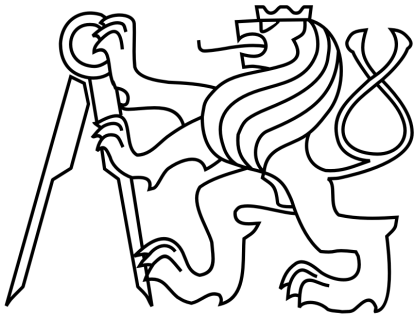
Czech Technical University in Prague

Faculty of Electrical Engineering
Department of Cybernetics
Multi-robot Systems



Data Collection Planning For Aerial Vehicles

Doctoral Thesis

Ing. Robert Pěnička

Prague, January 2020

Ph.D. programme: Electrical Engineering and Information Technology
Branch of study: Artificial Intelligence and Biocybernetics

Supervisor: Ing. Martin Saska, Dr. rer. nat.
Supervisor-Specialist: prof. Ing. Jan Faigl, Ph.D.

Acknowledgments

I would like to gratefully thank my supervisor Martin Saska for his great guidance during my PhD studies, uncountable advises, and sometimes for reminding me that reviewers are also only humans and that the first impression is what sometimes matters the most. My sincere gratitude also goes to my supervisor-specialist Jan Faigl for enthusiasm for solving new scientific problems, his pressure towards writing the best possible publications, and also for his attention to detail. Further, I would like to thank all my colleagues from Multi-robot systems (MRS) group for their friendship and help throughout my studies. I thank Vojtěch Vonásek, Petr Štěpán, Tomáš Báča, Vojtěch Spurný, Viktor Walter, Matěj Petrlík, Matouš Vrba, Daniel Heřt, Vít Krátký, Pavel Petráček, Petr Štibinger and David Žaitlík. Without their enormous work on MRS autonomous aerial vehicle system, it would not be possible to accomplish the results presented in this thesis. Special thanks go to Tomáš Krajník, who led my Bachelor thesis and thus allowed me to find out what it is like to be a member of a team working on an interesting scientific project. I am very grateful to Vojtěch Vonásek who initially showed me how to write decent scientific text while leading my Diploma thesis and further advised me two important rules at the beginning of my PhD studies: 'do not lose yourself in work unimportant for dissertation' and 'do only things that you think are important and worth solving'. Finally, I would like to thank all my friends and my family for their support. Above all, I am extremely grateful to my wife Kristýna for supporting me during my studies and research.

During my PhD studies, I have been supported by taxpayers of the great Czech Republic through PhD scholarship. The Czech Technical University in Prague supported my work with grants SGS15/157/OHK3/2T/13 and SGS17/187/OHK3/3T/13. The Czech Science Foundation supported this work through projects No. 16-24206S, No. 17-16900Y, No. 19-20238S, and No. 19-22555Y. The Ministry of Education of the Czech Republic has funded my research by OP VVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics". The National Grid Infrastructure MetaCentrum provided access to computing and storage facilities under the programme CESNET LM2015042. The Khalifa University of Science funded our participation in the MBZIRC 2017 and MBZIRC 2020 competitions that also motivated this work.

Copyright

Abstract

This thesis concerns planning for data collection missions with Unmanned Aerial Vehicles (UAV). The data collection is one of the many use cases of UAVs where the onboard sensors can produce various data such as thermal images in search and rescue scenarios, visual images for surveillance, or LiDAR point clouds for inspection. The task of the data collection planning is to find a plan that visits a predefined set of target locations to retrieve the data. The data collection planning is essential for all mentioned scenarios to achieve, with respect to a planning objective, efficient deployment of UAVs in data collection missions. The goal of this thesis is to make the data collection with UAVs feasible and more effective. Two planning objectives of the data collection planning are used in the thesis: 1) minimization of the path cost visiting all target locations, and 2) maximization of the collected data using a predefined path cost. This thesis is a compilation of five journal publications with contributions towards the data collection planning with UAVs and all focus on unique aspects of such planning. The important aspect that influences the feasibility of plans for the nonholonomic or dynamically constrained UAVs is the ability to visit the target locations in planned order when traveling a certain speed. It is addressed by employing the curvature-constrained Dubins vehicle for modeling the UAV in data collection planning. This, however, requires to create a novel data collection formulations for the Dubins vehicle model. We further show that at the cost of a more complex optimization problem, the amount of collected data can be increased by using a non-zero sensing range when collecting the data without the precise visit of the target locations, e.g., for a long-range sensor attached to UAV. Both data collection planning using Dubins vehicle and with non-zero sensing range are formulated as an Integer Linear Program (ILP) that is solved optimally for certain problem sizes. For the objective of minimizing the data collection plan cost, we propose methods that can be used for a multi-vehicle variant of data collection with Dubins vehicle model and non-zero sensing range. Finally, we extend the proposed planning methods for environments with obstacles. Such environments represent more realistic data collection scenarios, however, they require to solve a challenging combination of the multi-goal routing problem with collision-free motion planning. The proposed planning methods are mostly based on Variable Neighborhood Search, Self-Organizing Map, and ILP formulations. All methods are evaluated in numerous test instances from the literature and the feasibility of found plans is experimentally verified with real UAVs.

Keywords Unmanned Aerial Vehicles · Data Collection Planning · Non-holonomic Motion Planning · Traveling Salesman Problem · Orienteering Problem · Operational Research · Motion Planning · Dubins Vehicle

