Noname manuscript No.
(will be inserted by the editor)

Unsupervised Learning based Solution of the Close Enough Dubins Orienteering Problem

Jan Faigl

Received: date / Accepted: date

Abstract This paper reports on the application of novel unsupervised learning based method called the Growing Self-Organizing Array (GSOA) to data collection planning with curvature-constrained paths that is motivated by surveillance missions with aerial vehicles. The planning problem is formulated as the Close Enough Dubins Orienteering Problem (CEDOP) which combines combinatorial optimization with continuous optimization to determine the most rewarding data collection path that does not exceed the given travel budget and satisfies the motion constraints of the vehicle. The combinatorial optimization consists of selecting a subset of the most rewarding data to be collected and the schedule of data collection. On the other hand, the continuous optimization stands to determine the most suitable waypoint locations from which selected data can be collected together with the determination of the headings at the waypoints for the used Dubins vehicle model. The existing purely combinatorial approaches need to discretize the possible waypoint locations and headings into some finite sets, and the solution is computationally very demanding because the problem size is quickly increased. On the contrary, the employed GSOA performs online sampling of the waypoints and headings during the adaptation of the growing structure that represents the requested curvature-constrained data collection path. Regarding the presented results, the proposed approach provides solutions to orienteering problems with competitive quality, but it is significantly less computationally demanding.

J. Faigl
Department of Computer Science
Faculty of Electrical Engineering
Czech Technical University in Prague
Technická 2, 166 27 Prague, Czech Republic
E-mail: faiglj@fel.cvut.cz

Keywords Data Collection Planning · Surveillance Missions and Aerial Vehicles · Growing-Self-Organizing Array · GSOA

1 Introduction

In this paper, we address a variant of routing problem motivated by surveillance and data collection missions performed by aerial vehicles. The motivation problem is to maximize the sum of the collected rewards from a set of target locations while respecting the limited operational time and motion constraints of the vehicles. Each target locations has associated a reward value that characterizes the importance of the collected data. The reward is collected whenever the vehicle passes the target location within the specified sensing/-communication range, e.g., a snapshot of the area is taken by a downward-looking camera in surveillance missions [1] or data are read from wireless sensor networks [2,3]. Moreover, the requested data collection path has to respect motion constraints of the vehicle that can be modeled by Dubins vehicle [4] with a constant forward velocity and minimum turning radius, which fits motion constraints of fixed-wing aircraft.

The addressed data collection planning problem to determine the most rewarding data collection schedule respecting the limited travel budget can be formulated as a variant of the Orienteering Problem (OP) [5], which can be stated as follows. For a given set of the target locations, each with the associated reward, and specified initial and terminal locations of the vehicle, the OP stands to maximize the sum of the collected rewards by a tour that does not exceed the given travel budget. The OP can be considered as the Knapsack problem to choose the most rewarding locations combined with the solution of the Traveling Salesman Problem (TSP) to find the shortest tour connecting the locations, and

thus assure the tour length does not exceed the limited travel budget. The OP is at least NP-hard as it becomes the TSP for the travel budget that allows visiting all the locations; the OP becomes the TSP [5].

The standard formulation of the OP has been generalized for exploiting non-zero sensing range in [6,7], where the problem is called the Orienteering Problem with Neighborhoods (OPN). However, in the case of the disk-shaped neighborhood defined by the particular sensing range, the problem is similar to the Close Enough TSP (CETSP) that has been introduced in [8] to address data collection planning for retrieving data about electricity, water or gas consumption using wireless communication. Therefore, the problem is called the *Close Enough Orienteering Problem* (CEOP) to emphasize that the disk-shaped neighborhood is considered in this paper. Motion constraints of aerial vehicles in the solution of the OP has been addressed in [9] where the *Dubins Orienteering Problem* (DOP) is introduced. DOP has been later generalized for non-zero sensing range in [10,11] and the problem is called the *Close Enough Dubins Orienteering Problem* (CEDOP). In [9], the solution of DOP is based on a variant of the combinatorial Variable Neighborhood Search (VNS) meta-heuristic [12], which has also been used for solving CEDOP in [10].

Although the VNS-based approach [9,10] provides a solution of CEDOP, it is computationally very demanding because it explicit samples possible waypoint locations in addition to heading values at the waypoints to allow determining the optimal Dubins tour connecting the waypoints [13]. The size of such discretized problem quickly grows with each possible waypoint location that can be, in general, selected from an infinite set defined by the disk-shaped neighborhood of each target location. The computational requirements of DOP and CEDOP have been addressed by unsupervised learning approach based on Self-Organizing Map (SOM) in [14] and [11], respectively.

In the current paper, the results of [14,11] are further developed using novel unsupervised learning procedure called *Growing Self-Organizing Array* (GSOA) [15] that can be considered as a simplified and consolidated work on the previous SOM-based algorithms for various routing problems. The GSOA is inspired by the SOM approaches for data collection planning with an underlying solution of the TSP, but it provides a simplified notation specifically focused on a combination of the sequencing part of the combinatorial optimization with continuous optimization such as selection of the waypoints or headings from continuous sets associated to each target location. Based on the presented results, the herein proposed GSOA-based solution to orienteering problems provides competitive and in most of the cases better solutions than the existing unsupervised learning approaches but with noticeably less computational requirements. Moreover, the developed solution allows considering specific sensing ranges for individual locations, and it also allows solving problems for a team of data collection vehicles. The paper contributions are considered as follows.

- Novel GSOA-based approach to the CEDOP generalized for
- the CEDOP instances with individual sensing range;
- and multi-vehicle CEDOP instances.
- Novel learning rule that improves solutions of the evaluated instances of the orienteering problem in comparison to the previous SOM-based approaches.

The paper is organized as follows. An overview of the related work is presented in the following section. All the addressed variants of the orienteering problem are formally introduced in Section 3 together with an overview of the Dubins vehicle model. The GSOA-based unsupervised learning for CEDOP is proposed in Section 4. Achieved results and comparison with the previous approaches are reported in Section 5. Concluding remarks are dedicated to Section 6.

2 Related Work

The CEDOP is a variant of the orienteering problem with two important generalizations of the regular OP [5]. First, in CEDOP, it is allowed to collect the reward within some specified distance from the target, which may considerably reduce the total tour length [16]. Without the limited travel budget, such a routing problem is known as the TSP with Neighborhoods (TSPN) and specifically with the disk-shaped neighborhood as the Close Enough TSP (CETSP) [17,18]. However, none of the previous approaches has been directly employed in solving the OP with Neighborhoods (OPN) [19, 20] despite several approaches including approximation and heuristic algorithms for the TSPN have been proposed, including prior work on unsupervised learning [21,22]. Probably the only approach already deployed to the OPN is based on the SOMs [6,7], where the preference of highly rewarding target locations is addressed by a duplication of the targets according to their rewards and repeated adaptation of the network to such targets. The duplication increases the computational requirements that have been addressed by improved adaptation procedure [23], which provides competitive results but it is significantly less computationally demanding.

The second generalization is related to the curvature-constrained path that is modeled as Dubins vehicle [4]. The main difficulty of routing problems with Dubins vehicle is that the length of the path depends not only on the distance between the waypoint locations but also on the particular heading angles of the vehicle at the waypoints. Thus, a solution of the Dubins TSP (DTSP) [24] combines challenges of the combinatorial optimization of the underlying TSP with the determination of the most suitable heading values which

can be arbitrarily selected from the interval $[0, 2\pi)$. An important result has been shown for a given sequence of waypoint locations. High-quality solutions, i.e., almost optimal values of the headings at the waypoints, can be found using lower bound solution [25] by iterative refinement of the possible heading values [13]. However, the sequence is not known in the DTSP. Moreover, even the waypoint locations themselves can be chosen from the neighborhood defined by non-zero sensing range, which better fits properties of real vehicles in surveillance missions. Then, the DTSP becomes the DTSP with Neighborhoods (DTSPN) [26, 27], where the most suitable waypoint locations have to be determined in addition to the heading angles.

Computational challenges of the DTSPN have been addressed by approximation [28], heuristics [29–31], and sampling-based [32] algorithms, but also by evolutionary methods [33, 27], and recently proposed unsupervised learning-based approaches [34, 35]. On the other hand, the operational time of unmanned aerial vehicles is often limited, and therefore, finding a cost-efficient path to visit all locations may not provide feasible data collection schedule that respects the limited travel budget T_{max} of the vehicle. Thus the problem is to determine a path not exceeding T_{max} such that the vehicle visits the most important target locations and safely returns to the defined terminal location, e.g., for refuel or battery replacement.

Even though many approaches for the Dubins TSP and TSPN have been proposed in the literature, only a few works can be found for the generalization of the Orienteering Problem for planning with Dubins vehicle, i.e., the *Dubins Orienteering Problem* (DOP). A possible way to solve DOP can be based on the decoupled approach in which a subset of the target locations to be visited by the vehicle is determined as the regular Euclidean OP. Then, the final data collection path can be found as a solution of the DTSP for such a subset. However, such a tour may exceed the given travel budget, and therefore, a direct solution of DOP is preferred.

The first direct approach to DOP has been proposed in [9], where the randomized variant of the Variable Neighborhood Search (VNS) metaheuristic [12] has been employed to deal with the constraints of Dubins vehicle. The algorithm follows sampling-based approach for the DTSP, and possible headings at each target location are sampled into a discrete set of headings, e.g., 16 heading values have been used for the results presented in [9], and the problem is solved as purely combinatorial optimization.

The VNS-based approach [9] can be generalized to consider possible locations from which data can be retrieved and thus solve CEDOP. For each target location, a discrete set of possible locations can be sampled, and the problem can be solved in a very similar way as the original solution of DOP [9]. However, it is necessary to consider particular headings for each such sampled location, and thus the problem size quickly grows with the number of samples. Based on the early results reported in [10], the VNS-based solution seems to be very computationally demanding.

The studied CEDOP can be formulated as the recently proposed generalization of the OP called the Set Orienteering Problem [36], but it is computationally demanding similarly to the VNS-based approach as the size of discretized CEDOP is quickly increasing because, for each discretized possible waypoint location, several samples of heading values are needed to find a solution of the satisfiable quality [10].

The recent advancements on the SOM-based solution to the OP [37, 7, 23] and DTSPN [34, 35] provide the groundwork to address challenges of CEDOP by the unsupervised learning. The main advantage of the unsupervised learning to Dubins routing problems is that the heading values are sampled during the learning. Thus, it may provide Dubins paths of similar length with a lower number of considered samples of the vehicle headings. Regarding the results reported in [11], it is significantly faster than explicitly considered headings samples in the VNS-based approach.

The first SOM-based attempt to DOP has been proposed in [14] and further generalized to CEDOP in [11]. The herein presented results are based on these early achievements using SOM, but they are further developed considering novel unsupervised learning based procedure for routing problems called *Growing Self-Organizing Array* (GSOA) [15]. Even though the GSOA is inspired by the previous successful deployments of the SOM-based approaches to various TSP-like routing problems, it simplifies the notation, and it uses a growing array of nodes. The learning procedure explicitly considers a determination of the waypoint locations as non-zero sensing range is a practical assumption and its exploitation saves the travel cost by avoiding precise visitation of the target locations [22]. Regarding [15] and [38], the GSOA provides better solutions with lower computational requirements than the previous SOM-based approaches, and therefore, its considered in this paper.

The GSOA for the OP is generalized to the CEDOP and further extended by a novel approach for selecting the subset of targets to be visited. The proposed solution is compared with the previous SOM-based solutions of the OP, CEOP, and DOP to show benefits of the proposed novel GSOA-based algorithm. Finally, the proposed approach allows to consider specific sensing ranges per each particular target, and it also enables a solution of the OP with a team of vehicles, i.e., the Team Orienteering Problem [20]. Feasibility of the proposed CEDOP algorithm in a solution of these variants of orienteering problem is demonstrated for selected instances to support the applicability of the proposed GSOA-based approach to other routing problems.

3 Problem Statement

The addressed *Close Enough Dubins Orienteering Problem* (CEDOP) is a variant of the Orienteering Problem (OP) with the disk-shaped neighborhood at which the reward of the corresponding target location can be collected and where the requested tour respects the motion constraints of Dubins vehicle. The particularly addressed variants of the OP are formally introduced here to show the main challenges of the Close Enough OP and curvature-constrained path of Dubins vehicle.

The motivation of the studied problem is data collection and surveillance missions where the problem is to determine a cost-efficient curvature-constrained path to retrieve the most valuable measurements from a set of sensors S such that, the total tour length does not exceed the given travel budget $\mathrm{T_{max}}$. Each sensor $s_i \in S$ has associated reward r_i that can be remotely collected by the vehicle within the distance δ. The initial and terminal locations of the vehicle are prescribed as $\boldsymbol{s}_1$ and $\boldsymbol{s}_n$ and their associated rewards are zero, $r_1 = r_n = 0$, where n is the total number of the locations S, $n = |S|$, as in the regular OP [39]. The sensors are placed in a plane and to simplify the notation, the location of the sensor s_i is denoted $\boldsymbol{s}_i \in \mathbb{R}^2$.

