_b^i$ defines the initial part of the Bézier curve $\mathbf{X}_i$, which starts at $\boldsymbol{\nu}_i$, is determined as

$$\mathbf{t}_a^i = l_a^i \begin{bmatrix} \cos(\theta_i)\cos(\psi_i) \\ \sin(\theta_i)\cos(\psi_i) \\ \sin(\psi_i) \end{bmatrix}. \tag{14}$$

A new neuron $\boldsymbol{\nu}_i$ is inserted into the ring in the winner neuron selection and its initial values of the angles θ_i and ψ_i are determined together with the lengths l_a^i and l_b^i from its neighboring neurons to fit the current neighboring Bézier curves. During the learning epoch, only the waypoint locations of the neurons are modified in the adaptation of the network towards the respective $\boldsymbol{s}_p$, i.e., the two control points of the Bézier curves, similarly as in the TSPN.

The explicit optimization of the TTE is performed after each learning epoch during the regeneration. The optimization of the whole trajectory is performed locally using the LIO principle [8], and the values of θ_i, ψ_i, l_a^i, and l_b^i associated with each $\boldsymbol{\nu}_i$ are numerically optimized with respect to the velocity

[1] Here, it is worth noting that using the associated $\boldsymbol{s}_p$ to winner neurons as the waypoint locations is used in the SOM for the DTSP [11], and the Dubins tour is found as a solution of the Dubins touring problem [26].



(a) Learning epoch 6 (b) Learning epoch 22 (c) Learning epoch 37

(d) Learning epoch 46 (e) Learning epoch 57 (f) Final solution

Fig. 4. SOM evolution during surveillance planning with Bézier curves.

profile of the trajectory defined by the three consecutive neurons in the ring $\boldsymbol{\nu}_{i-1}$, $\boldsymbol{\nu}_i$, and $\boldsymbol{\nu}_{i+1}$. The LIO procedure is a multi-variable variant of the hill-climbing optimization technique, where particular variables are consecutively optimized in each iteration of this local numerical optimization, see [8] for further details. The representation (12) enables an independent optimization of the particular variables θ_i, ψ_i, l_a^i, and l_b^i because tangents computed by (13) and (14) implicitly satisfy the smooth constraint (5).

The numerical optimization is performed with the step 0.5% of the particular variable range, and thus the step for the angles θ_i and ψ_i is 0.01π. Three iterations of the local optimization of the whole ring are performed to reduce the computational burden. Notice the waypoint locations of the neurons are only slightly changed in the final learning epochs (see an example of the network evolution in Fig. 4), and thus the optimization is actually performed multiple times for almost the same waypoint locations. The velocity profile for Bézier curve is computed numerically using 200 uniformly distributed samples for the range $\tau \in [0, 1]$ according to (1) and the procedure described in Section II-A.

The proposed algorithm can be summarized as follow.[2]

▷ **Initialization:**

1) Create the ring $\mathcal{N}$ with n neurons around the depot $\boldsymbol{s}_1$.
2) Set the learning gain $\sigma = 12.41n + 0.6$, the learning rate $\mu = 0.5$, and the gain decreasing rate $\alpha = 0.1$. Set the epoch counter $i = 1$.

▷ **Learning Epoch:**

3) For each $\boldsymbol{s}$ in the randomized set $\boldsymbol{s} \in \Pi(S)$
   a) *Winner neuron*: determine $\boldsymbol{\nu}^*$ and $\boldsymbol{s}_p$ as in Fig. 3.
   b) *Adapt* $\boldsymbol{\nu}^*$ and its neighbors towards $\boldsymbol{s}_p$ using (10).

[2]An implementation of the algorithm is available at https://purl.org/faigl/sw.

▷ **Update and Termination:**

4) *Ring regeneration*: remove all non-winner neurons and perform LIO-based optimization of the trajectory.
5) *Update learning parameters:* $\sigma = (1 - \alpha)\sigma$, $i = i + 1$.
6) *Termination condition:* If $i \geq i_{max}$ or *winner neurons are negligibly close to their respective* $\boldsymbol{s}_p$ (e.g., less than 10^{-3}) or *all winner neurons are inside the* δ*-neighborhood of the respective sensing location* `Stop` the adaptation. Otherwise go to Step 3.
7) *Return* the found trajectory.