Abstrakt

Tato práce se zabývá plánováním misí sběru dat bezpilotními vzdušnými prostředky (Unmanned Aerial Vehicles – UAV). Sběr dat je jedním z mnoha využití UAV, při kterém mohou být palubní sensory použity například záchrannými složkami k pátrání, dohledu nad zvolenou oblastí, nebo inspekci budov. Úkolem plánování sběru dat je nalezení plánu, který navštíví předem definovaná cílová místa. Plánování misí je klíčovým problémem efektivního využití UAV vzhledem k očekávanému výsledku mise. Cílem této práce je umožnit efektivní nasazení UAV s ohledem na optimalizační kritéria: 1) minimalizace délky cesty navštívení všech definovaných cílů a 2) maximalizace dat s omezením délky cesty. Text disertační práce je souborem pěti časopiseckých publikací, které se zaměřují na specifické aspekty plánování sběru dat UAV prostředky. Důležitým aspektem ovlivňujícím proveditelnost plánu neholonomních nebo dynamicky omezených UAV je schopnost navštívit cílová místa v naplánovaném pořadí při dodržení předepsané letové rychlosti. Tento aspekt je řešen využitím modelu Dubinsova vozidla s omezeným poloměrem zatáčení. Model Dubinsova vozidla vyžaduje nové formulace plánování přes více cílů, které v práci navrhujeme společně s metodami řešení a experimentálním ověřením. Dále ukazujeme, že je možné zlepšit kvalitu řešení v podobě zvýšení nasbíraných dat explicitním zahrnutím dosahu palubního sensoru, a to za cenu zvýšené komplexnosti optimalizační úlohy. Varianty plánování s modelem Dubinsova vozidla i s uvažováním dosahu sensoru jsou formulovány celočíselným lineárním programováním (Integer Linear Program – ILP) a řešeny optimálně pro určité velikosti problémů. Pro optimalizační kritérium minimalizace délky cesty jsou navrženy metody plánování pro více prostředků s modelem Dubinsova vozidla a uvažováním dosahu sensoru. Plánovací metody jsou dále zobecněny pro prostředí s překážkami, které představují realističtější scénář nasazení autonomních UAV. V prostředí s překážkami je však nutné řešit kombinaci bezkolizního plánování pohybu s kombinatorickým plánováním přes více cílů. Navržené metody plánování sběru dat jsou založeny na algoritmech Variable Neighborhood Search, Self-Organizing Map a formulaci ILP. Metody jsou ověřeny na testovacích scénářích z literatury a proveditelnost plánů je experimentálně otestována na reálných UAV.

Klíčová slova Bezpilotní vzdušné prostředky · Plánování robotického sběru dat · Neholonomní plánování pohybu · Problém obchodního cestujícího · Plánování cest · Operační výzkum · Plánování cest · Dubinsovo vozidlo

Contents

Abbreviations

AA	Alternating Algorithm
ATSP	Asymmetric Traveling Salesman Problem
CEDOP	Close Enough Dubins Orienteering Problem
COP*	Clustered Orienteering Problem
COP\CorOP	Correlated Orienteering Problem
DIP	Dubins interval problem
DOP	Dubins Orienteering Problem
DOPN	Dubins Orienteering Problem with Neighborhoods
DTP	Dubins Touring Problem
DTRP	Dubins Touring Regions Problem
DTSP	Dubins Traveling Salesman Problem
DTSPN	Dubins Orienteering Problem with Neighborhoods
EOP	Euclidean Orienteering Problem
ETSP	Euclidean Traveling Salesman Problem
GTSP	Generalized Traveling Salesman Problem
ILP	Integer Linear Programming
KP	Knapsack Problem
LIO	Local Iterative Optimization
LKH	Lin-Kernighan-Helsgaun algorithm
MBZIRC	Mohamed Bin Zayed International Robotics Challenge
MILP	Mixed-Integer Linear Programming
MIP	Mixed-Integer Programming
MPC	Model Predictive Control
OP	Orienteering Problem
OPN	Orienteering Problem with Neighborhoods
PC-TSPN	Prize-Collecting Traveling Salesman Problem with Neighborhoods
POP	Physical Orienteering Problem
PRM	Probabilistic Roadmap method
PTSP	Physical Traveling Salesman Problem
RRT	Rapidly-Exploring Random Trees
RVNS	Randomized Variable Neighborhood Search
SEC	Subtour Elimination Constraints
SOM	Self-Organizing Map
SOP	Set Orienteering Problem
TOP	Team Orienteering Problem
TSP	Traveling Salesman Problem
TTE	Travel Time Estimation
UAV	Unmanned Aerial Vehicle
VNS	Variable Neighborhood Search
VTOL	Vertical Take-Off and Landing
WSN	Wireless Sensor Network

Chapter 1

Introduction

Research on Unmanned Aerial Vehicles (UAVs) is one of the most rapidly developing fields in mobile robotics. One of the reasons is that UAVs, often called drones, have become more affordable due to their popularity among wider public and enthusiasts. Even the off-the-shelf products are nowadays capable of semi-autonomous behavior such as collision avoidance, navigation through the environment, or following a person. Such UAVs are used by security and rescue forces during search and rescue missions, by filmmakers or photographers as suitable platforms for high altitude shots, and of course by the wider public mostly as high-tech toys. However, the applications of the remotely controlled UAVs go beyond the mentioned ones; the UAVs are mostly used as ideal platforms for carrying onboard cameras and other sensors for remote sensing [20] in various fields of application. These applications can be categorized as data collection missions.

In robotics, scientists and engineers focus mainly on developing all kind of autonomous behavior where the UAV does not require to be remotely controlled. Nevertheless, the most targeted fields of application of the fully autonomous UAVs are the same as for the remotely controlled, and thus the autonomous UAVs are often intended as an effective way for long-range autonomous data collection. Especially the multirotor UAVs with Vertical Take-Off and Landing (VTOL) are popular due to their ability to hover over a single location while measuring desired phenomena. The fixed-wing UAVs, on the other hand, can provide longer flight time and travel distance. Despite that, the VTOLs have higher maneuverability and thus easier reachability of desired target locations in environments with obstacles. However, one of the most crucial aspects of deployment of the autonomous UAVs in data collection missions, that highly influences its effectiveness and feasibility, is the mission and path planning.

This thesis focuses on data collection planning for UAVs. It is based on five core journal publications [1c]–[5c] that share the common goal of allowing more effective deployments of aerial vehicles in the data collection missions. Data collection planning throughout this thesis is understood as a multi-goal path planning where multiple target locations are present in the environment, and the UAV is required to visit them to, e.g., measure desired phenomena. One of the most common objectives of the data collection planning is to plan the shortest possible path over the target locations or to maximize the collected data with respect to limited flying time due to a battery capacity. The data collection planning problem thus belongs to a class of routing problems [21] where the visit to multiple targets has to be optimized [22] by, e.g., changing the sequence to visit the individual target locations or, for a multi-vehicle variant, by assigning the target locations to a different vehicle. Figure 1.1 shows an example of a data collection plan for a UAV with the objective of maximization of the collected data using a predefined path cost.

The planning of data collection missions for UAVs has multiple unique challenges that influence plan feasibility and its quality. The five core publications that form this thesis address the following **challenges (1)–(5) that are also the research goals of this thesis**.