Because of the limited travel budget $\mathrm{T_{max}}$, it may not be possible to collect data from all the sensors, and thus a subset S_k of k sensors $S_k \subseteq S$ has to be determined such that the sum of the collected rewards is maximized and the path for collecting the rewards from the selected subset S_k does not exceed $\mathrm{T_{max}}$. The initial and terminal locations are prescribed which in turns means $s_1 \in S_k$ and $s_n \in S_k$, and therefore, we need to determine the data collection schedule as a sequence of visits to the sensors $\Sigma = (s_{\sigma_1}, \ldots, s_{\sigma_k})$, where $1 \leq \sigma_i \leq n$, $\sigma_1 = 1$, and $\sigma_k = n$. For $\delta = 0$, the data collection path directly connects the selected sensors S_k, and its length has to respect $\mathrm{T_{max}}$. The problem becomes the regular OP that can be formally defined as Problem 1.

Problem 1 (Orienteering Problem (OP))

$$
\begin{aligned}
&\underset{\Sigma, k, S_k}{\textit{maximize}} \quad R = \sum_{i=1}^{k} r_{\sigma_i} \\
&\textit{subject to} \\
&\qquad \sum_{i=2}^{k} \left\| (\boldsymbol{s}_{\sigma_{i-1}}, \boldsymbol{s}_{\sigma_i}) \right\| \leq T_{max}, \qquad (1) \\
&\qquad \Sigma = (\sigma_1, \ldots, \sigma_k),\ 1 \leq \sigma_i \leq n \\
&\qquad 2 \leq k \leq n,\ \sigma_1 = 1,\ \sigma_k = n \\
&\qquad S_k = \{s_{\sigma_1}, \ldots, s_{\sigma_k}\},\ S_k \subseteq S \\
&\qquad s_{\sigma_1} = s_1, s_{\sigma_k} = s_n
\end{aligned}
$$

where R is the sum of the collected rewards and $\left\| (\boldsymbol{s}_{\sigma_{i-1}}, \boldsymbol{s}_{\sigma_i}) \right\|$ is Euclidean distance between the locations $\boldsymbol{s}_{\sigma_{i-1}}$ and $\boldsymbol{s}_{\sigma_i}$.

If the reward can be determined within $\delta > 0$ range, it is also requested to determine a waypoint location $\boldsymbol{p_i} \in \mathbb{R}^2$ for each selected sensor $s_i \in S_k$ such that $\|(\boldsymbol{p}_i, \boldsymbol{s}_i)\| \leq \delta_i$. A waypoint to collect data from s_i is in the disk-shaped neighborhood centered at $\boldsymbol{s_i}$ with the radius δ_i. The initial and terminal locations of the vehicle are strictly prescribed, and therefore, s_1 and s_n are always selected and considered without the sensing range, i.e., $\delta_1 = \delta_n = 0$. Hence, the first and last waypoints of data collection path are always $\boldsymbol{p}_1 = \boldsymbol{s}_1$ and $\boldsymbol{p}_n = \boldsymbol{s}_n$, respectively. Note that if the path connecting $\boldsymbol{s}_1$ with $\boldsymbol{s}_n$ exceeds the budget $\mathrm{T_{max}}$, the problem does not have a feasible solution. The problem is to determine S_k, the sequence of visits to the sensors Σ together with the respective waypoint locations $P = (\boldsymbol{p}_{\sigma_1}, \ldots, \boldsymbol{p}_{\sigma_k})$ such that data from s_{σ_i} can be retrieved from $\boldsymbol{p}_{\sigma_i}$ and the path connecting P respects $\mathrm{T_{max}}$. The problem can be formulated as the *Close Enough OP* (CEOP) that can be formally defined as Problem 2.

Problem 2 (Close Enough Orienteering Problem (CEOP))

$$
\begin{aligned}
&\underset{\Sigma, k, S_k, P}{\textit{maximize}} \quad R = \sum_{i=1}^{k} r_{\sigma_i} \\
&\textit{subject to} \\
&\qquad \sum_{i=2}^{k} \left\| (\boldsymbol{p}_{\sigma_{i-1}}, \boldsymbol{p}_{\sigma_i}) \right\| \leq T_{max}, \qquad (2) \\
&\qquad \Sigma = (\sigma_1, \ldots, \sigma_k),\ 1 \leq \sigma_i \leq n \\
&\qquad 2 \leq k \leq n,\ \sigma_1 = 1,\ \sigma_k = n \\
&\qquad S_k = \{s_{\sigma_1}, \ldots, s_{\sigma_k}\},\ S_k \subseteq S \\
&\qquad P = (\boldsymbol{p}_{\sigma_1}, \ldots, \boldsymbol{p}_{\sigma_k}),\ \left\| (\boldsymbol{p}_{\sigma_i}, \boldsymbol{s}_{\sigma_i}) \right\| \leq \delta_i \\
&\qquad \delta_{\sigma_1} = \delta_{\sigma_k} = 0,\ \boldsymbol{p}_{\sigma_1} = \boldsymbol{s}_1,\ \boldsymbol{p}_{\sigma_k} = \boldsymbol{s}_n
\end{aligned}
$$

where P is a sequence of waypoints from which data from the respective sensors S_k are collected. If all the sensing ranges are the same (except for s_1 and s_n which are always without the neighborhood) a single value δ is considered in the rest of this paper to simplify the notation.

In data collection planning for Dubins vehicle, the path connecting the waypoints has to respect not only $\mathrm{T_{max}}$ but also motion constraints of Dubins vehicle, i.e., the minimum turning radius ρ. Therefore the data collecting path has to be found as a sequence of Dubins maneuvers. The state of Dubins vehicle can be described as $\boldsymbol{q} = (x, y, \theta)$ where $\boldsymbol{p} = (x, y)$ is the vehicle position in the plane $\boldsymbol{p} \in \mathbb{R}^2$ and θ is the vehicle heading $\theta \in [0, 2\pi)$, i.e., $\theta \in \mathbb{S}^1$, and thus $\boldsymbol{q} \in SE(2)$. The model of Dubins vehicle [4] assumes the vehicle is moving with a constant forward velocity v and its minimum turning radius is ρ. For the control input u, the vehicle motion can be described as

$$
\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = v \begin{bmatrix} \cos\theta \\ \sin\theta \\ u \cdot \rho^{-1} \end{bmatrix}, \quad |u| \leq 1. \qquad (3)
$$

Based on that, a waypoint $\boldsymbol{q}_i = (\boldsymbol{p}_i, \theta_i)$ for retrieving data from the sensor s_i consists of the waypoint location $\boldsymbol{p}_i$ and the expected heading θ_i of data collection vehicle modeled as Dubins vehicle.

The optimal curvature-constrained path respecting (3) that connects two states $q_i, q_j \in SE(2)$ is a straight line segment (S) or consists of S and arcs with the curvature ρ that can be one of two types: left (L) and right (R); and it can be computed analytically as one of six possible Dubins maneuvers: LSL, LSR, RSL, RSR, LRL, and RLR [4]. Thus, in the solution of the orienteering problem, it is requested to determine the subset of sensors S_k, the sequence of visits Σ to S_k together with the respective waypoint locations, but it is also needed to determine the optimal headings at the particular waypoints such that the sum of the lengths of Dubins maneuvers visiting the waypoints respects $\mathrm{T_{max}}$ and the sum of the collected rewards is maximized. Such a problem can be formally defined as Problem 3.

Problem 3 (Close Enough Dubins Orienteering Problem (CEDOP))

$$
\begin{aligned}
&\underset{\Sigma,k,S_k,P,\Theta}{\textit{maximize}} \quad R = \sum_{i=1}^{k} r_{\sigma_i} \\
&\textit{subject to} \\
&\qquad \sum_{i=2}^{k} \mathcal{L}(\boldsymbol{q}_{\sigma_{i-1}}, \boldsymbol{q}_{\sigma_i}) \leq T_{max}, \\
&\qquad \Sigma = (\sigma_1, \ldots, \sigma_k),\ 1 \leq \sigma_i \leq n \\
&\qquad 2 \leq k \leq n,\ \sigma_1 = 1,\ \sigma_k = n \\
&\qquad S_k = \{s_{\sigma_1}, \ldots, s_{\sigma_k}\},\ S_k \subseteq S \\
&\qquad \boldsymbol{q}_{\sigma_i} = (\boldsymbol{p}_{\sigma_i}, \theta_{\sigma_i}),\ \boldsymbol{q}_{\sigma_i} \in SE(2) \\
&\qquad \Theta = (\theta_{\sigma_1}, \ldots, \theta_{\sigma_k}),\ \theta_{\sigma_i} \in [0, 2\pi) \\
&\qquad P = (\boldsymbol{p}_{\sigma_1}, \ldots, \boldsymbol{p}_{\sigma_k}),\ \left\|(\boldsymbol{p}_{\sigma_i}, \boldsymbol{s}_{\sigma_i})\right\| \leq \delta_i \\
&\qquad \delta_{\sigma_1} = \delta_{\sigma_k} = 0,\ \boldsymbol{p}_{\sigma_1} = \boldsymbol{s}_1,\ \boldsymbol{p}_{\sigma_k} = \boldsymbol{s}_n
\end{aligned}
\quad , \quad (4)
$$

where $\mathcal{L}(\boldsymbol{q}_{\sigma_{i-1}}, \boldsymbol{q}_{\sigma_i})$ is the length of Dubins maneuver from $\boldsymbol{q}_{\sigma_{i-1}}$ to $\boldsymbol{q}_{\sigma_i}$ [4].

CEDOP can be considered as a combination of two combinatorial optimization subproblems and two continuous optimization subproblems. The combinatorial optimization is in determining the subset of sensors $S_k \subseteq S$ to be visited and the determination of the permutation of their visits Σ. The continuous optimization parts are the determination of $k-2$ waypoint locations (because of $s_1, s_n \in S_k$) within the continuous disk-shaped neighborhood of the respective sensors together with the determination of the k optimal headings of Dubins vehicle (3) at the waypoints. However, to determine the optimal path connecting two states in CEDOP, we need to determine the respective locations of such waypoints $\boldsymbol{p}_i, \boldsymbol{p}_j \in \mathbb{R}^2$ and also the particular headings $\theta_i, \theta_j \in \mathbb{S}^1$ of the vehicle at the waypoints.

4 Proposed Unsupervised Learning for the Close Enough Dubins Orienteering Problem

The proposed solution of CEDOP follows early results on SOM-based approach [14, 11], but the novel unsupervised learning procedure called the *Growing Self-Organizing Array* (GSOA) [15] is employed. The GSOA is an iterative learning procedure where the solution is represented as growing array of nodes $\mathcal{N} = (\nu_1, \ldots, \nu_M)$, where each node ν_i is associated with a particular location $\boldsymbol{\nu}_i \in \mathbb{R}^2$, sensor $s \in S$, and waypoint location $\boldsymbol{p}_s$ at which the sensor s can be covered within the sensing range δ, i.e., $\|(\boldsymbol{p}_s, \boldsymbol{s})\| \leq \delta$. The learning is organized into a fixed number of learning epochs, and during each learning epoch, $\mathcal{N}$ is sequentially adapted to all sensors S that are picked in a random order to avoid local optima.

In GSOA for TSP-like routing problems [15], a single adaptation of $\mathcal{N}$ towards $s \in S$ consists of determining a new winner node ν^* together with its waypoint location $\boldsymbol{p}_s$ for covering s, and a movement of the winner node ν^* and its neighbouring nodes towards $\boldsymbol{p}_s$ using the neighbouring function similarly as in SOM. Since n new winner nodes can be added to $\mathcal{N}$ in every learning epoch, the nodes from the previous epoch are removed at the end of each learning epoch, and only the new winner nodes are preserved for the next epoch. Hence, a solution of the routing problem is available after each learning epoch using the associated waypoint locations to the nodes. Thus, the solution is extracted by traversing the array $\mathcal{N}$ and using the waypoints associated with the preserved nodes.

There are two main challenges to employ the GSOA learning procedure in a solution of the orienteering problems with Dubins vehicle. The first challenge is in addressing the selection of the subset of sensors S_k that can be covered within the limited travel budget $\mathrm{T_{max}}$, where it is necessary to decide which sensor should be visited and which should be preferably avoided. The second challenge is also related to $\mathrm{T_{max}}$ and the problem is how to satisfy the motion constraints of Dubins vehicle (3). In particular, we need to determine heading values for the currently determined waypoint locations and validate if Dubins path connecting the waypoints is feasible and its length does not exceed $\mathrm{T_{max}}$. The advantage of the GSOA is in explicit consideration of waypoints, which also simplifies the procedure in comparison to the previous SOM-based approach [11].

The rest of this section is organized as follows. The main concepts of generalizing the GSOA for the CETSP [15] to orienteering problems with Dubins vehicle as a solution of CEDOP is presented in the following section. Solution representation and summary of the used notation are provided in Section 4.2. The proposed GSOA unsupervised learning for CEDOP is described in Section 4.3. Computational com-

plexity analysis is reported in Section 4.4 and an example of solutions with a team of vehicles is shown in Section 4.5.