The computational complexity of a single learning epoch depends on the number of objects of interest n and the number of neurons M, which does not exceed $2n$ because of the ring regeneration, and thus it can be bounded by $O(n^2)$. Notice, the network adapts to all n objects of interest, therefore n winner neurons is the lower bound of the number of neurons because of the ring regeneration in Step 4. The number of learning epochs is set to the constant $i_{max} = 100$, and thus the computational complexity depends on LIO which is performed three times for the whole ring with up to n neurons. Hence the computational complexity of the proposed learning procedure can be bounded by $O(n^2)$. The real required computational time is reported in Section IV.

IV. Results

The proposed 3D surveillance planning has been verified in a series of scenarios for our multi-rotor UAV with $v_{vert} = 1$ ms^{-1}, $a_{vert} = 1$ ms^{-2}, $v_{horiz} = 5$ ms^{-1}, and $a_{horiz} = 2$ ms^{-2}. First, we consider a 2D scenario and compare the proposed approach with the solution of the DTSP with various minimal turning radii ρ and the forward velocity limited to $v = \sqrt{\rho a_{horiz}}$. However, the TTE for the DTSP is computed from the velocity profiles in which the vehicle accelerates on straight line segments up to v_{horiz} and then decelerates to pass the turn maneuver with v. The selected DTSP approaches are heuristics AA [1] and LIO [8] where the sequence of visits is found as the optimal solution of the ETSP [28], SOM-based algorithm [11], and Memetic algorithm [14] with the computational time limited to 10 and 100 seconds. In addition to the DTSP, the evaluated approaches can also solve instances of the DTSPN, and therefore, a solution of the DTSPN has been compared to the proposed surveillance planning with Bézier curves and sensing distance $\delta > 0$. The AA relies on a sequence of waypoint locations, and therefore, a solution of the ETSP with Neighborhoods (ETSPN) found by [29] has been used. The proposed SOM-based algorithm with Bézier curves is denoted SOM (Bézier), and the DTSPN approach [11] is denoted SOM (Dubins). The particular SOM-based learning parameters are set as in Section III and [11], respectively. The population size of the Memetic algorithm is set to $20n$ [14].

All algorithms are considered as randomized and the presented results are average values from 20 trials, i.e., the solution quality is reported as the average value of the TTE. The reported computational times have been measured for the C++ implementation of all algorithms that have been run on the same single core of the i7-67000K CPU running at 4 GHz.