(1) Feasibility of traveling with nonholonomic fixed-wing UAV or dynamically constrained VTOL UAV between individual target locations is the first challenge of this thesis. Ordinary data collection planning uses cost straight line paths between target locations with sharp corners. Such paths can be feasible for holonomic ground robots, but they are unfeasible for the UAVs traveling a certain speed. The feasibility aspect has been tackled in the core publication [1c] and further elaborated in the remaining core publications by using

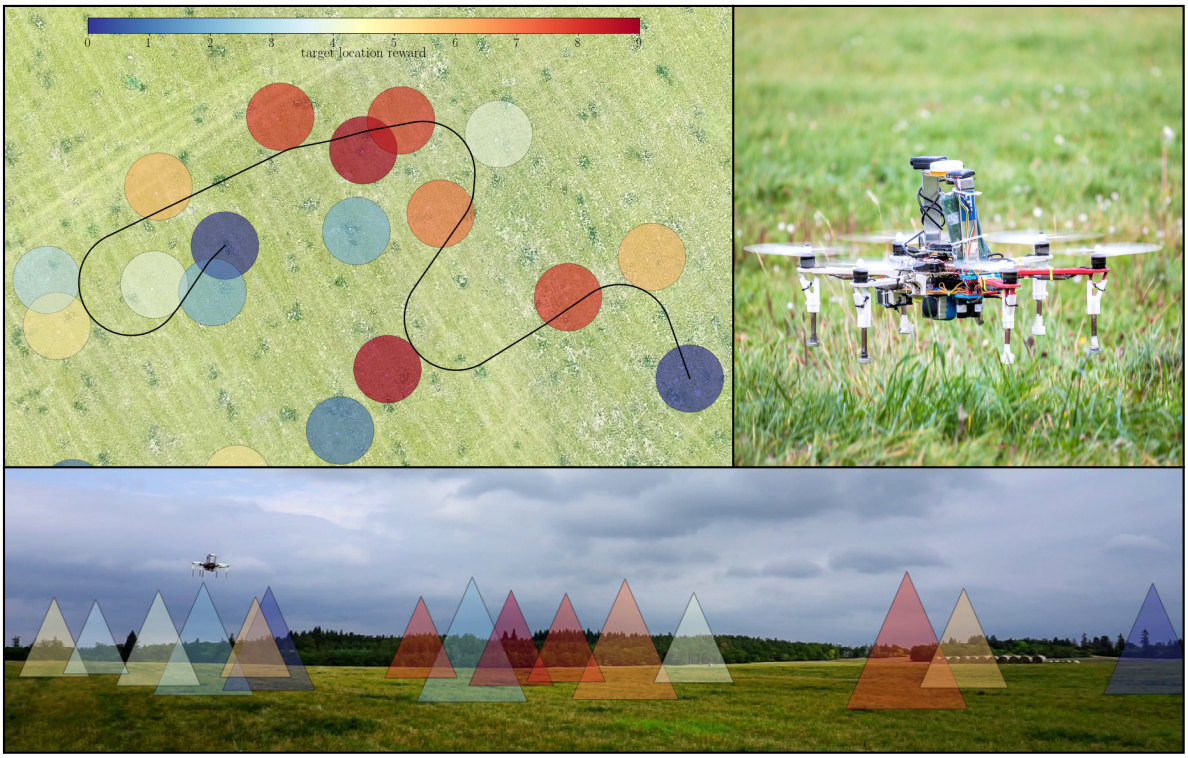


Figure 1.1: Example of visual data collection with maximization of the collected reward (data) from target locations using non-zero sensing range of employed onboard sensor and with limited flight budget of UAV [2c].

curvature-constrained Dubins vehicle [23] to model the UAV. The routing with Dubins vehicle, however, requires to determine not only the sequence of visits to the target locations but also heading angles of the vehicle at each visited location. This adds additional complexity to the multi-goal planning problems and represents a real challenge due to the strong influence of the selection of the heading angles to the final solution length.

(2) Planning with respect to the limited time of flight is crucial for the flying robots with current battery technology that restricts flight duration to several tens of minutes. Therefore, the second challenge is the direct formulation of the data collection planning for UAVs as a budget limited routing problem to address situations where only part of target locations is reachable. The majority of the core publications [1c]–[3c], [5c] use a variant of the Orienteering Problem (OP) [24] to formulate the data collection for UAVs. The objective of the OP is to maximize the collected reward (data) within a limited travel budget (flight time). This is in contrast to the core publication [4c] with a variant of the Traveling Salesman Problem (TSP) [25], where a path visiting all the given locations with minimal flight time as the main cost is to be found. The TSP objective is, however, usable only for a cases where all given locations are to be visited, and the cost of the plan is relatively smaller than the vehicle budget. Therefore, the proposed planning methods mostly focus on the limited budget OP formulation for UAVs.

(3) Data collection planning with non-zero sensing range can significantly improve the efficiency of data collection and it is the third challenge of this thesis. The advantage of aerial vehicles to collect the data from high altitudes also allows, in some cases, to gather the data within, e.g., circular neighborhoods (sensing radius) around the target locations [2c]–[4c] without visiting them precisely. This can improve the amount of collected data (solution

quality) within the same period, for example, in a visual data collection with a wide field of view cameras, for data collection with a long-range sensor, or in information gathering from wireless sensor networks. The non-zero sensing range, however, requires to be directly addressed in data collection planning formulation. Then, while solving the routing problem, the particular data collection positions have to be found and optimized within the sensing range of visited target locations.

(4) Finding optimal solutions for data collection with Dubins vehicle and non-zero sensing range is the fourth challenge of this thesis. The data collection planning with Dubins vehicle and also with non-zero sensing range can be formulated as an Integer Linear Programming (ILP) problem for a given sampling of Dubins vehicle heading angle and data collection positions within the sensing range of target locations. A recent ILP formulation called Set Orienteering Problem (SOP) [26] from the Operational Research field can be used for the data collection planning, as shown in the core publication [3c].

(5) Data collection planning in environments with obstacles is the last considered challenge of this thesis. The core publication [5c] addresses the aspect of the planning in environments with obstacles where the multi-goal routing problem of the OP has to be solved simultaneously with the point-to-point collision-free motion planning [27]. Such planning has to find collision-free paths mutually between all target locations, and the costs of such collision-free paths have to be ideally as small as possible. Both demands are computationally demanding. Therefore, the planning can not be addressed separately, by firstly finding the collision-free paths and then using them for finding the multi-goal sequence, without decreasing the solution quality. We show that the data collection planning in environments with obstacles requires a single method that optimizes only the collision-free paths that are concurrently used in the multi-goal route planning.

The presented work and thus, the application of the proposed data collection planning methods, are motivated by three following robotic scenarios where the autonomous UAVs can be effectively used. The scenarios mainly differ in a way how the target locations for a specific scenario are defined or determined, and in individual scenario constraints. However, they share the common multi-goal path planning part that has to be solved.

The first scenario can be called information gathering or simple data collection, and we assume a priori knowledge of the target locations in the environment. Such a scenario consists of UAVs equipped with an onboard sensor that is required to reach the target locations and measure or collect the desired data [28] such as images, weather-related data, or even radiation measurements. Another example of deploying UAVs in information gathering is Wireless Sensor Networks (WSN). In WSN, the sensory units are placed in the environment, and the considered UAVs can be used for retrieving the measured data from the sensor units by wireless communication with a limited range [29], [30]. Hence, the objective of the planning for the simple data collection is the maximization of the gathered information from the given locations within predefined vehicle flight time. Minimizing the information gathering path can be considered if all target locations can be feasibly visited within the given travel budget.