4.1 Extension of the GSOA for CEDOP

Selecting a subset S_k of S sensors – Since the array of nodes $\mathcal{N}$ describes not only the order but also the waypoints, a data collection tour is almost instantly available during the learning procedure. A new winner node ν^* is determined as the closest point of the data collection path $\mathcal{P}_C(\mathcal{N})$ represented by the current array of nodes $\mathcal{N}$. Besides, also the particular new waypoint is determined for each winner node, and therefore, the prolongation of $\mathcal{P}_C(\mathcal{N})$ can be computed for the case the new waypoint is visited by the path. If the length of such a path would exceed $\mathrm{T_{max}}$, it probably does not make sense to include the particular sensor in the solution. However, when $\mathcal{P}_C(\mathcal{N})$ is saturated regarding $\mathrm{T_{max}}$, no new winner node and thus sensor would be included in the solution. Therefore, the idea of removing nodes from $\mathcal{N}$ in benefit of collecting data from more rewarding sensors introduced in the previous unsupervised learning approaches [23, 14, 11], is here generalized to removing up to l_{max} winner nodes of the current epoch. The nodes are considered in the order of increasing ratio $\varsigma(\nu_i)$ that is computed as

$$\varsigma(\nu_i) = \frac{r(\nu_i.s)}{\|(\boldsymbol{\nu}_i, \nu_i.\boldsymbol{p}_s)\|}, \tag{5}$$

where $\nu_i.\boldsymbol{p}_s$ is the waypoint location to cover the sensor $\nu_i.s$ associated to the node ν_i, $r(\nu_i.s)$ is the reward of that sensor, and $\boldsymbol{\nu}_i$ is the current location of the node ν_i. The distance $\|(\boldsymbol{\nu}_i, \nu_i.\boldsymbol{p}_s)\|$ can be prolonged during the adaptation to a new waypoint location. Therefore, the neighboring nodes are also adapted towards their waypoint locations to support a selection of rewarding sensors. If the length of the path $\mathcal{P}_C(\mathcal{N})$ with the new node ν^* and removed l_{max} nodes is still over the budget $\mathrm{T_{max}}$, $\mathcal{N}$ is reverted to the state before removing the nodes, ν^* is not added to $\mathcal{N}$, and the particular sensor is not included in the subset S_k.

Satisfying motion constraints of Dubins vehicle – For the Euclidean OP, the data collection path $\mathcal{P}_C(\mathcal{N})$ can be directly constructed using the waypoint locations P associated to the winner nodes. However, for Dubins vehicle, we need particular heading values of the vehicle at the waypoint locations. Even though $\mathcal{N}$ provides a sequence of waypoint locations and the requested headings can be determined as a solution of the so-called Dubins Touring Problem (DTP) by an iterative refinement procedure [13], it would be computationally demanding because of too many path length queries are needed during the learning. Therefore, similarly as in SOM for Dubins planning [35, 14, 11], additional heading values are associated to each node $\nu_i \in \mathcal{N}$. Then, the length of Dubins path represented by the current winner nodes $\mathcal{P}_C(\mathcal{N})$ is determined as a solution of the DTP for a fixed set of headings per each waypoint by the forward search method schematical described in Fig. 1. Further details about solution of the DTP can be found in [13] or [14].

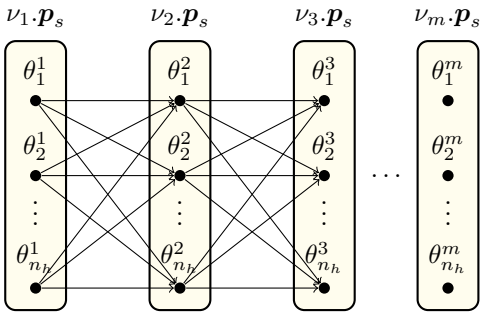


Fig. 1 The length of Dubins path represented by the current winner nodes $(\nu_1, \ldots, \nu_m)$ is determined by a forward search procedure in a graph with m layers corresponding to the m winner nodes, where each layer consisting of n_h nodes according to the heading values associated with the node. The solution of the OP always starts at the initial locations $\boldsymbol{s}_1$ and terminates at $\boldsymbol{s}_n$. Therefore the graph always has at least two layers, and thus at least $2n_h$ vertices. The connections between the vertices of two consecutive layers represent particular optimal Dubins maneuvers [4] between the corresponding states defined by the waypoint location $\nu_i.\boldsymbol{p}_s$ of the node ν_i and the particular heading values θ_j^i, $1 \leq j \leq n_h$. Since the path is open, the optimal solution for the given sequence and particular values of the headings is determined in $O(mn_h^2)$. The waypoint locations and heading values are determined during the unsupervised learning of the GSOA.

4.2 Solution Representation and Notation

The following notation is used to distinguish the nodes, nodes locations, and the waypoints associated with the nodes. The nodes are organized in an array of nodes denoted $\mathcal{N} = \{\nu_1, \ldots, \nu_M\}$, where M is the current number of nodes in the array. Each node $\nu_i \in \mathcal{N}$ is associated with the point $\boldsymbol{\nu}_i$ that is in the same space as the sensor locations. The node is also associated with the sensor $s \in S$ for which it has been selected as the winner node. Besides, each node is further associated with the particular waypoint location $\boldsymbol{p}_s$ to cover s within δ distance. Finally, up to $2h+1$ heading values are associated to the node from which up to $2h+1$ possible waypoints (at the location $\boldsymbol{p}_s$) are used in constructing the graph for solving the associated DTP, see Fig. 1. In the herein addressed problem, sensors are located in a plane, and thus the location of the sensor $s \in S$ is $\boldsymbol{s} \in \mathbb{R}^2$ and the node, and waypoint locations are $\boldsymbol{\nu}, \boldsymbol{p} \in \mathbb{R}^2$. The sensor, its location, and the waypoint location are also denoted as $\nu_i.s$, $\nu_i.\boldsymbol{s}$, and $\nu_i.\boldsymbol{p}_s$ to explicitly describe with which node they are associated to. The notation and the used symbols are summarized in Table 1.

Table 1 Notation and Symbols Used

S	Set of n sensors to be visited / covered
s	Sensor $s \in S$
$\boldsymbol{s}$	Location of the sensor s, $\boldsymbol{s} \in \mathbb{R}^2$
$\boldsymbol{s}_1$	Prescribed initial location, i.e., the sensor s_1
$\boldsymbol{s}_n$	Prescribed terminal location, i.e., the sensor s_n
r_i or $r(s_i)$	Reward r_i associated to the sensor s_i, $r_i \geq 0$
δ_i or $\delta(s_i)$	Sensing/communication range of the sensor $s_i \in S$
$\mathrm{T}_{\max}$	Given travel budget
S_k	Requested subset of sensors to be visited $S_k \subseteq S$
Σ	Requested order of visits to S_k
P	Sequence of waypoint locations $P \subset \mathbb{R}^2$
Θ	Headings at the waypoints for each $\theta \in \Theta$ and $\theta \in \mathbb{S}^1$
R	Sum of the rewards collected by visiting/covering S_k
$\mathcal{N}$	GSOA array (sequence) of nodes
M	Current number of nodes in $\mathcal{N}$, $M = \lvert\mathcal{N}\rvert$
ν_i	Node of the GSOA, $\nu_i \in \mathcal{N}$
ν_1	First node of $\mathcal{N}$, always corresponding to s_1
ν_M	Last node of $\mathcal{N}$, always corresponding to s_n
$\boldsymbol{\nu}_i$	Location of the node $\nu_i \in \mathcal{N}$, $\boldsymbol{\nu}_i \in \mathbb{R}^2$
$\nu.s$	Sensor $s \in S$ associated to $\nu \in \mathcal{N}$
$\nu.\boldsymbol{s}$	Location of the sensor $s \in S$ associated to $\nu \in \mathcal{N}$
$\nu.\boldsymbol{p}_s$	Waypoint location $\boldsymbol{p}_s \in \mathbb{R}^2$ associated to $\nu \in \mathcal{N}$
θ or $\nu.\theta$	Heading value associated to ν with the waypoint location $\nu.\boldsymbol{p}_s$ that forms the waypoint $\boldsymbol{q} = (\boldsymbol{p}_s, \theta)$ of ν
$\mathcal{L}(\mathcal{N})$	Length of Dubins path represented by $\mathcal{N}$
$\varsigma(\nu_i)$	Ratio computed as (5) associated to ν_i
$r(\nu_i.s)$	Reward of the sensor $s \in S$ associated to $\nu_i \in \mathcal{N}$
σ, μ, α	Parameters of the unsupervised learning: learning gain σ, learnig rate μ, and gain decreasing rate α
h	Number of supporting headings on one side of the Dubins maneuver defined by two consecutive nodes
l_{max}	Maximal number of winner nodes removed from $\mathcal{N}$ in benefit of new sensor to be visited
i_{max}	Maximal number of learning epochs

4.3 The GSOA Unsupervised Learning for CEDOP

The learning procedure always starts with two nodes ν_1 and ν_M that correspond to the prescribed initial and terminal locations. Thus ν_1 and ν_M are always in the array $\mathcal{N}$ and they are always associated with the locations $\boldsymbol{s}_1$ and $\boldsymbol{s}_n$. These nodes are never removed nor adapted during the learning to assure the initial and terminal locations are part of the solution. Besides, they are always considered as winner nodes regardless of learning epoch. Except for these two specific nodes, $\mathcal{N}$ is evolved during the learning and adapted towards the selected sensor locations. However, the requested data collection path has to respect the minimum turning radius ρ of Dubins vehicle (3). Therefore, the array $\mathcal{N}$ is conditionally adapted towards the particular sensor only if the winner nodes of the array represent Dubins path connecting the waypoints with the length $\mathcal{L}(\mathcal{N})$ not exceeding $\mathrm{T}_{\max}$. The Dubins path is determined as a solution of the DTP using the waypoint locations associated to the winners and set of supporting heading values associated to the nodes as it is shown in Fig. 1.

The solution of the DTP provides the most suitable heading values from the set of sampled headings associated with the winner nodes. The headings from the DTP solution are used as the new main headings of the nodes. The new main heading is also used to update the supporting headings around the main heading as follows. For each node, h heading values are determined in the interval $[0, \pi)$ on each side of a single Dubins maneuver defined by the waypoints associated to the corresponding nodes, see Fig. 2.[1] Thus, the total number of heading values (including the main heading) associated with the node (and its waypoint) is $n_h = 2h + 1$.

If $\mathrm{T}_{\max}$ is satisfied for Dubins path with a newly inserted waypoint of the current winner node, the array adaptation is performed as a movement of the winner node and its neighboring nodes in the array towards the waypoint location of the winner node. The solution of the problem is defined by the selected sensors (i.e., those associated with the winner nodes of the current learning epoch) and the corresponding waypoints. Besides, the solution progress and information about the searched part of the solution space is encoded in the distance of the winner nodes to their respective waypoints. The distance is further used to remove not promising sensors from the solution as a part of the ratio (5).

An overview of the complete GSAO for solving CEDOP is depicted in Algorithm 1. Detail description of the winner node determination is provided in Section 4.3.1 and the adaptation is described in Section 4.3.2 that is followed by a comment on the solution extraction in Section 4.3.3.

4.3.1 GSOA Winner Node Determination

For a currently considered sensor $s \in S \setminus \{s_1, s_n\}$, a new winner node ν^* together with the particular waypoint location $\boldsymbol{p}_s$ and also heading values are determined. The current winner nodes of $\mathcal{N}$ are utilized to construct Dubins path using the associated waypoints with the main heading found as a solution of the related DTP. Notice, the first node ν_1 and the last node ν_M are always the winner nodes because of the initial and terminal locations, see Fig. 2. Then, the closest point of Dubins path to the sensor location $\boldsymbol{s}$ is determined as the point $\boldsymbol{c}$. The Dubins path is a sequence of Dubins maneuvers, and each maneuver consists of a straight line segment and parts of a circle [4], and therefore, such a point $\boldsymbol{c}$ can be found analytically.

The particular waypoint location $\boldsymbol{p}_s$ is found as a point on the straight line segment $(\boldsymbol{c}, \boldsymbol{s})$ that is within the δ distance from $\boldsymbol{s}$

$$\boldsymbol{p}_s = \begin{cases} \boldsymbol{s} + (\boldsymbol{c} - \boldsymbol{s}) \frac{\delta - \epsilon}{\|(\boldsymbol{c}, \boldsymbol{s})\|} & \text{for } \|(\boldsymbol{c}, \boldsymbol{s})\| > \delta \\ \boldsymbol{c} & \text{otherwise} \end{cases}, \tag{6}$$

[1] Together with the determination of the main heading from Dubins path represented by the array $\mathcal{N}$, the supporting headings implement online sampling of the heading values during the learning.