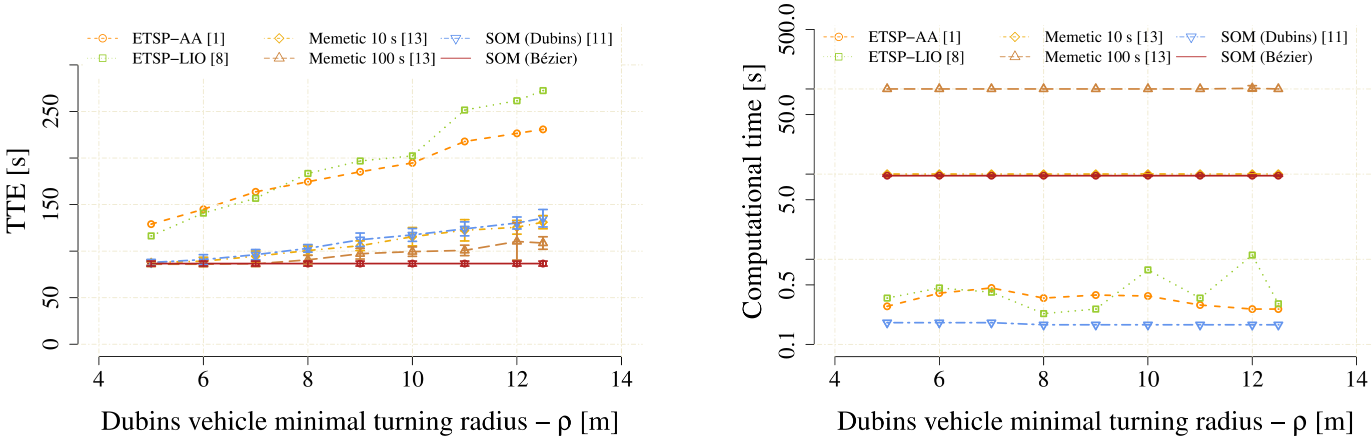


Fig. 5. Comparison of the proposed planning with Bézier curves with the DTSP solutions for different vehicle's minimal turning radius ρ.

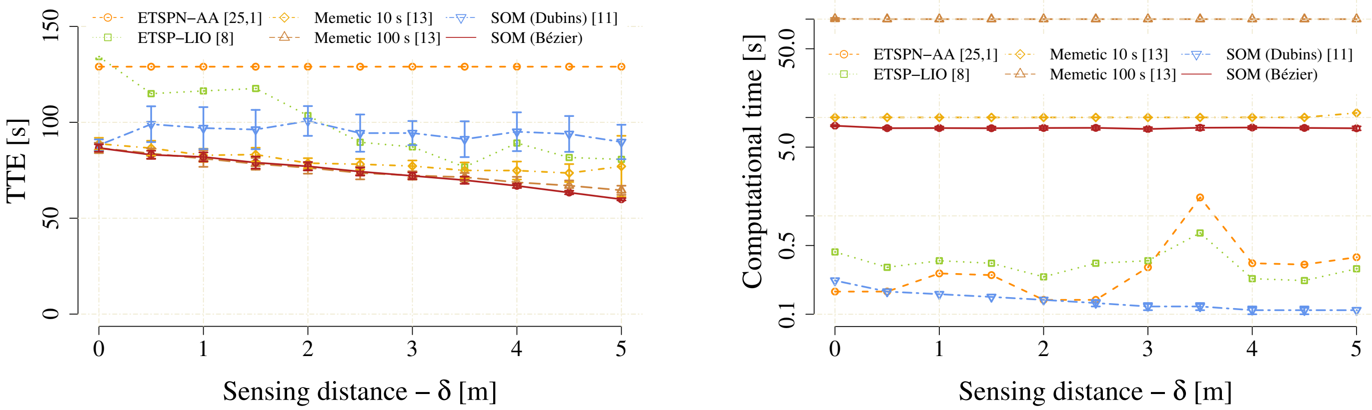


Fig. 6. Influence of the δ-neighborhood to the TTE and computational time.

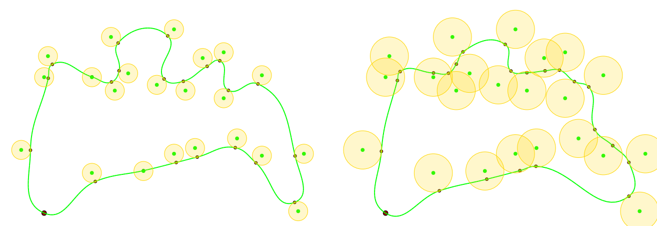
Comparison with DTSP Solvers

The average values of the TTE for $5\text{ m} \leq \rho \leq 12.5\text{ m}$ in the problem with 22 sensing locations are depicted in Fig. 5. Although trajectories found as a solution of the DTSP with a short ρ provide a similar value of the TTE as the proposed approach based on the Bézier curves, the most suitable ρ depends on the particular instance. Moreover, the Dubins vehicle model assumes a constant forward velocity, and therefore, without accelerating along the straight segments, the found DTSP solutions would always require more time than trajectories composed of Bézier curves along which the vehicle accelerates and decelerates according to the trajectory curvature. The solutions as Bézier curves are found in less than 10 seconds.