Surveillance is the second scenario where UAVs can be effectively deployed for data collection missions [31]. In surveillance, UAVs are required to monitor a given area or infrastructure persistently while traveling over target locations that have to be found for a specific monitoring task. An example of surveillance applications can be flood monitoring using unmanned aerial vehicles [32]. Security and rescue forces can use classical or thermal imaging cameras onboard to assist in search and rescue after disasters like earthquakes, forest fires and

floods [32]–[34]. Also, all inspection tasks belong to this scenario, such as inspection of power lines [35], wind turbines [36], or power plants [37]. The inspection tasks also usually share the requirement to find the target locations prior to the data collection planning to inspect the required structure. In path planning for surveillance missions, the objective is usually to minimize the required time to survey/inspect a given area or infrastructure.

The last scenario that motivates this work is Coverage Path Planning (CPP) [38] where the UAVs are requested to scan a given area entirely with an onboard sensor and either create a map of the environment or store sensory data for post-processing. In the CPP, the target locations can be placed in the environment uniformly, e.g., in a grid and the boustrophedon back and forth path [39] is usually sufficient for coverage of the area without directly considering the target locations placement. However, in the presence of no-fly-zones, the area has to be decomposed into smaller parts and the multi-goal data collection planning can be used to schedule the order of visiting the parts for the plow-like coverage [40]. UAVs used in such missions typically use onboard cameras to scan the area and consequent reconstruction of the area can be used for example in archeology [41], [42]. Similarly a UAV with LIDAR sensor can be used for creating a terrain profile map [43]. The objective of the CPP is usually the minimization of the coverage path cost.

The rest of this thesis is organized as follows. In the next chapter, we summarize the state-of-the-art in data collection planning for aerial vehicles and highlight the contributions of this thesis with respect to the related work. In Chapters 3–7, a short introduction of individual core publications is followed by the manuscripts themselves. Chapter 8 discusses the achieved results of the core publications related to the research goals of the thesis. The concluding remarks and future work is summarized in Chapter 9.

Chapter 2

Contributions and related work

This chapter overviews the contributions of the author with the primary focus on the five journal articles [1c]–[5c] that form the core of the thesis. However, we refer to the individual core publications for a detailed state-of-the-art overview. We further outline the most relevant related works together with other author's contributions [6a]–[19a] related to the thesis. The remainder of this chapter is divided into three sections based on the similarity of the contributions of the individual core publications.

2.1 Budget limited data collection planning

One of the most crucial deployment limitations of today's UAVs is the limited time of flight, which is usually restricted to several tens of minutes. Therefore, we mainly focused our research on formulations of the data collection planning that takes the limited budget into account.

The most regular multi-goal routing problem, and thus formulation of the data collection planning, in the literature is the Traveling Salesman Problem (TSP) [25], [44]. In the TSP, the objective is to minimize the cost to visit a given set of target locations. The solution approaches for the TSP thus have to find a sequence in which the particular target locations are visited using the most efficient path, which is known to be an NP-hard problem [45]. The TSP can be used for planning the UAV operations [22], however, it does not directly address the limited time of flight of nowadays UAVs.

The Orienteering Problem (OP) [24] is a variant of the TSP with profits for vehicles with a limited budget. The OP stands to find a path connecting a subset of target locations, each with associated reward, such that the collected reward is maximized within a given travel budget. The ordinary OP, with predefined starting and ending target locations, uses Euclidean distances between target locations. Solving the OP thus requires to find the most rewarding subset of the target locations to visit and, at the same time, to minimize the path visiting the targets in the subset such that its length is within the prescribed budget. The OP is, therefore, a combination of the well-known NP-hard Knapsack problem and the TSP.

The OP has many variants and has been tackled by many solution approaches [46]. It can be formulated as an Integer Linear Programming (ILP) problem [47] and solved optimally using Branch and Bound [48] or Branch and Cut [49]. However, heuristic approaches require less computational time and can, in many cases, achieve optimal solutions. The heuristics for the OP are based on evolutionary algorithm [50], greedy randomized adaptive search procedure [51], or ant colony optimization and tabu search [52]. Among others, the Variable Neighborhood Search (VNS) approach [53] becomes, due to its low computation time and high solution quality, basis for our VNS-based methods for the OP variants being solved in the core publications of this thesis. The VNS [54] is metaheuristic by Mladenović and Hansen for combinatorial optimization applicable to numerous problems [55]. VNS uses predefined neighborhood operators in an iterative pursuit of improving initial greedy solution inside *shaking* and *local search* procedures. While the *shaking* randomly changes the incumbent solution to get from possible local minimum, the *local search* tries to find, on such a randomly created solution, a path with higher reward than the incumbent solution.

The ordinary OP uses straight line paths to connect the target locations and thus Euclidean distances as precomputed costs. This is, however, unfeasible for the nonholonomic

fixed wing UAVs and also for dynamically constrained VTOL UAVs traveling a certain speed. Therefore, in the first core publication [1c], we introduce a novel variant of the OP which we call Dubins Orienteering Problem (DOP). The objective of the DOP is similar to the ordinary OP, but it uses Dubins vehicle model [23] with curvature constraint induced by a limited turning radius. Using Dubins vehicle model ensures smoothness of the found paths without sharp corners. The Dubins vehicle state is from $SE(2)$, and thus heading angles of the vehicle have to be found at visited target locations in order to connect the adjacent Dubins maneuvers feasibly. The DOP can be considered to be even more challenging than the regular OP due to the requirement to find appropriate heading angles simultaneously with the target subset selection and the sequence to visit them. The selection of the heading angles highly influences the solution length and thus limits solution feasibility due to the limited budget. See the comparison of the solutions of the OP and DOP in Fig. 2.1. The core publication [1c] introduces a VNS-based method for the DOP. We refer to Chapter 3 for more information about the DOP and the VNS-based method for the DOP.