Algorithm 1: Unsupervised learning of the Growing Self-Organizing Array (GSOA) for solving CEDOP

Input: $S = \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n\}$ – the locations of sensors to be covered, each sensor $s \in S$ has particular disk-shaped δ-neighborhood and reward $r_i \geq 0$
Input: $\boldsymbol{s}_1$, $\boldsymbol{s}_n$ – the prescribed initial and terminal locations with $\delta = 0$
Input: $\mathrm{T}_{\max}$ – the given limited travel budget
Input: (σ, μ, α) – the initial value of the learning gain $\sigma = 10$, the gain decreasing rate $\alpha = 0.0005$, and learning rate $\mu = 0.6$
Input: (h, l_{max}) – the number of supporting headings per each waypoint location (node) h at each side of Dubins path passing the waypoint and the maximal number of possible removed winner nodes l_{max} in benefit of added new winner node to $\mathcal{N}$, values $h = 3$ and $l_{max} = 3$ are utilized for the results reported in this paper
Input: i_{max} – the maximal number of learning epochs and $i_{max} = 150$ is used
Output: (Σ, S_k, P, Θ) – Σ defines the order of visits to the selected subset of the sensors $S_k \subseteq S$, P is the sequence of waypoint locations to visit the sensors S_k with the particular headings Θ

▷ *Initialization*
$\mathcal{N} \leftarrow \{\nu_1, \nu_M\}$ – initial and terminal locations of the array, $\nu_1.s = s_1$, $\nu_1.\boldsymbol{s}_p = \boldsymbol{s}_1$, $\nu_2.s = s_2$, and $\nu_2.\boldsymbol{s}_p = \boldsymbol{s}_n$. Initial h heading values are set around the heading corresponding to a straight line segment connecting $\boldsymbol{s}_1$ and $\boldsymbol{s}_n$. Notice the nodes ν_1 and ν_M are never adapted nor removed from $\mathcal{N}$ to assure the initial and terminal locations are part of the solution. `// it is assumed` $\mathrm{T}_{\max} \geq \|(\boldsymbol{s}_1, \boldsymbol{s}_n)\|$
$(\Sigma, S_k, P, \Theta) \leftarrow (\Sigma = (1, n), S_k = \{s_1, s_2\}, P = \{\boldsymbol{s}_1, \boldsymbol{s}_n\}, \Theta = (\theta_1, \theta_n)$, where θ_1 and θ_n corresponds to the segment $(\boldsymbol{s}_1, \boldsymbol{s}_n)$
$R \leftarrow 0$ `// set the initial sum of rewards to 0 because of` $r_1 = r_n = 0$
$i_{max} \leftarrow \min(i_{max}, 1/\alpha)$ `// ensure` σ `will always be above 0`
$i \leftarrow 0$ `// set the learning epoch counter`
while $i \leq i_{max}$ **do**
  ▷ *Learning epoch*
  foreach $\boldsymbol{s}$ *in a random permutation of* $S \setminus \{s_1, s_n\}$ **do**
    $\mathcal{N}' \leftarrow \mathcal{N}$ `// save the current array of nodes`
    $\mathcal{N}^i_{win} \leftarrow \{\nu \in \mathcal{N} \setminus \{\nu_1, \nu_M\} : \nu$ is a winner node in the i-th epoch and $r(\nu_s) > r(s)\}$ `// the current winner nodes`
    $\nu^*(\boldsymbol{\nu}^*, \nu^*.\boldsymbol{s}_p, \theta) \leftarrow \mathrm{determine\_winner}(\mathcal{N}, s, \delta_s)$ `// Determine winner node according to Fig. 4`
    $\mathcal{N} \leftarrow \mathrm{insert\_winner}(\mathcal{N}, \nu^*)$
    ▷ *Check the travel budget* T_{max}
    $l \leftarrow 1$
    while $\mathcal{L}(\mathcal{N}) > T_{max}$ *and* $l \leq l_{max}$ **do**
      $\nu_{min} \leftarrow \mathrm{argmin}_{\nu_i \in \mathcal{N}^i_{win}} \varsigma(\nu_i)$ `// select the node with the lowest ratio` ς_i `according to (5)`
      $\mathcal{N}^i_{win} \leftarrow \mathcal{N}^i_{win} \setminus \{\nu_{min}\}$
      $\mathcal{N} \leftarrow \mathcal{N} \setminus \{\nu_{min}\}$
      $l \leftarrow l + 1$
    ▷ *Conditional adapt*
    if $\mathcal{L}(\mathcal{N}) \leq T_{max}$ **then**
      foreach *node* $\nu \in \mathcal{N}$ *in the* d*-neighborhood of the node* ν^* *such that* $0 \leq d \leq 0.2M$ **do**
        Adapt ν towards $\nu^*.\boldsymbol{s}_p$ using (7) and the neighbouring nodes that have been added to the array in the current epoch are also adapted using (8); both adaptations with the neighbouring function (9)
    else
      $\mathcal{N} \leftarrow \mathcal{N}'$ `// Revert the changes to the array and do not adapt towards` s
  ▷ *Update the best solution found so far*
  $\mathcal{N} \leftarrow \{\nu \in \mathcal{N} : \nu$ is the winner node of the current epoch $i\}$ `// Remove all not winning nodes from` $\mathcal{N}$
  $\sigma \leftarrow (1 - i\alpha)\sigma$ `// decrease the learning gain`
  $i \leftarrow i + 1$ `// update the epoch counter`
  Traverse the array $\mathcal{N}$, extract the selected sensors S'_k and construct the sequence of visits Σ' with the corresponding waypoint locations P' and the best heading values Θ' (regarding the solution of the associated DTP as in Fig. 1)
  $R' \leftarrow \sum_{s_i \in S'_k} r(s_i)$ `// Compute the sum of the collected rewards`
  if $R' \leq R$ **then**
    $(\Sigma, S_k, P, \Theta) \leftarrow (\Sigma', S'_k, P', \Theta')$ `// update the best solution found so far`
return (Σ, S_k, P, Θ)

where a small constant ϵ, e.g., $\epsilon = 0.001$, is subtracted from δ to avoid possible numerical imprecisions. The heading θ of Dubins vehicle on Dubins path at the location $\boldsymbol{c}$ is used as the main heading of the newly determined waypoint $(\boldsymbol{p}_s, \theta)$ and $2h$ additional supporting heading values are sampled around θ. The winner node ν^* is then inserted between the nodes of $\mathcal{N}$ that correspond to the particular Dubins maneuver on which $\boldsymbol{c}$ has been determined. However, the location of the winner node $\boldsymbol{\nu}^*$ is set as the closest point of the segment represented by the two consecutive nodes, and thus the distance $\|(\boldsymbol{\nu}^*, \boldsymbol{p}_s)\|$ can be considered as a "learning error" that is being minimized, but it is mainly utilized in determining the nodes (sensors) to be removed from the solution to fit $\mathrm{T}_{\max}$ using the ratio (5).

An example of the first winner node determination for the configuration depicted in Fig. 2 is visualized in Fig. 3.

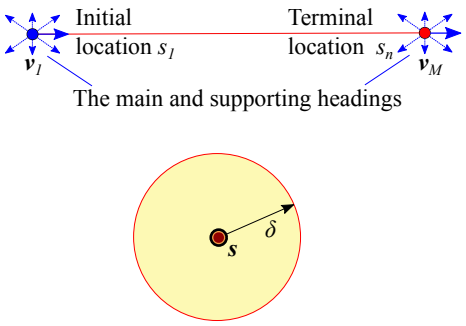


Fig. 2 Example of the initial Dubins path represented by ν_1 and ν_M that correspond to the initial and terminal locations $\boldsymbol{s}_1$ and $\boldsymbol{s}_n$, respectively. The main headings of ν_1 and ν_M are defined from the straight line segment $(\boldsymbol{s}_1, \boldsymbol{s}_n)$ and $h = 3$ supporting heading values are placed on each side of Dubins maneuver.

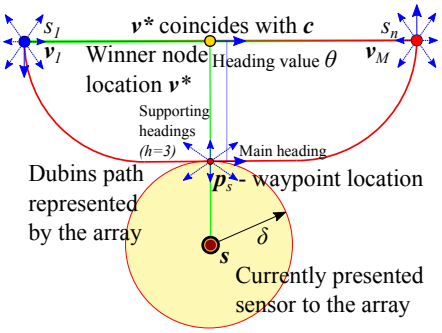


Fig. 3 Example of the winner node determination for $\mathcal{N} = (\nu_1, \nu_M)$. Because only two nodes are in $\mathcal{N}$, Dubins path represented by $\mathcal{N}$ is a straight line segment $(\boldsymbol{s}_1, \boldsymbol{s}_n)$ and the closest point $\boldsymbol{c}$ of Dubins path to $\boldsymbol{s}$ is coincident with the winner node location $\boldsymbol{\nu}^*$. Once the winner node ν^* is inserted into $\mathcal{N}$ together with the determined waypoint location $\boldsymbol{p}_s$ and corresponding heading values, the array of nodes $\mathcal{N}$ represents a new Dubins path connecting $\boldsymbol{s}_1$ with $\boldsymbol{s}_n$ over the waypoint $(\boldsymbol{p}_s, \theta)$.

Dubins path represented by $\mathcal{N}$ is a straight line segment connecting $\boldsymbol{s}_1$ and $\boldsymbol{s}_n$ and the closest point $\boldsymbol{c}$ of Dubins path is coincident with the winner node location $\boldsymbol{\nu}^*$. The winner node ν^* is inserted between the ν_1 and ν_M nodes as they are the only nodes of the array $\mathcal{N}$. In this particular case, the heading value θ determined as the heading at $\boldsymbol{c}$ fits the shortest Dubins path connecting $\boldsymbol{p}_s$, but in general, another heading value from the supporting headings can be more suitable for the shortest Dubins path.

A little bit more complex situation for the array with three nodes $\mathcal{N} = (\nu_1, \nu_2, \nu_M)$ is visualized in Fig. 4. In this case, the node ν_2 is already associated with its waypoint location $\nu_2.\boldsymbol{p}_s$ to cover the sensor $\nu_2.s$ and Dubins path visiting the locations $\boldsymbol{s}_1$, $\nu_2.\boldsymbol{p}_s$, and $\boldsymbol{s}_n$ consists of two Dubins maneuvers. The closest point $\boldsymbol{c}$ of Dubins path does not coincide with the winner node location $\boldsymbol{\nu}^*$ that is the closest point of the straight line segment $(\boldsymbol{\nu}_2, \boldsymbol{\nu}_M)$ to the sensor location $\boldsymbol{s}$. The new Dubins path visiting the locations $\boldsymbol{s}_1$, $\nu_2.\boldsymbol{p}_s$ with the newly determined waypoint location $\nu^*.\boldsymbol{p}_s$ and the terminal location $\boldsymbol{s}_n$ is determined as the solution of the DTP (see Fig. 1) using the heading value θ with the additional supporting heading values. Also, in this case, the

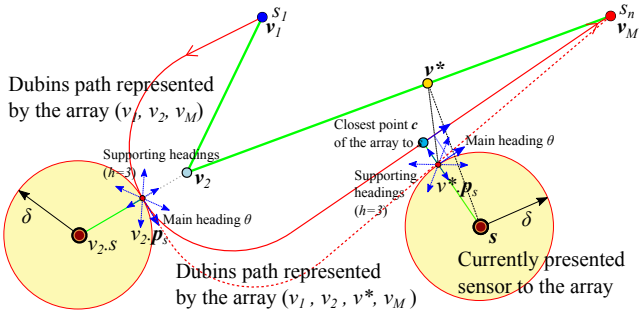


Fig. 4 Example of the winner node determination for the array with three nodes $\mathcal{N} = (\nu_1, \nu_2, \nu_M)$. Dubins path represented by $\mathcal{N}$ consists of two Dubins maneuvers connecting $\boldsymbol{s}_1$ with the waypoint location $\nu_2.\boldsymbol{p}_s$ and further with $\boldsymbol{s}_n$. The closest point $\boldsymbol{c}$ of the array to $\boldsymbol{s}$ is not coincident with the winner node location $\boldsymbol{\nu}^*$ that is the closest point of the segment defined by two consecutive nodes to $\boldsymbol{s}$, which is the segment $(\boldsymbol{\nu}_2, \boldsymbol{\nu}_M)$ in this case. Dubins path visiting the waypoint $\nu^*.\boldsymbol{p}_s$ is visualized as dashed red curve.

main heading θ seems to provide the shortest Dubins path connecting $\boldsymbol{p}_s$. If the length of the determined Dubins path does not exceed $\mathrm{T}_{\max}$, e.g., eventually after removing up to l_{max} winner nodes, the new winner node ν^* is adapted towards $\nu^*.\boldsymbol{p}_s$ to decrease the distance of the winner node to its waypoint.

4.3.2 GSOA Adaptation of the Winner Node

Contrary to the winner node determination where Dubins path connecting the waypoints is considered, the adaptation is performed in the Euclidean space similarly as in the SOM-based solution of the TSP [15] or solution of the OP [23]. The main intention of the adaptation is to "learn a preference" of selecting possible highly rewarding sensors. The distance between the node location $\boldsymbol{\nu}$ to the location of its waypoint $\nu.\boldsymbol{p}_s$ is used in removing nodes (Lines 12–17, Algorithm 1) using the ratio (5). Because the ratio depends on the distance, and the power of the adaptation corresponds to the reward associated with the sensors, the adaptation is stronger for highly rewarding sensors.