Influence of Increasing δ and Comparison with DTSPN Solvers

A reduction of the TTE by exploiting a non-zero sensing distance δ has been studied for the same problem as in the previous case and with δ in the range $0 \leq \delta \leq 5$ m. Based on the results in Fig. 5, the DTSPN solvers have been used with $\rho = 5$ m because of the fastest trajectories in the DTSP.

Regarding the results in Fig. 6, solutions provided by all algorithms, except the AA, benefit from $\delta > 0$, but the best results are provided by the proposed surveillance planning



(a) $\delta = 2.5$ m, TTE $= 68.8$ s (b) $\delta = 5.0$ m, TTE $= 57.6$ s

Fig. 7. Selected best found solutions provided by SOM (Bézier) algorithm for the sensing distance δ.

with Bézier curves. The computational requirements slightly decrease because of faster convergence caused by entering the winner neurons into the δ-neighborhood of the sensing locations. The SOM-based DTSPN solver requires less than 0.3 seconds because of the close-form solution of the Dubins maneuvers, which is far faster than the numerical optimizations of the Bézier curves in the proposed approach. However, SOM (Dubins) provides significantly worse results than SOM (Bézier) for $\delta > 0$. Selected solutions are depicted in Fig. 7.

Influence of Altitude Differences for 3D Surveillance Scenarios

The influence of different altitudes of the sensing locations to the TTE has been studied for 3D scenarios created from the same 2D instances with three ranges of the altitude differences:

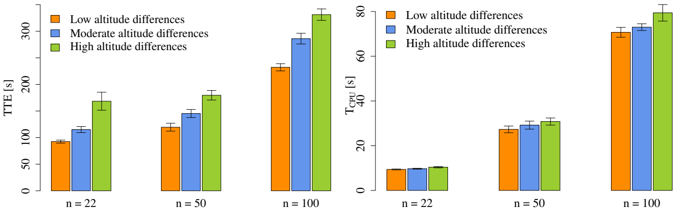
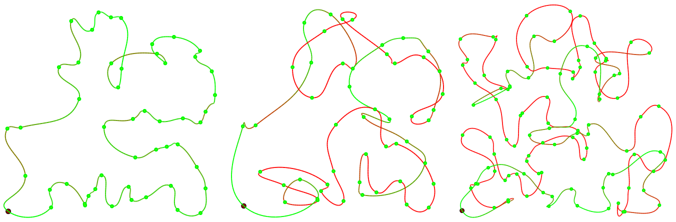


Fig. 8. Influence of altitude differences to the TTE and computational time.



(a) $n = 50$, *low*, TTE = 109.6 s (b) $n = 50$, *high*, TTE = 161.4 s (c) $n = 100$, *high*, TTE = 319.8 s

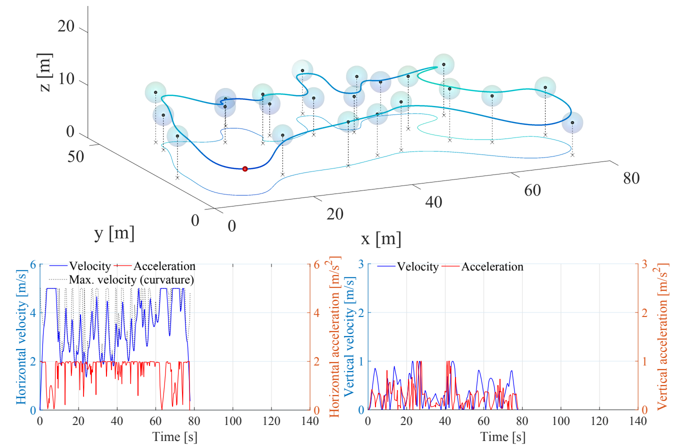
Fig. 9. Selected best found solutions found by the proposed SOM-based approach with Bézier curves for problems with n sensing locations and *low* and *high* altitude differences. The found trajectories are shown as 2D projections with the altitude visualized by the curve color (from green to red for a low altitude to the highest altitude).

low in the range [5, 10] meters; moderate with differences in the range [5, 15]; and high altitude differences in the range [5, 20]. The low altitude differences mostly require horizontal travels, and thus it is expected the vehicle velocity would be saturated in v_{horiz} while for high altitude differences, the vehicle would be limited by v_{vert}. For each range of the altitude differences, ten random instances have been created from the same 2D instances with 22 sensing locations and two random instances with 50 and 100 sensing locations.