The amount of collected data (reward) can be significantly improved by exploiting non-zero sensing range to save travel cost, i.e., by using neighborhood around target locations where the data can be collected without visiting precise position of the target locations. The Orienteering Problem with Neighborhoods (OPN) uses the non-zero sensing range and has been firstly introduced in [8a] together with a Self-Organizing Map (SOM) based solution approach. The OPN uses a circular neighborhood around each target location which in most cases helps to increase the collected reward compared to the ordinary OP. However, the OPN further requires to find appropriate positions of visit within the neighborhoods of the target locations while solving the subset selection and sequencing of the targets. The variant of the OPN for Dubins vehicle, called Dubins Orienteering Problem with Neighborhoods (DOPN), has been initially introduced in [9a] along with the proposed VNS-based solution approach. See the comparison of the OPN and DOPN with other mentioned OP variants in Fig. 2.1. The VNS for the DOPN is an extension of our previous work [1c] where the border of circular neighborhood of each target is equidistantly sampled together with sampling of the heading angles. The DOPN is also known as the Close Enough Orienteering Problem with Dubins vehicle (CEDOP) and has been addressed using unsupervised learning of the SOM in [10a]. Nevertheless, the second core publication of this thesis [2c] significantly extends our previous VNS-based approach [9a] for the DOPN by using continuous optimization of the samples [56]. Chapter 4 presents in detail the developed VNS-based approach for the DOPN published in the core publication [2c].

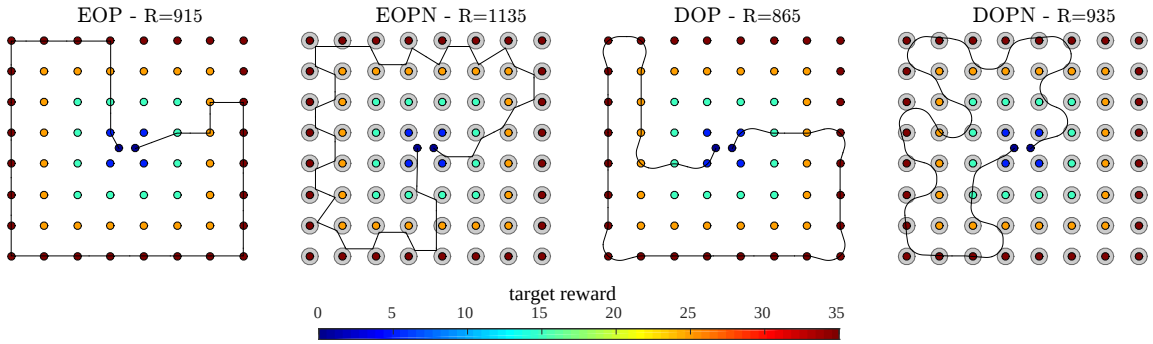
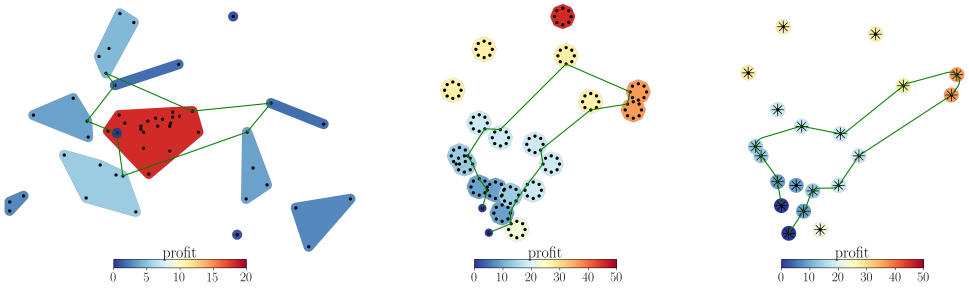


Figure 2.1: Example solutions of the OP, OPN, DOP, and DOPN with respective collected rewards R [2c].

The variants of the OP are useful for modeling robotic data collection planning. The OP is mostly studied in the context of Operational Research, as it belongs to a class of routing problems with profits together with, e.g., Traveling Salesman Problems with Profits [57]. One of the recently proposed routing problems with profits is the Set Orienteering Problem [26], which has been introduced together with Mixed-Integer Linear Programming (MILP) formulation and a matheuristic solution algorithm. The SOP is a variant of the OP where customers (nodes) are grouped in clusters (sets), and each set is associated with a reward that is collected by visiting at least one node in the respective sets. The objective of the SOP is to maximize the collected reward by visiting the sets. In the core publication [3c], we propose a novel ILP formulation for the SOP and propose a VNS-based method for the SOP (VNS-SOP). We show that the sampling-based approaches to the OPN and DOP can be formulated as the SOP, where the neighborhood or heading angle samples server as the nodes of the SOP. The original OPN and DOP targets then represent the clusters of the SOP that consists of the sample nodes. The proposed ILP formulation is shown to perform significantly faster than the original MILP formulation. The developed VNS-SOP is generalization of our previous VNS approaches for the DOP [1c] and DOPN [2c]. See Fig. 2.2 with example solutions of the ordinary SOP instance *11berlin52* and instances of the OPN and the DOP solved as the SOP. We refer to Chapter 5 for more details about the proposed method published in [3c].



11berlin52 OPN Set 2 DOP Set 2

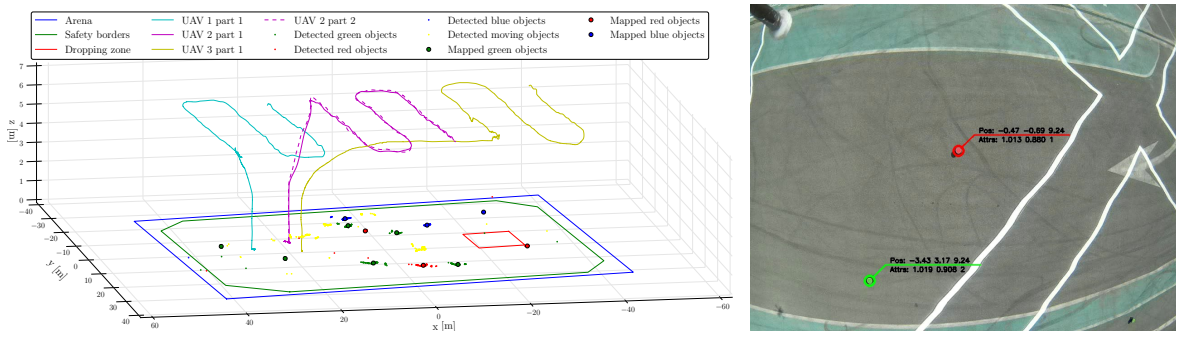
Figure 2.2: Example solutions of the SOP on selected dataset instances from [3c] containing ordinary SOP dataset *11berlin52* and dataset for OPN and DOP formulated as the SOP.