For a new winner node ν^*, the adaptation of the array $\mathcal{N}$ is a movement of the locations of the winner node ν^* and its neighboring nodes towards the new waypoint location $\nu^*.\boldsymbol{p}_s$ associated with ν^* that can be expressed as

$$\boldsymbol{\nu}' = \boldsymbol{\nu} + R_s \mu f(\sigma, d)(\nu^*.\boldsymbol{p}_s - \boldsymbol{\nu}), \tag{7}$$

where μ is the learning rate ($\mu = 0.6$), R_s is the ratio of the reward $r(s)$ of the sensor s and the maximal reward of the sensors in the set S, σ is the learning rate, and d is the distance (in the number of nodes) of the node ν from the winner node ν^* in the array $\mathcal{N}$. In addition to controlling the power of adaptation by the value of R_s, the neighbouring nodes added to the array in the current epoch are also adapted towards their waypoint locations in the herein proposed GSOA-based solution of the OP. The adaptation of the

neighboring node ν can be expressed as

$$\boldsymbol{\nu}'' = \boldsymbol{\nu}' + R_{\nu.s}\mu f(\sigma, d)(\nu.\boldsymbol{p}_s - \boldsymbol{\nu}'), \tag{8}$$

where d has the same meaning as in (7), but the neighboring node ν adjusted towards $\nu^*.\boldsymbol{p}_s$ by (7) is adapted towards its waypoint location $\nu.\boldsymbol{p}_s$ using $R_{\nu.s}$ computed from the reward $r(\nu.s)$ of the sensors $\nu.s$ associated to the neighboring node ν of the new node ν^*. Hence, the node ν first moves away from its waypoint location to the position $\boldsymbol{\nu}'$ and then a bit back to the position $\boldsymbol{\nu}''$. This way, the distance of the nodes associated with highly rewarding sensors is less prolonged than the distance of the nodes associated with low rewarding sensors, which can be then more likely removed because of a low value of (5). The same neighboring function used in the unsupervised learning of SOM is utilized in the GSOA

$$f(\sigma, d) = \begin{cases} e^{\frac{-d^2}{\sigma^2}} & \text{for } d < 0.2M \\ 0 & \text{otherwise} \end{cases}, \tag{9}$$

which decreases the power of adaptation of the neighbouring nodes to the winner neuron ν^* with increasing distance d.

The adaptation is performed only if the current winner nodes of $\mathcal{N}$ represent Dubins path which length is shorter or equal to $\mathrm{T_{max}}$ (Line 18, Algorithm 1). Otherwise, the array of nodes $\mathcal{N}$ is reverted to the state before the winner selection (Line 22, Algorithm 1). The winner node determination and its conditional adaptation are repeated for every sensor in S. After considering all the sensors, the learning epoch terminates, and a solution represented by the winner nodes in $\mathcal{N}$ is extracted.

4.3.3 Solution Extraction

At the end of each learning epoch, all nodes that have not been added to $\mathcal{N}$ in the current epoch are removed, except the persistent ν_1 and ν_M. Since each node has associated waypoint together with possible heading values, the array is sequentially traversed and Dubins path to visit the waypoint locations is determined as a solution of the DTP. If the current new found solution represents a feasible path with a higher sum of the collected rewards R' than the current R, the best solution found so far is updated (Line 29, Algorithm 1).

The unsupervised learning procedure is repeated for the i_{max} learning epochs. The winner nodes from the previous epoch are preserved. Hence Dubins maneuvers connecting waypoints that are associated with the two consecutive nodes in the array $\mathcal{N}$ are utilized for determination of the closest point of Dubins path represented by $\mathcal{N}$ (its associated waypoints). On the other hand, the data collection path as a solution of CEDOP that has to satisfy $\mathrm{T_{max}}$ is constructed using the new winners of the current epoch (including the persistent winners ν_1 and ν_M. Thus the unsupervised learning can be considered as a stochastic search that is biased by the preference of the highly rewarding sensors, and the main role of the learning is in decreasing the distance of the winner node locations to their respective waypoint locations.

An example of the GSOA evolution in the solution of CEDOP with individual sensing range per each particular sensor is visualized in Fig.5.

4.4 Computational Complexity

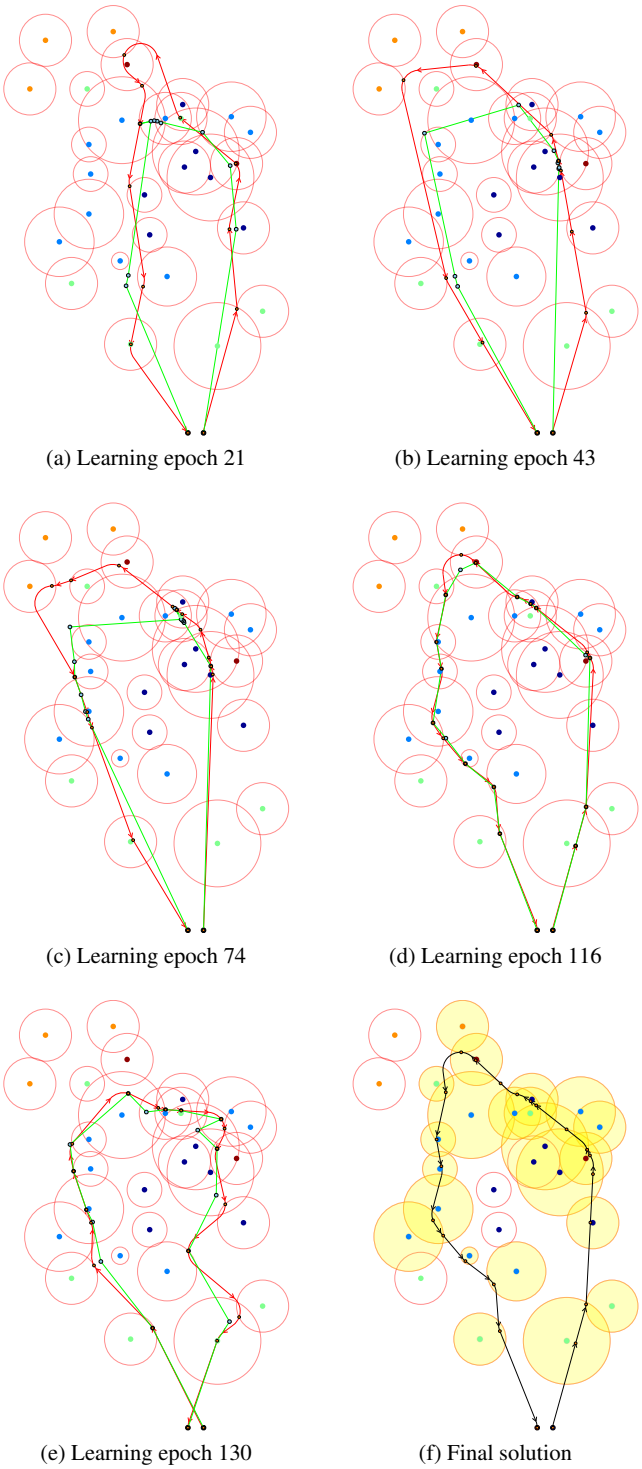
The computational complexity of the proposed GSOA unsupervised learning procedure for solving CEDOP depends on the number of sensors S and supporting heading values h because of the solution of the DTP. For n sensors, the number of nodes in the array $\mathcal{N}$ can be bounded by $2n$ as n winner nodes are preserved for the next learning epoch, where up to n new nodes can be determined. In a single adaptation of $\mathcal{N}$ to the sensor $s \in S$, Dubins path represented by the array of nodes is found to determine the winner node and its waypoint location. Since the number of headings is $2h + 1$ and the maximal number of nodes in $\mathcal{N}$ is $2n$, the winner node determination can be bounded by $O(nh^2)$ because of solving the DTP.

For each considered sensor, up to l_{max} nodes (sensors) can be removed from the current $\mathcal{N}$ to reduce the Dubins path length. The nodes are removed in the order of increasing ratio $\varsigma(\nu_i)$ (5) which can be maintained in $O(\log n)$ using a binary heap. After that, the adaptation is performed as a movement of the winner node and its neighboring nodes towards the waypoint location that can be bounded by $O(n)$ as the number of moved nodes never exceeds $2n$. A single consideration of sensor s can be bounded by $O(nh^2 + \log n + 2n)$. For n sensors, the computational complexity of a single learning epoch can be thus bounded by $O(n^2h^2)$, which is performed i_{max} times.

The unsupervised learning does not depend on the sensing range δ because the waypoints are determined during the winner node selection. Besides, relatively few supporting headings are considered, which is in the direct contrast with the VNS-based approach [10] where possible waypoint locations are sampled, and each sample is further considered for sampling possible heading values. Therefore, the problem addressed by an explicit discretization quickly grows and it becomes computationally very demanding.

4.5 Solution of CEDOP with a Team of Vehicles

Similarly to the SOM-based solution of the OP for several vehicles [37] or the Team Orienteering Problem reported in [23]; also the proposed GSOA for CEDOP can be utilized for a solution with a team of data collection vehicles.



(a) Learning epoch 21 (b) Learning epoch 43

(c) Learning epoch 74 (d) Learning epoch 116

(e) Learning epoch 130 (f) Final solution

Fig. 5 Example of the evolution of the GSOA in a solution of the CEDOP instance with individual sensing ranges δ_i and the turning radius of Dubins vehicles $\rho = 1$. The number of learning epoch is $i_{max} = 150$, but the final solution is determined at the end of the epoch number 116. The visited sensors are highlighted as yellow disk denoting the individual sensing range. The red curve is the current Dubins path satisfying the limited T_{max}. The green curve represents connected locations of the nodes $\mathcal{N}$. Small colored disks are the sensor locations, where the color denotes the reward value (red is high and low is blue). The travel budget is T_{max}=50, and the final solution length is 49.962. The sum of the rewards collected by the found Dubins path is $R = 530$ from the total sum of rewards 670.

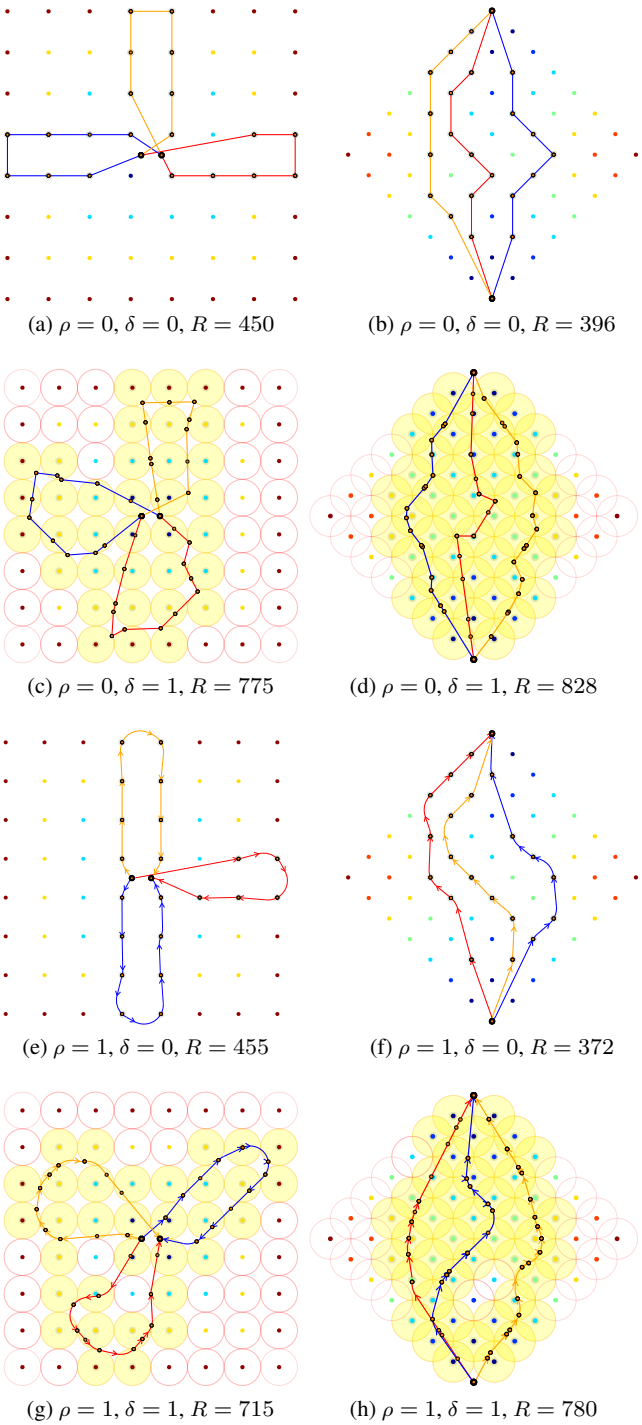
The main idea is to create an individual array of nodes for each vehicle. Then the most suitable array for the winner insertion and adaptation is determined according to the maximum gained reward per array. Detail description is out of the scope of this paper, and therefore, only an example of solutions are depicted in Fig. 6 to demonstrate the GSOA can be utilized for the solution of CEDOP with a team of vehicles.

5 Results

The proposed GSOA for CEDOP has been evaluated using the OP benchmarks [20,40] that have been utilized in the evaluation of DOP [9] and in early results on SOM-based unsupervised learning approaches [14,11]. The only existing solvers for DOP and CEDOP are the combinatorial optimization VNS based on a discretization of the continuous parts of the problem [9,10] and the SOM-based approach [14,11]. Although the SOM-based solution of the regular OP and CEOP have been evaluated in [7,23], the herein proposed approach is based on novel unsupervised learning of the GSOA [15] that is designed for routing problems motivated by data collection mission with non-zero sensing/communication range. Thus, the proposed GSOA for CEDOP is also compared with the existing SOM approaches for the Euclidean OP and CEOP instances in the standard OP benchmarks to provide a complete overview of the performance of the unsupervised learning methods.