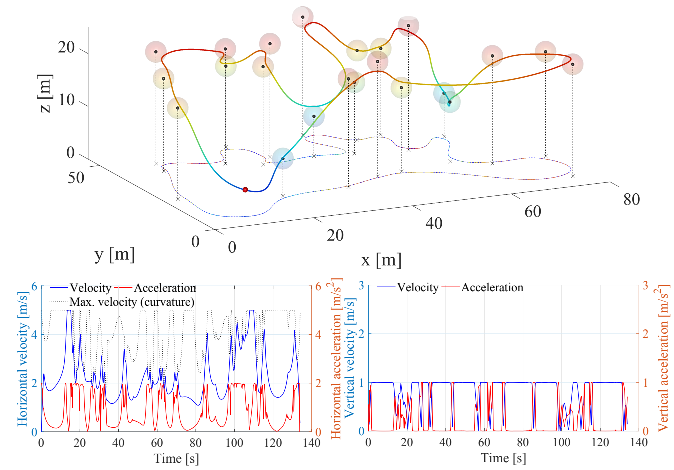
Average values of the TTE and the required computational time are reported in Fig 8. The results support the expectation that higher altitude differences increase the TTE as the vehicle is saturated at the vertical velocity limit. This can be seen in an example of the found solutions shown in Fig. 9 where the red parts of the found surveillance trajectories are at a high altitude and green parts are at a low altitude. The visualized solutions also indicate that a consecutive visitation of the waypoints at a similar altitude is preferred because of the limited vertical velocity. Examples of the velocity profile are shown in Fig. 10. The results also indicate that higher altitude differences cause optimization of Bézier curves more demanding as the required computational time is noticeably increased.

Real Experimental Verification

Finally, a feasibility of the determined 3D trajectories has been experimentally verified in a practical deployment in a scenario with sensing locations at different altitudes. The task of the UAV was to take a snapshot of objects of interest by a camera attached to the UAV with the field of view limited to 4 meters. However, we set the sensing distance $\delta = 2$ m because of possible error of the employed model predictive controller [21] used for the trajectory following, which ensures



(a) low altitude differences



(b) high altitude differences

Fig. 10. An example of the 3D surveillance problems with low and high altitude differences and the velocity and acceleration profiles.

that all the objects of interest would be successfully captured also in cases of small disturbances. The real trajectory has been captured by the RTK GPS with the precision less than 2 cm, which has also been used to control the UAV along the planned trajectory. A snapshot from the experiment and the planned trajectory and two real trajectories (from two performed trials) are visualized in Fig. 11. It can be noticed from the real trajectories that in few cases, the real trajectory does not pass the δ-neighborhood, but it passes (δ+2)-neighborhood, and thus all objects have been captured in both experimental trials.

V. Conclusion

In this paper, we propose a novel unsupervised learning based solution of the surveillance planning with a 3D smooth trajectory parameterized as a sequence of Bézier curves. The proposed approach employs an adaptation procedure of the Self-Organizing Map (SOM) for a solution of the sequencing part of the problem, i.e., a determination of the sequence to visiting the given set of sensing locations. The presented results indicate the proposed solution is a suitable alternative to the solution of the Dubins Traveling Salesman Problem (DTSP) in 2D scenarios, where the Bézier curves better fit