The data collection planning with limited budget can also be done using different motion primitives discretization than the Dubins vehicle model. Examples of such approaches are the Hermite Orienteering Problem [11a] and OP for Dubins airplane model [12a]. The Hermite OP uses the Hermite splines to connect the target locations and can directly address planning for non-constant speed vehicles. However, a velocity profile of the vehicle has to be found for each individual spline. Extension of the DOP to 3D is the Dubins Airplane Orienteering Problem (DA-OP) [12a]. The DA-OP has also been addressed using the VNS-based approach similar to the one in [1c]; however, it uses time-optimal paths for the Dubins airplane [58] as the motion primitives between the target locations. Several other OP formulation variants have been further used in the context of data collection planning for UAVs. A multi-vehicle variant of the OP called the Team Orienteering Problem (TOP) [59] has been proposed for opportunistic surveillance with UAVs. An optimal multilevel graph search technique is proposed for solving the TOP for only small problem instances with Dubins vehicle model of UAV, however, with fixed heading angles at each target location. The TOP for Dubins vehicle is firstly fully addressed in [13a] using Greedy Randomized Adaptive Search Procedure (GRASP). Furthermore, the proposed problem formulation called the Dubins Team Orienteering Problem with Neighborhoods (DTOPN) allows to use non-zero sensing range and collect the reward from close vicinity of given target locations. Finally, the Correlated Orien-

teering Problem (COP) [60] has been proposed for persistent monitoring tasks with UAV and solved using Mixed Integer Quadratic Programming (MIQP). The MIQP formulation uses correlation weights that describe a portion of reward collected from neighboring targets while visiting a particular target. The COP thus model data collection scenarios where visiting some target also provides partial information of other neighboring target locations such as in weather-related data collection.

2.2 Surveillance and spatial coverage with team of aerial vehicles

The surveillance and spatial coverage planning is the main topic of the core publication [4c] and also of other co-authored and related articles [6a], [7a]. The usual objective of the surveillance and spatial coverage is to minimize the cost of data collection path, i.e., the formulations are variants of the TSP. The main reason is that the surveillance missions typically assume the ability to visit all target locations within the limited budget, and the coverage path planning requires to visit all target locations to scan the given area entirely. Furthermore, the surveillance scenarios can require repeated usage of the same plan, e.g., for monitoring of border around restricted area, which makes finding the shortest path even more desirable. The motivation for the three listed articles is the Challenge 3 of 2017 Mohamed Bin Zayed International Robotics Challenge (MBZIRC) [61] held in Abu Dhabi. The Challenge 3 featured a set of colored ferromagnetic objects that had to be found in the arena and collected by a team of three UAVs. Initial scanning of the arena at a high altitude provides only estimated positions of the objects with a possibly high number of false positive detections. Therefore, a fast flight at lower altitude over the detections is performed to verify the locations and to identify the rewards of the objects. See the initial scanning of the arena in the MBZIRC Challenge 3 in Fig. 2.3. The core publication [4c] focuses on planning such fast flights for the team of three UAVs and formulates the problem as a variant of multi-vehicle TSP. Therefore, we further overview the most relevant TSP variants related to the data collection planning with UAVs.



(a) Scanning trajectories used in MBZIRC Challenge 3. (b) UAV onboard Object detection.

Figure 2.3: MBZIRC Challenge 3 mapping of colored objects in competition arena [6a].

The ordinary Euclidean TSP (ETSP), also denoted as the TSP, uses Euclidean distances between target locations and can be formulated as ILP problem and solved optimally using, for example, Concorde solver [62]. For a fast solution of the TSP, a large number of heuristic approaches [44] can be found in the literature, where one of the best performing and widely used is the Lin–Kernighan heuristic [63], [64]. However, the ETSP uses straight line segments with sharp turns between target locations and therefore is unfeasible for nonholonomic or

dynamically constrained UAVs.

Similarly to the limited budget data collection planning formulated as the DOP, the TSP can be formulated for Dubins vehicle [23]. The TSP for the Dubins vehicle (DTSP) [65], apart from determining the order in which the given target locations are visited, also requires determining the heading angles of Dubins vehicle at each target location. The DTSP can be addressed by various approaches where one of the simplest is the Alternating Algorithm (AA) [66]. The AA is a decoupled approach, where a solution of the ETSP is found first without considering the curvature constraint. Afterward, the tour for Dubins vehicle is constructed such that straight lines connect the even edges of the ETSP solution (thus determining headings in all target locations), and the shortest Dubins maneuvers connect the odd edges. The DTSP can also be solved optimally, however, only with respect to a given sampling of heading angles of Dubins vehicle at target locations. Using a finite discrete set of possible heading angles, the DTSP becomes the Generalized Asymmetric TSP (GATSP) [67] that can be further transformed to the Asymmetric TSP (ATSP) using Noon-Bean transformation [68]. The ATSP instance is afterward solvable as a regular TSP instance using heuristics [63] or optimally using, e.g., the Concorde solver [62]. Other existing approaches to the DTSP include unsupervised learning methods based on the evolution of SOM [69] and evolutionary Memetic algorithm [70].

In surveillance and information gathering scenarios, the solution path can be further improved/shortened by considering non-zero sensing range [71]. In Dubins Traveling Salesman Problem with Neighborhoods (DTSPN) the target locations are considered as visited if the found path has a waypoint within distance δ from the respective target locations [72]. The DTSPN can be considered more computationally challenging than the DTSP as it additionally requires to determine the position within the neighborhood of each target location to be visited by the data collection path. Solving the DTSPN can be, similarly to the DTSP, addressed by the sampling-based approach [73], where both neighborhood positions and heading angles are sampled. The generated GATSP instance can be afterward solved either using heuristics or optimally. Alternatively, the DTSPN can be solved using the LIO algorithm proposed in [56], by using genetic algorithms [74], with evolutionary techniques [75] or similar to the DTSP by using the decoupled AA approach with ETSP solution [66].

The multi-vehicle extension of the ordinary TSP is the Multiple Traveling Salesman Problem (m-TSP) [76]. In the *minsum* variant, the objective is to minimize the summed length of all vehicle paths. This variant can, however, lead to a deformed solution with only one vehicle visiting the entire set of target locations without employing all vehicles. The *minmax* m-TSP variant minimizes the length of the longest tour among all vehicles which leads to utilizing all vehicles equally. Both *minmax* [77] and *minsum* [78] m-TSP variants can be solved using exact algorithms. In the case of *minsum* using ILP formulation with transformation [79] to single vehicle TSP. The mTSP has also been addressed using Ant Colony Optimization [80], with genetic algorithms [81] and neural networks [82]. Among other soft-computing techniques and heuristics, the VNS has also been proposed for the m-TSP in [83].