The benchmarks for the OP consists of a set of problems with particular locations, each with the associated reward, and accompanied with a set of specific values of the travel budget T_{max} [40]. The considered benchmarks are Set 1, Set 2, and Set 3 proposed by Tsiligirides [41] and diamond-shaped Set 64 and squared-shaped Set 66 [42]. Each problem set is accompanied with budget values in the range 5 to 130 which gives 89 instances of the OP. Each such an instance can be further specified by the minimum turning radius ρ of Dubins vehicle that is considered from the set $\rho \in \{0, 0.5, 1.0, 1.5, 2.0\}$, which can give 445 instances of DOP. Moreover, for non-zero sensing range, additional instances can be defined for the CEOP and DOP with $\delta > 0$.

Due to such an excessive number of instances, the evaluation for the CEOP, DOP, and CEDOP are made for the selected instances to focus on the influence of ρ and δ. Therefore, particular results are presented for instances with a single value of the travel budget T_{max} that is selected to be in the middle of the T_{max} range specified for the particular problem set. Such a budget provides challenges to balance the solution quality of the routing part (to have the shortest path possible) with the selection of the subset S_k with the most rewarding sensors. In addition, increasing the sensing range δ allows to visit more sensors, and for a certain value, all sensors can be covered even for relatively small budget T_{max}.



(a) $\rho = 0$, $\delta = 0$, $R = 450$ (b) $\rho = 0$, $\delta = 0$, $R = 396$

(c) $\rho = 0$, $\delta = 1$, $R = 775$ (d) $\rho = 0$, $\delta = 1$, $R = 828$

(e) $\rho = 1$, $\delta = 0$, $R = 455$ (f) $\rho = 1$, $\delta = 0$, $R = 372$

(g) $\rho = 1$, $\delta = 1$, $R = 715$ (h) $\rho = 1$, $\delta = 1$, $R = 780$

Fig. 6 Found solutions for team orienteering problems [40] with turning radius $\rho \in \{0, 1\}$, and sensing range $\delta \in = \{0, 1\}$. (left): the problem instance p5.3.g with three vehicles, $T_{max} = 17.4$, and the total sum of rewards 1680; (right): the problem instance p6.4.k with four vehicles, $T_{max} = 16.2$, and the total sum of rewards 1344.

The particular problem instances with the selected T_{max} are listed in Table 2.

Both the VNS-based and unsupervised learning approaches are randomized. Therefore each problem instance is solved 20 times, and the reported performance indicators are the average sum of the collected rewards R, the average required computational time T_{cpu}, and the best-found solution from 20 trials is reported as R_{max}. Besides, the overall performance of the algorithms as an aggregated value is evaluated using the relative percentage error (RPE) defined as the relative error between the reference value R_{ref} and R_{max} over the performed trials per particular problem instance, where R_{ref} is the highest reward found by the evaluated algorithms. The RPE is computed as

Table 2 Selected instances of the OP

Set 1 – Instance with 32 locations and $T_{max} = 46$
Set 2 – Instance with 21 locations and $T_{max} = 30$
Set 3 – Instance with 33 locations and $T_{max} = 50$
Set 64 – Instance with 64 locations and $T_{max} = 45$
Set 66 – Instance with 66 locations and $T_{max} = 60$

$$\text{RPE} = (R_{ref} - \text{R}_{max})/R_{ref} \cdot 100\%. \quad (10)$$

The robustness of the algorithm over the performed trials is shown as the average relative percentage error ARPE

$$\text{ARPE} = (R_{ref} - \text{R})/R_{ref} \cdot 100\%. \quad (11)$$

The average values of the respective indicators RPE and ARPE among all evaluated instances in the particular benchmark sets are denoted by $\overline{\text{RPE}}$ and $\overline{\text{ARPE}}$, respectively, to report overall expected performance of the methods in the particular problem sets.

The recommended settings of the VNS-based solvers [9, 10] is utilized, and the number of heading values in solving DOP instances is $n_h = 13$. For CEDOP, the number of heading samples is $n_h = 8$ per each of the waypoint location sample that are sampled at the δ perimeter around each sensor location, and $n_w = 8$ samples are utilized, which gives 64 possible waypoints per each sensor $s \in S$ for the VNS-based approach. The utilized stopping criterion is the maximal number of 10 000 iterations with the maximal 5 000 iterations without improvement. Only two parameters are relevant for the SOM-based algorithms: the number of learning epochs i_{max} and the number of additional supporting headings h. Regarding the results reported in [14, 11] the following settings is considered: $i_{max} = 150$ and $h = 3$. The same values are also used for the proposed GSOA algorithm and with one more parameter l_{max}, which denotes the maximal number of the eventually removed winner nodes, which has been set to $l_{max} = 3$. Due to the way how the supporting headings are constructed in SOM and GSOA approaches, the value of $h = 3$ means $n_h = 7$ heading values per each waypoint locations for both types of unsupervised learning approaches.

All the algorithms (VNS, SOM, and GSOA) have been implemented in C++ and run within the same computational environment using a single core of the Intel Core i7-6700K CPU running at 4 GHz, except the computationally demanding VNS-based solution of CEDOP, which has been run on a

Table 3 Aggregated results for all the instances of the Orienteering Problem

Set	CGW [42]	SOMv1 [7]		SOMv2 [23]			**Proposed GSOA**		
	$\overline{\text{RPE}}$	$\overline{\text{RPE}}$	$\overline{\text{ARPE}}$	$\overline{\text{RPE}}$	$\overline{\text{ARPE}}$	Speed ×	$\overline{\text{RPE}}$	$\overline{\text{ARPE}}$	Speed ×
Set 1	0.10	0.25	2.45	**0.10**	**1.05**	2.1	0.36	2.80	2.4
Set 2	0.92	**0.92**	1.94	**0.92**	**1.10**	7.4	**0.92**	2.56	7.7
Set 3	0.00	**0.00**	1.73	**0.00**	**0.89**	13.5	0.33	3.24	24.2
Set 64	0.40	2.83	6.32	**1.62**	**4.37**	11.4	1.74	4.64	15.0
Set 66	0.43	2.10	6.14	1.54	5.27	20.4	**0.80**	**2.26**	26.9

computational grid. The VNS-based algorithm pre-computes all Dubins maneuvers between all the waypoints prior the combinatorial optimization with an optimized structure for inserting/removing waypoints into Dubins tour. On the other hand, both the SOM and GSOA approaches are searching for the suitable headings during the unsupervised learning, and all the requested Dubins maneuvers are computed on demand and no special optimizations in Dubins path computation are utilized for the presented results.

The achieved results and comparison for the particular OP, CEOP, DOP, and CEDOP instances are reported in the following sections together with the discussion of the results and performance of the proposed GSOA-based method.

5.1 Empirical Evaluation in the OP and CEOP Instances

A performance overview of the proposed GSOA in solving the regular OP (considering the full range of T_{max} for each problem set) is reported in Table 3, where the column *Speed* × denotes how much is the particular unsupervised learning approach faster than the first SOM-based solution to the OP introduced in [7]. In absolute numbers, a solution is found in tens of milliseconds, and the most demanding instances are from Set 64 and Set 66 which are found in around 80 ms for the utilized computational environment. Example of solutions found by the proposed GSOA for the selected problems according to Table 2 are depicted in Fig. 7.

Two SOM-based approaches denoted SOMv1 [7] and SOMv2 [23] are compared with the proposed GSOA. The best results provided by the unsupervised learning approaches are highlighted in bold. The results indicate that the best performing unsupervised learning is SOMv2 [23] that is together with the GSOA significantly faster than early SOMv1. The proposed GSOA provides overall a bit worse solutions than SOMv2, but significantly better results are provided for the large instances from Set 66. On the other hand, none of the unsupervised learning approaches provides better results than one of the first heuristics for the OP denoted CGW [42].

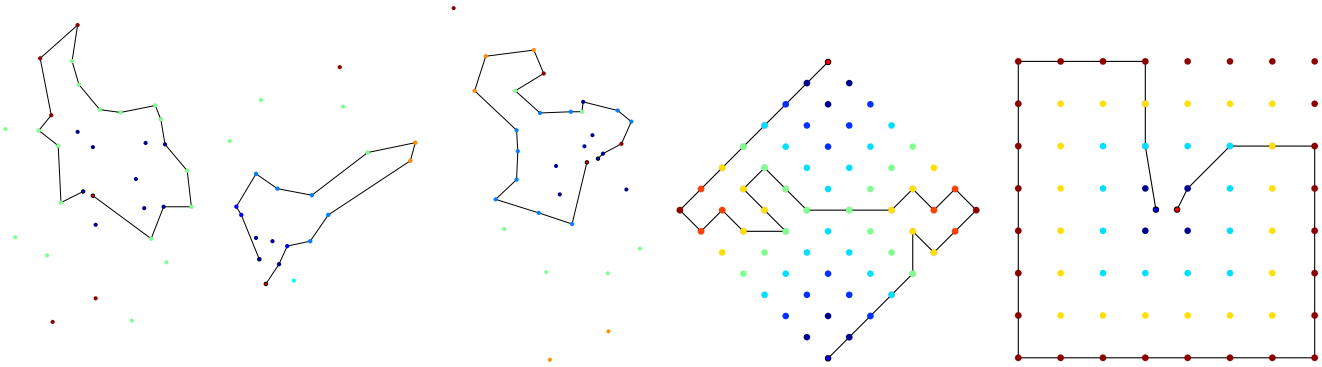
The main benefit of the unsupervised learning approaches is in exploiting non-zero sensing range δ. The performance indicators for the selected instances of Set 3, Set 64, and Set 66 problem sets with T_{max} according to Table 2 and $\delta \in \{0.5, 1.0, 1.5, 2.0\}$ are reported in Table 4 with the overview of the computational requirements in Fig. 9. It can be noticed that considering $\delta \geq 1.5$ for Set 64 allows to collect all the rewards as increasing δ does not increase R_{max}. The proposed GSOA-based solver outperforms the previous SOM-based approaches almost in all evaluated instances. The most demanding is the early approach [7], where the rewards are addressed by duplicating the sensors, which has been addressed in [23] by weighted adaptation, which is also employed in the proposed GSOA. Overall, the best performing algorithm is the GSOA and examples of found solution for selected sensing ranges, which do not allow to collect all the rewards, are depicted in Fig. 8.

Table 4 Results for the Close Enough Orienteering Problem

Problem	SOMv1 [7]		SOMv2 [23]		**Proposed**	
(T_{max}, δ)	R_{max}	R	R_{max}	R	R_{max}	R
Set 3 (50, 0.5)	560	518	460	415	**570**	**544**
Set 3 (50, 1.0)	620	591	600	535	**630**	**608**
Set 3 (50, 1.5)	**690**	650	650	627	670	**663**
Set 3 (50, 2.0)	710	689	700	683	**720**	**706**
Set 64 (45, 0.5)	1026	952	1026	947	**1038**	**999**
Set 64 (45, 1.0)	1296	1247	**1344**	**1332**	**1344**	**1332**
Set 64 (45, 1.5)	**1344**	1340	**1344**	**1344**	**1344**	**1344**
Set 64 (45, 2.0)	**1344**	1341	**1344**	**1344**	**1344**	**1344**
Set 66 (60, 0.5)	985	901	765	626	**1010**	**982**
Set 66 (60, 1.0)	1300	1233	**1495**	**1462**	1480	1436
Set 66 (60, 1.5)	1585	1543	**1650**	**1628**	**1650**	1624
Set 66 (60, 2.0)	1665	1641	**1680**	**1679**	**1680**	1678

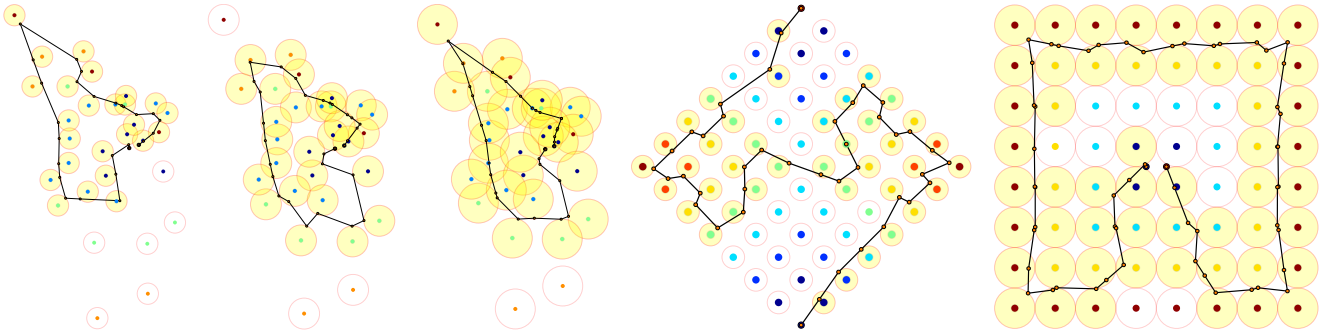
5.2 Empirical Evaluation in the DOP and CEDOP Instances

Performance of the proposed and existing solvers in orienteering problems with Dubins vehicle is firstly evaluated for the selected DOP instances. The only existing solvers are denoted VNS-DOP [9] and SOM-DOP [14]. The results are reported in Table 5, where the best performing unsupervised



(a) Set 1, T_{max}= 46, R= 175 (b) Set 2, T_{max}= 30, R= 265 (c) Set 3, T_{max}= 50, R= 500 (d) Set 64, T_{max}= 45, R=780 (e) Set 66, T_{max}= 60, R=890

Fig. 7 Solutions of the selected OP instances found by the GSOA.