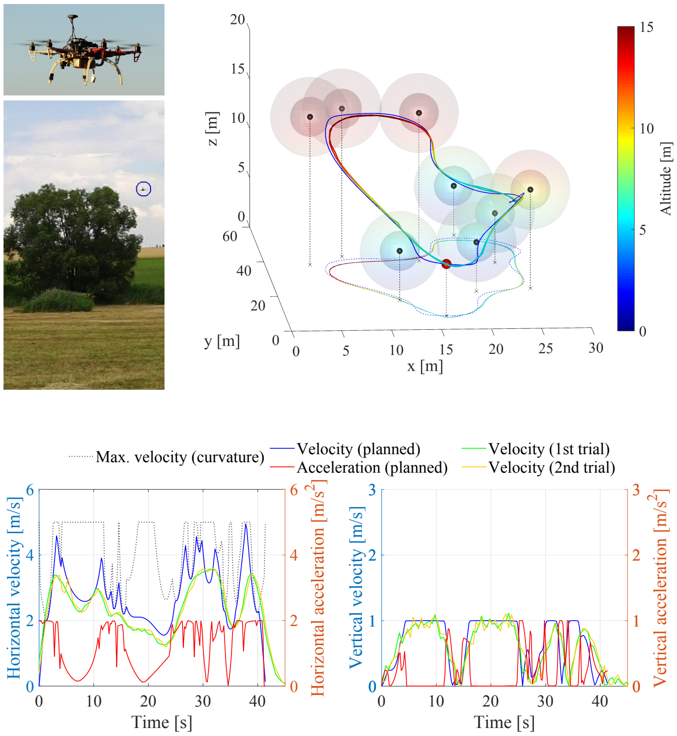


Fig. 11. A snapshot of UAV deployed in the experimental verification of the 3D surveillance scenario with the planned and real vehicle trajectories from two trials. The planned and real profiles of the horizontal and vertical velocities are shown at the bottom.

the motion constraints of the multi-rotor vehicles that are limited by the maximal velocity and acceleration and not by the vehicle's minimal turning radius. In addition, the proposed approach allows to exploit the non-zero sensing distance δ and save the travel time by determination of waypoint locations in the δ-neighborhood of the sensing locations. However, the main benefit of the proposed approach is the ability to find 3D smooth trajectories to visit sensing locations at different altitudes. On the other hand, the proposed approach is more demanding than the SOM-based solution of the DTSP because of the relatively complex computation of the velocity profile and TTE. Besides, the SOM for the DTSP has been generalized to multi-vehicle planning, which might also be possible for the presented approach, and thus it is a subject of our future work.

References

[1] K. Savla, E. Frazzoli, and F. Bullo, "On the point-to-point and traveling salesperson problems for Dubins' vehicle," in *Proceedings of the American Control Conference*. IEEE, 2005, pp. 786–791.

[2] K. J. Obermeyer, P. Oberlin, and S. Darbha, "Sampling-based path planning for a visual reconnaissance unmanned air vehicle," *Journal of Guidance, Control, and Dynamics*, vol. 35, no. 2, pp. 619–631, 2012.

[3] J. T. Isaacs, D. J. Klein, and J. P. Hespanha, "Algorithms for the Traveling Salesman Problem with Neighborhoods Involving a Dubins Vehicle," in *American Control Conference*, 2011, pp. 1704–1709.

[4] R. Pěnička, J. Faigl, P. Váňa, and M. Saska, "Dubins orienteering problem," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1210–1217, April 2017.

[5] P. Oberlin, S. Rathinam, and S. Darbha, "Today's traveling salesman problem," *Robotics & Automation Magazine, IEEE*, vol. 17, no. 4, pp. 70–77, Dec 2010.

[6] J. Ny, E. Feron, and E. Frazzoli, "On the dubins traveling salesman problem," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 265–270, Jan. 2012.

[7] X. Yu, "Optimization Approaches for a Dubins Vehicle in Coverage Planning Problem and Traveling Salesman Problems," Ph.D. dissertation, Auburn University, 2015.

[8] P. Váňa and J. Faigl, "On the dubins traveling salesman problem with neighborhoods," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 4029–4034.

[9] P. Isaiah and T. Shima, "Motion planning algorithms for the Dubins Travelling Salesperson Problem," *Automatica*, vol. 53, pp. 247–255, 2015.