The multi-robot variant of the DTSPN is denoted as m-DTSPN and has been proposed by Macharet et al. in [84] together with the memetic algorithm. The multi-robot TSP for Dubins vehicle has been proposed later by the same authors in [85]. Using both extensions for Dubins vehicle and for non-zero sensing range simultaneously has been proposed in [70] along with an evolutionary algorithm to solve the m-DTSPN. However, the multi-robot variant of the TSP for Dubins vehicle can also be addressed with m-TSP approaches using the sampling of Dubins vehicle heading angle and optionally of the neighborhood positions [25], similarly to the DTSP(N).

The core publication [4c] addresses the fast trajectory planning for verifying the objects in Challenge 3 of MBZIRC 2017 as the *minmax* variant of the m-DTSPN. The proposed method is based on unsupervised learning framework using growing SOM [86]. Furthermore, the SOM-based flexible framework is shown for planning trajectories using Bézier curve model of the UAV which can better exploit the maximal velocities and accelerations of the UAVs. However, the main contribution in [4c] of the author of this thesis is the proposed VNS-based method for planning the m-DTSPN. The method generalizes the VNS-based solver for m-TSP in [83] for Dubins vehicle and also for non-zero sensing range. See Fig. 2.4 with solutions of the m-DTSP planned with the proposed VNS-based solver for verifying detected objects locations and rewards. We refer to Chapter 6 for detailed information about the core publication [4c].

Figure 2.4: Solution of the Multiple Dubins Traveling Salesman Problem obtained by the VNS-based method [4c].

2.3 Data collection planning in environments with obstacles

This section overviews the contribution and related work of the last core publication [5c]. In [5c], we address the data collection planning in environments with obstacles for a limited budget UAV. We propose a novel OP variant which we call the Physical Orienteering Problem (POP). The POP stands to determine a collision-free path in environments with obstacles, and at the same time to maximize the collected reward from the target locations using a limited budget. The problem thus combines collision-free point-to-point motion planning [27] between individual target locations together with the combinatorial optimization of the OP. We further describe the most relevant motion planning approaches and methods that combine routing problems with motion planning.

The task of point-to-point motion planning is to find a motion between two points in an environment while avoiding obstacles. It can be described by a notion of configuration space [27] where the goal is to find a path between two configurations of a robot such that all configurations in the path do not collide with obstacles. One way to address this problem, especially convenient for high-dimensional configuration spaces, is to use sampling-based methods. Sampling-based methods create random samples in the free part of the configuration space without obstacles and then, using the graph search and collision detection methods, find a path between desired terminal configurations using graph of sampled configurations. Among others, the Rapidly-exploring Random Tree (RRT) [87] and the Probabilistic Roadmaps (PRM) [88] are fundamental. The RRT method uses a tree graph representation of samples rooted in the starting configuration, and continuously and randomly expands the tree until the goal

configuration is reached with sufficient precision. The PRM method initially creates a rather large number of samples that are further tried to be connected into k-degree roadmap using k nearest neighbors of each sample and collision detection. The collision-free paths are then found in the tree graph or in the roadmap using a graph search algorithm. However, for the multi-goal routing problems such as the TSP and OP, we need the shortest paths possible to minimize the multi-goal path or to maximize collected reward from targets using limited path cost. The minimality of collision-free path is addressed by the asymptotically optimal Rapidly-exploring Random Tree (RRT*) and Probabilistic Roadmaps (PRM*) methods [89].

The combination of the TSP routing with collision-free motion planning has been studied in the context of video games as the Physical TSP (PTSP) [90]. The PTSP can be addressed as a decoupled problem where we first find the collision-free roadmap, and then the TSP solution can be found using, e.g., Concorde solver [62]. The creation of a collision-free roadmap for multi-goal planning has been proposed using Space-Filling Forest (SFF) [14a] method or using multi-tree Transition-based RRT (TRRT) [91]. Further works that combine the routing with motion planning have been studied for Autonomous Underwater Vehicles (AUV) planning [92]–[94]. The most similar problem to the proposed POP is the Prize Collection Traveling Salesman Problem (PC-TSP) used in [95] for modeling data collection planning for AUV. The method proposed for the PC-TSP uses an initial PRM navigation roadmap between target locations that is further used to steer the growth of a motion tree (a variant of the RRT) towards the solution of the PC-TSP found on the roadmap. The only known variant of the OP that considers environments with obstacles is the multi-vehicle Team Orienteering Problem presented in [59]. The method in [59] uses an occupancy grid-based approach for collision-free planning; however, it is capable of solving only small problem instances with fixed heading angles of Dubins vehicle.

The proposed method for the POP in the core publication [5c] combines the asymptotically optimal PRM* method for collision-free motion planning with the VNS-based method for the combinatorial OP part. The POP is introduced as the OP in configuration space and thus can be generalized for various configuration spaces such as the presented Dubins vehicle or 3D building environment. The proposed VNS-PRM* method uses an initial PRM* roadmap that is continuously expanded at every VNS iterations. The VNS part explores the combinatorial solution space of the POP on the current roadmap and tries to maximize the collected reward from the target locations. Furthermore, the VNS part selects the most rewarding solutions, with even a little over-budget cost, that are then more intensively sampled when expanding the PRM* roadmap. See the example solutions of the POP found using the proposed VNS-PRM* method in Fig. 2.5. We refer to Chapter 7 which is devoted to the core publication [5c].

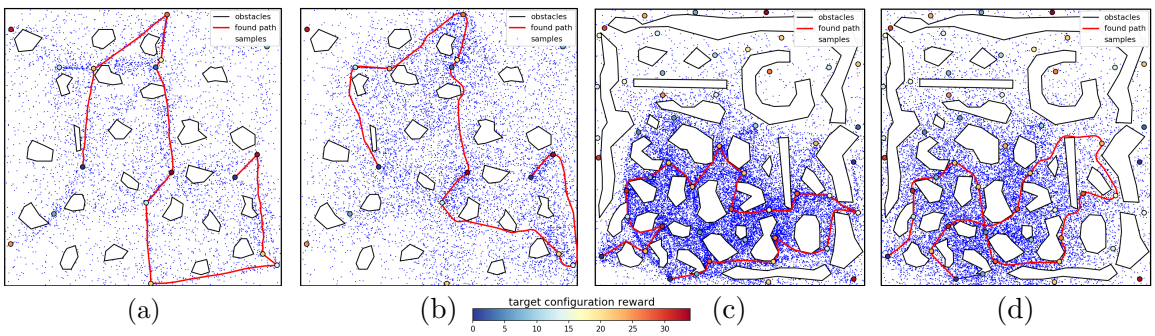


Figure 2.5: Example solutions of the POP on (a),(b) *potholes* and (c),(d) *dense* scenarios for (a),(c) Euclidean and (b),(d) Dubins vehicles [5c].