(a) Set 3, T_{max}= 50, $\delta = 1.0$, R = 620 (b) Set 3, T_{max}= 50, $\delta = 1.5$, R = 670 (c) Set 3, T_{max}= 50, $\delta = 2.0$, R = 720 (d) Set 64, T_{max}= 45, $\delta = 0.5$, R = 1020 (e) Set 66, T_{max}= 60, $\delta = 1.0$, R = 1475

Fig. 8 Selected solutions of the CEOP instances found by the proposed GSOA.

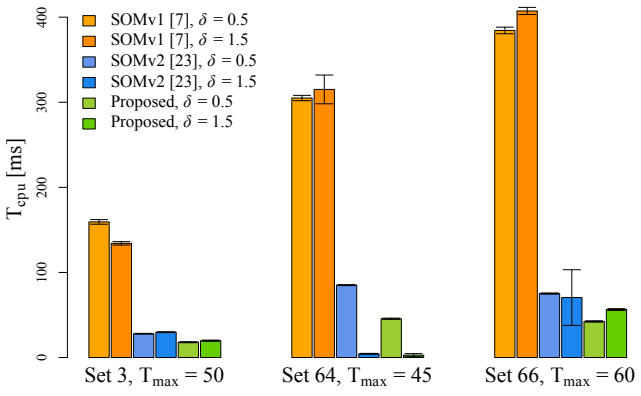


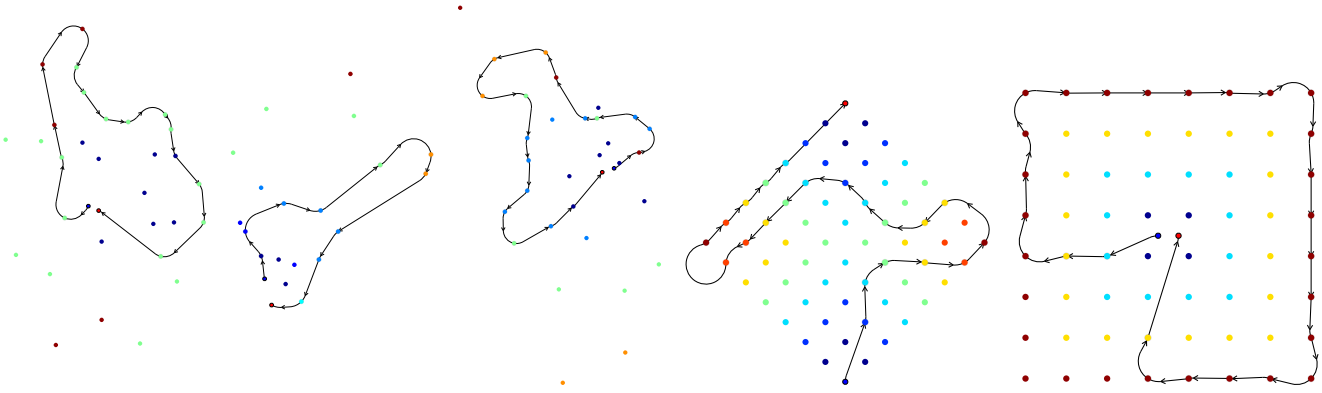
Fig. 9 Computational requirements in solution of the CEOP instances.

learning approach in particular problem instance is highlighted in bold. Notice that for increased minimum turning radius ρ, solutions with a lower sum of the collected rewards are found than for shorter ρ. It is because, for longer radii, a relatively long Dubins maneuver is required to connect two close sensors, e.g., see the solution of Set 64 in Fig. 10.

The proposed GSOA-based approach provides noticeably better results than SOM-DOP [14] and it also requires a bit lower computational requirements. In general, the determination of Dubins maneuvers is significantly more demanding than using Euclidean distance in the regular OP, where the solutions are found in tens of milliseconds contrary to units and tens of seconds for DOP. Overall, all the evaluated algorithms provide competitive solutions, but in absolute numbers, the best solutions are found by the VNS-DOP [9], and the unsupervised learning approaches provide a bit worse solutions. Notice that a choice to cover one sensor instead of another can make the difference in the sum of rewards about 5, 10, or even more, as the solution is sensitive to a proper selection of the subset S_k. The main differences in the solution quality are for the Set 64 instances, where sensors are placed in a diamond-shaped grid and the unsupervised learning methods stuck at local optima probably because of the limited number of supporting headings.

Table 5 Results for the Dubins Orienteering Problem

Problem instance (T_{max}, ρ)	VNS-DOP [9]			SOM-DOP [14]			**Proposed GSOA**		
	R_{max}	R	T_{cpu} [s]	R_{max}	R	T_{cpu} [s]	R_{max}	R	T_{cpu} [s]
Set 1 (T_{max}= 46, $\rho = 0.5$)	175	168	11.0	170	**168**	5.1	**175**	164	**4.3**
Set 1 (T_{max}= 46, $\rho = 1.0$)	165	158	11.3	**160**	**153**	6.3	**160**	149	**4.6**
Set 1 (T_{max}= 46, $\rho = 1.5$)	140	135	10.2	**140**	126	5.0	135	**128**	**4.3**
Set 2 (T_{max}= 30, $\rho = 0.5$)	255	252	4.4	**255**	**249**	**2.3**	**255**	245	**2.3**
Set 2 (T_{max}= 30, $\rho = 1.0$)	230	230	4.8	**240**	221	2.9	**240**	**224**	**2.3**
Set 2 (T_{max}= 30, $\rho = 1.5$)	210	210	3.6	195	181	2.2	**210**	**197**	**2.1**
Set 3 (T_{max}= 50, $\rho = 0.5$)	510	508	14.1	**510**	**492**	6.1	**510**	482	**5.1**
Set 3 (T_{max}= 50, $\rho = 1.0$)	470	470	13.6	**480**	**464**	6.8	**480**	454	**5.2**
Set 3 (T_{max}= 50, $\rho = 1.5$)	440	430	12.4	**440**	**403**	6.1	430	392	**4.9**
Set 64 (T_{max}= 45, $\rho = 0.5$)	792	778	61.2	**744**	**697**	24.5	714	692	**21.9**
Set 64 (T_{max}= 45, $\rho = 1.0$)	702	684	59.6	540	513	19.9	**570**	**537**	**19.2**
Set 64 (T_{max}= 45, $\rho = 1.5$)	636	603	55.3	456	413	**14.5**	**504**	**432**	15.6
Set 66 (T_{max}= 60, $\rho = 0.5$)	895	850	63.2	860	814	22.9	**895**	**867**	**18.5**
Set 66 (T_{max}= 60, $\rho = 1.0$)	890	846	71.9	835	790	22.2	**870**	**840**	**19.3**
Set 66 (T_{max}= 60, $\rho = 1.5$)	795	722	61.5	765	698	18.0	**780**	**730**	**16.3**



(a) Set 1, T_{max}= 46, $\rho = 1.0$, R= 160 (b) Set 2, T_{max}= 30, $\rho = 1.0$, R= 240 (c) Set 3, T_{max}= 50, $\rho = 1.0$, R= 480 (d) Set 64, T_{max}= 45, $\rho = 1.0$, R= 570 (e) Set 66, T_{max}= 60, $\rho = 1.0$, R= 870

Fig. 10 Selected solutions of the DOP instances found by the proposed GSOA.

Table 6 Aggregated Results for the Close Enough Dubins Orienteering Problem

Problem set $\rho = 1.0$	VNS			SOM		**Proposed GSOA**		
	$\overline{ARPE}$	T_{cpu} [s]	$\overline{RPE}$	$\overline{ARPE}$	T_{cpu} [s]	$\overline{RPE}$	$\overline{ARPE}$	T_{cpu} [s]
Set 3, $\delta = 0.5$	0.4	2,887.77	6.9	10.9	7.9	**4.1**	**6.8**	**5.0**
Set 3, $\delta = 1.0$	0.5	3,321.15	7.0	10.6	8.3	**3.7**	**6.7**	**4.6**
Set 64, $\delta = 0.5$	2.7	3,488.85	22.0	27.3	**20.2**	**17.4**	**22.6**	20.7
Set 64, $\delta = 1.0$	0.4	3,600.06	10.4	15.7	**22.2**	**7.9**	**12.8**	23.2
Set 66, $\delta = 0.5$	1.7	3,332.26	13.0	17.9	25.5	**10.9**	**14.0**	**22.4**
Set 66, $\delta = 1.0$	1.4	3,512.28	17.2	22.3	26.7	**14.3**	**18.5**	**24.0**

On the other hand, the relatively small problem sets Set 1, and Set 2 do not represent a significant challenge for the unsupervised learning approaches, and therefore, they are not

considered in the evolution of the solvers in the CEDOP instances.

The turning radius $\rho = 1$ is considered in the performance evaluation of the CEDOP instances, but with the full budget ranges for the instances from Set 3, Set 64, and Set 66 and the sensing range $\delta \in \{0.5, 1.0\}$. The aggregated results are reported in Table 6, where the reference solution R_{ref} used in computation (10) and (11) is the solution found by the VNS-based approach, and thus $\overline{\text{RPE}}$ is zero for it. The VNS-based method provides the best solutions but it is computationally very demanding, and solutions have been found using the computational grid with hundreds of processors, and thus the reported values of T_{cpu} are not for the same computational environment as for the unsupervised learning approaches. The times for the VNS-based approach are almost about one hour, and therefore, it is evident the SOM and GSOA approaches are significantly less demanding.

Table 7 Results for the Close Enough Dubins Orienteering Problem

Problem	VNS		SOM		**Proposed**	
(T_{max}, $\rho = 1$)	R_{max}	R	R_{max}	R	R_{max}	R
Set 3 ($\delta = 0.5$)	570	570	**530**	**516**	**530**	514
Set 3 ($\delta = 1.0$)	620	615	570	550	**580**	**560**
Set 64 ($\delta = 0.5$)	990	973	738	687	**780**	**726**
Set 64 ($\delta = 1.0$)	1326	1303	1068	1017	**1152**	**1064**
Set 66 ($\delta = 0.5$)	1055	1032	895	853	**930**	**916**
Set 66 ($\delta = 1.0$)	1485	1396	1040	984	**1110**	**1070**

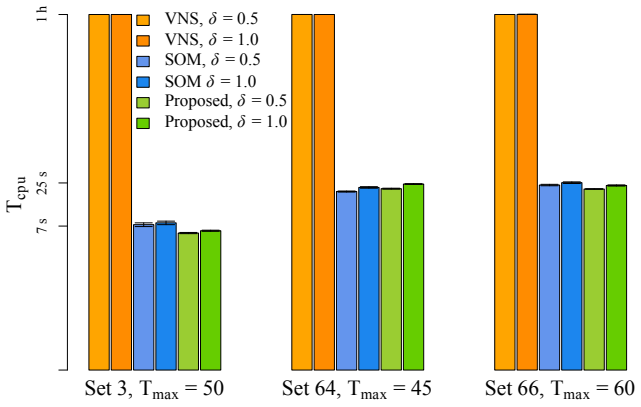
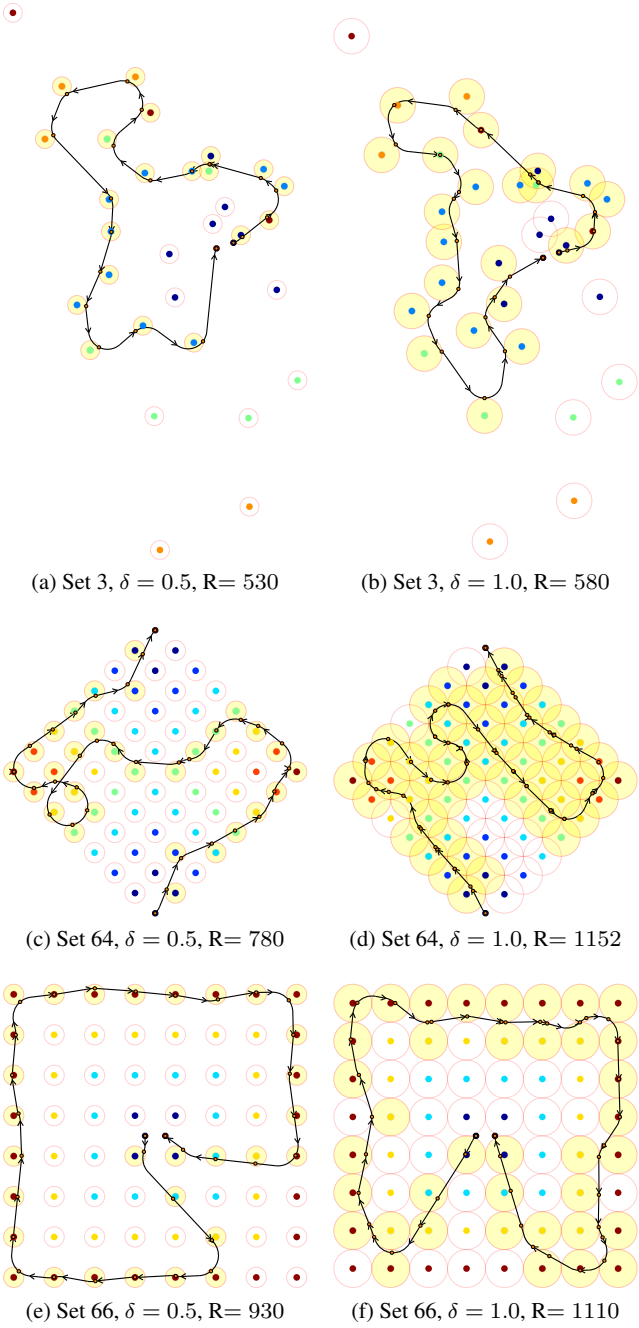


Fig. 11 Computational requirements of the evaluated solvers in the solution of CEDOP. Both unsupervised learning approaches SOM and GSOA provides a solution in less than 25 seconds, while the VNS-based approach needs about one hour.