[10] J. Faigl and P. Váňa, "Self-organizing map for the curvature-constrained traveling salesman problem," in *International Conference on Artificial Neural Networks*. Springer International Publishing, 2016, pp. 497–505.

[11] J. Faigl and P. Váňa, "Unsupervised learning for surveillance planning with team of aerial vehicles," in *International Joint Conference on Neural Networks (IJCNN)*, 2017, pp. 4340–4347.

[12] X. Yu and J. Hung, "A genetic algorithm for the dubins traveling salesman problem," in *IEEE International Symposium on Industrial Electronics*, 2012, pp. 1256–1261.

[13] D. G. Macharet, A. Alves Neto, V. F. da Camara Neto, and M. F. Campos, "Efficient target visiting path planning for multiple vehicles with bounded curvature," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 3830–3836.

[14] X. Zhang, J. Chen, B. Xin, and Z. Peng, "A memetic algorithm for path planning of curvature-constrained uavs performing surveillance of multiple ground targets," *Chinese Journal of Aeronautics*, vol. 27, no. 3, pp. 622–633, 2014.

[15] M. Neubauer and A. Müller, "Smooth orientation path planning with quaternions using B-splines," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 2087–2092.

[16] E. Papadopoulos, I. Papadimitriou, and I. Poulakakis, "Polynomial-based obstacle avoidance techniques for nonholonomic mobile manipulator systems," *Robotics and Autonomous Systems*, vol. 51, no. 4, pp. 229–247, 2005.

[17] Kwangjin Yang and S. Sukkarieh, "An Analytical Continuous-Curvature Path-Smoothing Algorithm," *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 561–568, Jun. 2010.

[18] Y. Wang, S. Wang, and M. Tan, "Path Generation of Autonomous Approach to a Moving Ship for Unmanned Vehicles," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 9, pp. 5619–5629, Sep. 2015.

[19] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *American Journal of Mathematics*, pp. 497–516, 1957.

[20] P. Bézier, "Numerical control: Mathematics and applications," *International Journal for Numerical Methods in Engineering*, vol. 6, no. 3, pp. 456–456, 1973.

[21] T. Báča, G. Loianno, and M. Saska, "Embedded model predictive control of unmanned micro aerial vehicles," in *21st International Conference on Methods and Models in Automation and Robotics (MMAR)*, 2016, pp. 992–997.

[22] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient motion primitive for quadrocopter trajectory generation," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1294–1310, Dec 2015.

[23] J. Faigl, V. Vonásek, and L. Přeučil, "Visiting convex regions in a polygonal map," *Robotics and Autonomous Systems*, vol. 61, no. 10, pp. 1070–1083, 2013.

[24] S. Somhom, A. Modares, and T. Enkawa, "A self-organising model for the travelling salesman problem," *Journal of the Operational Research Society*, vol. 48, no. 9, pp. 919–928, Sep 1997.

[25] T. Kohonen, M. R. Schroeder, and T. S. Huang, Eds., *Self-Organizing Maps*, 3rd ed. Springer-Verlag New York, Inc., 2001.

[26] J. Faigl, P. Váňa, M. Saska, T. Báča, and V. Spurný, "On solution of the dubins touring problem," in *European Conference on Mobile Robots (ECMR)*, 2017, pp. 1–6.

[27] M. Tucci and M. Raugi, "Stability analysis of self-organizing maps and vector quantization algorithms," in *International Joint Conference on Neural Networks (IJCNN)*, 2010, pp. 1–5.

[28] D. Applegate, R. Bixby, V. Chvátal, and W. Cook, "Concorde TSP Solver," 2003, [cited 9 Sep 2017]. [Online]. Available: http://www.tsp.gatech.edu/concorde.html

[29] X. Yu and J. Hung, "Optimal path planning for an autonomous robot-trailer system," in *38th Annual Conference on IEEE Industrial Electronics Society (IECON)*, 2012, pp. 2762–2767.