The proposed GSOA improves the performance of the unsupervised learning approaches in comparison to SOM [11] and overall it seems to be a bit faster. Detailed results for the selected instances according to Table 2, $\rho = 1$, and $\delta \in \{0.5, 1.0\}$ are reported in Table 7 with an overview of the computational requirements in Fig. 11. Selected solutions found by the proposed GSOA are depicted in Fig. 12.



(a) Set 3, $\delta = 0.5$, R= 530 (b) Set 3, $\delta = 1.0$, R= 580

(c) Set 64, $\delta = 0.5$, R= 780 (d) Set 64, $\delta = 1.0$, R= 1152

(e) Set 66, $\delta = 0.5$, R= 930 (f) Set 66, $\delta = 1.0$, R= 1110

Fig. 12 Selected solutions of CEDOP instances Set 3, T_{max}= 50; Set 64, T_{max}= 45; and Set 66, T_{max}= 60; all with $\rho = 1.0$ found by the proposed GSOA.

5.3 Discussion

The reported results support feasibility of the proposed GSOA-based approach to address computationally challenging CE-

DOP. The employed novel unsupervised learning procedure of the GSOA improves the solution and reduce computational requirements in comparison to the previous SOM-based approaches. A noticeable improvement is achieved for few instances of the CEOP, but significant improvement is achieved for the DOP and CEDOP, especially for Set 64 and Set 66. It is mostly caused by the GSOA, where a new node is always added to the array together with its waypoint location, but also by the improved adaptation and removal of up to l_{max} nodes using the ratio (5), which altogether support a selection of more rewarding sensors. On the other hand, the combinatorial optimization based on the VNS meta-heuristic provides better results at the cost of increased computational requirements. This is most visible in the solution of Set 64 where a short Dubins path for $\rho \geq 1$ requires appropriate determination of the headings to find a diagonal path as in the upper left part of the solution Set 64 in Fig. 10d, which has not been found for the right bottom part by the GSOA. More supporting headings may help to improve the solution at the cost of increased computational requirements. E.g., for the increased $h = 6$, which means 13 heading values per each waypoint, the sum of the collected rewards is increased about one hundred, which is still worse than the VNS-based solution while the computational time is about two times increased. Therefore, some of the speedup data structure as in the VNS-based method can be utilized to decrease the computational burden. Alternatively, a different strategy for creating supporting headings can be employed to fit the regular diamond shape of Set 64 better, as it is the only scenario where the VNS-based approach provides significantly better solutions of the DOP instances than unsupervised learning methods.

It is worth mentioning that the SOM and GSOA approaches use an online sampling of the headings during the adaptation and despite all Dubins maneuvers are repeatably computed without any pre-computation nor optimized implementation as in the VNS-based approach [10], they are both less computationally demanding. The particular results for DOP instances indicate that except Set 64, the proposed GSOA provides competitive solutions to the VNS-based approach. However, in the case of CEDOP with non-zero sensing range, the GSOA is stuck in local optima and does not explore the large search space. The simple local improvement heuristic utilized in the proposed conditional adaptation (i.e., the deletion of up to l_{max} nodes) provides only a limited capability of improving the solution and a more systematic search in the VNS-based method provides better results. Since the GSOA is about two orders of magnitude faster than the VNS-based approach in solving CEDOP instances, it provides an opportunity to combine some local heuristics and combinatorial operators to improve the solution quality, but still keep the computational requirements low. Therefore, it is expected that additional local improvements, e.g., iterative local optimization proposed in [30], combined with the memetic algorithm would improve the solution similarly as in [43] for the Euclidean TSP.

6 Conclusion

In this paper, a novel unsupervised learning procedure is proposed to solve a variant of the Dubins orienteering problem with disk-shaped neighborhoods called Close Enough Dubins Orienteering Problem (CEDOP). The proposed approach leverages on the previous SOM-based solvers for the OPN and DTSPN, but it employs recently proposed Growing Self-Organizing Array (GSOA) to improve SOM-based methods in solving variants of the orienteering problem. The presented results support the proposed approach is viable and show improvement of the unsupervised learning based methods in solving orienteering problems. The proposed approach provides competitive results with the existing VNS-based solution in instances with relatively sparse target locations.

The only instances where unsupervised learning methods perform significantly worse than the VNS-based method are from Set 64 where the target locations are placed in a diamond-shaped grid. In solving CEDOP with non-zero sensing range, the proposed approach provides worse solutions than the computationally very demanding VNS. It is because of relatively simple local rules for removing nodes and the visited targets during the learning to meet the requirements of the travel budget. On the other hand, the proposed approach is significantly less computationally demanding than the VNS albeit Dubins maneuvers are repeatedly computed during the learning. Therefore, a possible future research direction is to address the identified drawbacks of the proposed approach by more sophisticated structure to speed up the computation of Dubins tour represented by the array of nodes to further employ local optimization strategies for improving the solution quality and keep the computational requirements low. Besides, the proposed GSOA can be utilized for solving the addressed orienteering problems with a team of vehicles, which is also a subject of the future work.

Acknowledgments

The presented work has been supported by the Czech Science Foundation (GAČR) under research project No. 16-24206S.

Access to computing and storage facilities owned by parties and projects contributing to the National Grid Infrastructure MetaCentrum, provided under the programme "Projects of Large Research, Development, and Innovations Infrastructures" (CESNET LM2015042), is greatly appreciated.

References

1. K. Vicencio, B. Davis, I. Gentilini, in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2014), pp. 2985–2990
2. M. Di Francesco, S.K. Das, G. Anastasi, Data Collection in Wireless Sensor Networks with Mobile Elements: A Survey, ACM Transactions on Sensor Networks **8**(1), 1 (2011)
3. D. Bhadauria, O. Tekdas, V. Isler, Robotic data mules for collecting data over sparse sensor fields, Journal of Field Robotics **28**(3), 388 (2011)
4. 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)
5. B.L. Golden, L. Levy, R. Vohra, The orienteering problem, Naval Research Logistics (NRL) **34**(3), 307 (1987)
6. G. Best, J. Faigl, R. Fitch, in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2016), pp. 3164–3171
7. J. Faigl, R. Pěnička, G. Best, in *IEEE International Conference on Systems, Man, and Cybernetics* (2016), pp. 1315–1321
8. D.J. Gulczynski, J.W. Heath, C.C. Price, *The Close Enough Traveling Salesman Problem: A Discussion of Several Heuristics* (Springer US, Boston, MA, 2006), pp. 271–283
9. R. Pěnička, J. Faigl, P. Váňa, M. Saska, Dubins orienteering problem, IEEE Robotics and Automation Letters **2**(2), 1210 (2017)
10. R. Pěnička, J. Faigl, P. Váňa, M. Saska, in *International Conference on Unmanned Aircraft Systems (ICUAS)* (2017), pp. 1555–1562
11. J. Faigl, R. Pěnička, in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2017), pp. 5646–5652
12. P. Hansen, N. Mladenović, Variable neighborhood search: Principles and applications, European Journal of Operational Research **130**(3), 449 (2001)
13. J. Faigl, P. Váňa, M. Saska, T. Báča, V. Spurný, in *European Conference on Mobile Robots (ECMR)* (2017), pp. 1–6
14. J. Faigl, in *12th International Workshop on Self-Organizing Maps and Learning Vector Quantization, Clustering and Data Visualization (WSOM+)* (2017), pp. 125–132
15. J. Faigl, GSOA: growing self-organizing array - unsupervised learning for the close-enough traveling salesman problem and other routing problems, Neurocomputing **312**, 120 (2018)
16. Bo Yuan, M. Orlowska, S. Sadiq, On the Optimal Robot Routing Problem in Wireless Sensor Networks, IEEE Transactions on Knowledge and Data Engineering **19**(9), 1252 (2007)
17. B. Behdani, J.C. Smith, An integer-programming-based approach to the close-enough traveling salesman problem, INFORMS Journal on Computing **26**(3), 415 (2014)
18. F. Carrabs, C. Cerrone, R. Cerulli, M. Gaudioso, A novel discretization scheme for the close enough traveling salesman problem, Computers & Operations Research **78**, 163 (2017)
19. P. Vansteenwegen, W. Souffriau, D.V. Oudheusden, The orienteering problem: A survey, European Journal of Operational Research **209**(1), 1 (2011)
20. A. Gunawan, H.C. Lau, P. Vansteenwegen, Orienteering Problem: A survey of recent variants, solution approaches and applications, European Journal of Operational Research **255**(2), 315 (2016)
21. J. Faigl, V. Vonásek, L. Přeučil, Visiting Convex Regions in a Polygonal Map, Robotics and Autonomous Systems **61**(10), 1070 (2013)
22. J. Faigl, G.A. Hollinger, Autonomous data collection using a self-organizing map, IEEE Transactions on Neural Networks and Learning Systems **29**(5), 1703 (2018)
23. J. Faigl, in *International Joint Conference on Neural Networks (IJCNN)* (2017), pp. 2611–2620
24. J. Le Ny, E. Feron, E. Frazzoli, On the Dubins Traveling Salesman Problem, IEEE Transactions on Automatic Control **57**(1), 265 (2012). DOI 10.1109/TAC.2011.2166311
25. S. Manyam, S. Rathinam, A tight lower bounding procedure for the dubins traveling salesman problem, arXiv preprint arXiv:1506.08752v2 (2015). URL `http://arxiv.org/abs/1506.08752v2`. (cited on 04-24-2018)
26. J.T. Isaacs, J.P. Hespanha, Dubins traveling salesman problem with neighborhoods: a graph-based approach, Algorithms **6**(1), 84 (2013)
27. X. Zhang, J. Chen, B. Xin, Z. Peng, A memetic algorithm for path planning of curvature-constrained uavs performing surveillance of multiple ground targets, Chinese Journal of Aeronautics **27**(3), 622 (2014). DOI 10.1016/j.cja.2014.04.024
28. P. Oberlin, S. Rathinam, S. Darbha, Today's traveling salesman problem, Robotics & Automation Magazine, IEEE **17**(4), 70 (2010)
29. K. Savla, E. Frazzoli, F. Bullo, in *Proceedings of the American Control Conference* (IEEE, 2005), pp. 786–791
30. P. Váňa, J. Faigl, in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2015), pp. 4029–4034
31. P. Isaiah, T. Shima, Motion planning algorithms for the Dubins Travelling Salesperson Problem, Automatica **53**, 247 (2015)
32. K.J. Obermeyer, P. Oberlin, S. Darbha, Sampling-based path planning for a visual reconnaissance unmanned air vehicle, Journal of Guidance, Control, and Dynamics **35**(2), 619 (2012). DOI 10.2514/1.48949
33. D.G. Macharet, J.W.G. Monteiro, G.R. Mateus, M.F.M. Campos, Bi-objective data gathering path planning for vehicles with bounded curvature, Computers & OR **84**, 195 (2017)
34. J. Faigl, P. Váňa, in *International Conference on Artificial Neural Networks (ICANN)* (2016), pp. 497–505
35. J. Faigl, P. Váňa, in *International Joint Conference on Neural Networks (IJCNN)* (2017), pp. 4340–4347
36. C. Archetti, F. Carrabs, R. Cerulli, The set orienteering problem, European Journal of Operational Research **267**(1), 264 (2018)
37. G. Best, J. Faigl, R. Fitch, Online planning for multi-robot active perception with self-organising maps, Autonomous Robots **42**(4), 715 (2018)
38. J. Faigl, Data collection path planning with spatially correlated measurements using growing self-organizing array, Applied Soft Computing **75**, 130 (2019)
39. R. Ramesh, K.M. Brown, An efficient four-phase heuristic for the generalized orienteering problem, Computers & Operations Research **18**(2), 151 (1991)
40. The Orienteering Problem: Test Instances – Department of Mechanical Engineering. URL `http://www.mech.kuleuven.be/en/cib/op#section-0`. (cited on 04-24-2018)
41. T. Tsiligirides, Heuristic methods applied to orienteering, The Journal of the Operational Research Society **35**(9), 797 (1984)
42. I.M. Chao, B.L. Golden, E.A. Wasil, A fast and effective heuristic for the orienteering problem, European Journal of Operational Research **88**(3), 475 (1996)
43. J.C. Créput, A. Koukam, A memetic neural network for the Euclidean traveling salesman problem, Neurocomputing **72**(4-6), 1250 (2009)