Chapter 3

Dubins Orienteering Problem

In this chapter, we present the first core publication called the Dubins Orienteering Problem (DOP) [1c] published in the IEEE Robotics and Automation Letters in 2017.

[1c] **R. Pěnička**, J. Faigl, P. Váňa, and M. Saska, "Dubins orienteering problem," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1210–1217, 2017

In the publication, we present a novel extension of the Orienteering Problem. The regular OP has the objective to maximize the collected reward from the given target locations, each with associated reward, while the tour starts and ends at the desired locations and is limited by a given budget. In the introduced DOP, the target locations are considered to be connected using the Dubins maneuvers [23] instead of straight line paths as in the regular OP. The Dubins maneuvers are feasible and more appropriate for the considered Unmanned Aerial Vehicle than the straight line paths with sharp corners at the target locations used in the OP. The non-holonomic fixed-wing UAVs require to use, e.g., the curvature constrained Dubins vehicle. For the vertical take-off and landing multi-rotor UAVs, the curvature constrained motion is also useful when considering constant velocity movement or vehicle with limited velocity and acceleration in a plane.

The proposed solution for the DOP is based on the Variable Neighborhood Search (VNS) [54]. Contrary to the OP with optimization of the subset of visited targets and their sequence, the DOP additionally requires to find the appropriate heading angles of Dubins vehicle at the target locations, which significantly influences the path length and thus its feasibility due to the budget constraint. The proposed VNS-based method for the DOP uses an equidistant sampling of heading angle at each target location to address the continuous optimization of finding appropriate heading angles. The method iteratively uses a combination of *shaking* procedure, to get an incumbent solution from possible local optimum, and *local search* procedure to possibly improve the so far best-found solution. Both procedures use predefined operators that change the subset selection and the order of target locations in the solution vector. The appropriate heading angle samples are found using graph search for the shortest path in a given solution vector.

The computational results show the feasibility of the proposed approach both in simulations and in a real experiment with hexarotor UAV. The method is verified on existing datasets from literature [96] for various turning radii of Dubins vehicle. We overview the computational requirements and solution quality for different sampling density of heading angle showing that 12 heading samples are sufficient for high-quality solutions while the computational requirements keep increasing with the number of samples. The used VNS operators are compared with more complex operators that, however, only increase computational requirements without improvement of solution quality. The VNS method for the DOP is finally compared with a straightforward combination of approaches for ordinary OP and DTSP. The decoupled solution uses a subset selection by solving the OP and then determines the appropriate heading angles by solving sampling based DTSP on the subset. The proposed VNS-based method is shown to produce higher quality solutions for most dataset instances than the decoupled approach.

The contribution on the publication of the author of this thesis is 55 %, including the implementation of the proposed method and writing the manuscript. The co-authors contributed by giving the initial impulse for this research, valuable feedback, and help with evaluations of the proposed method.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/LRA.2017.2666261

IEEE ROBOTICS AND AUTOMATION LETTERS. PREPRINT VERSION. ACCEPTED JANUARY, 2017 1

Dubins Orienteering Problem

Robert Pěnička, Jan Faigl, Petr Váňa, and Martin Saska

***Abstract*—In this paper, we address the Orienteering Problem (OP) for curvature constrained vehicle. For a given set of target locations, each with associated reward, the OP stands to find a tour from a prescribed starting location to a given ending location such that it maximizes collected rewards while the tour length is within a given travel budget constraint. The addressed generalization of the Euclidean OP is called the Dubins Orienteering Problem (DOP) in which the reward collecting tour has to satisfy the limited turning radius of the Dubins vehicle. The DOP consists not only of selecting the most valuable targets and determination of the optimal sequence to visit them, but it also involves the determination of the vehicle's heading angle at each target location. The proposed solution is based on the Variable neighborhood search technique, and its feasibility is supported by an empirical evaluation in existing OP benchmarks. Moreover, an experimental verification in a real practical scenario further demonstrates the necessity of the proposed direct solution of the Dubins Orienteering Problem.**

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

I. INTRODUCTION

IN this paper, we study a generalization of the Orienteering Problem (OP) [1] for curvature-constrained vehicles. The problem is called the Dubins Orienteering Problem (DOP), and its objective is to maximize the total collected rewards by visiting a subset of the given target locations by Dubins vehicle [2] while the length of the collecting tour does not exceed a given travel budget. The proposed generalization of the existing OP with Euclidean distance [3], further denoted as the Euclidean OP (EOP), is motivated by data collection scenarios with Unmanned Aerial Vehicles (UAVs) that can be modeled as the non-holonomic Dubins vehicle [4].

The Orienteering Problem can be considered as a variant of the Traveling Salesman Problem (TSP). In contrast to the TSP, in which the goal is to minimize the tour length to visit all the targets, the OP objective is to maximize the total sum of the collected rewards while the reward collecting tour does not exceed the specified travel budget. Thus, the OP is more suitable formulation for cases where visiting all the targets is unfeasible with the given travel budget.

In the EOP, the distance between the target locations corresponds to the length of the straight line segment connecting them and the objective is to select a maximal reward subset of target locations for which the length of the path visiting them is shorter or equal to the predefined maximal total path length.

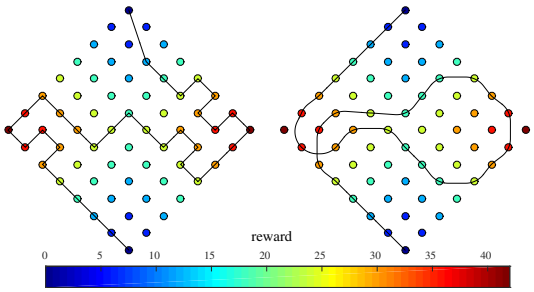


Fig. 1. Solutions of the Dubins Orienteering Problem on Set 64 for the budget $T_{max} = 50$ and different turning radii ρ. For $\rho = 0$, the problem becomes the ordinary EOP with the sum of the collected rewards $R = 900$ (on left), while for $\rho = 1.3$ the problem has to be directly solved as the DOP to satisfy T_{max} and the collected reward is $R = 714$ (on right). In both cases, the constructed path lengths are maximally 0.2 bellow the allowed T_{max}.

Although the objective in the DOP is similar to the EOP, i.e., to maximize the collected reward within the given travel budget, the final reward collecting path has to satisfy the limited curvature constraint, as shown in Fig. 1, and thus the final path consists of a sequence of optimal Dubins maneuvers [2] connecting the selected target locations. Therefore, a solution of the DOP requires determining particular heading angles at the target locations to minimize the length of Dubins maneuvers between the targets. Regarding computational complexity, the DOP can be considered as more challenging than the EOP as changing only one heading angle or target location in the reward collecting path usually enforces the change of all heading angles of nearby connected target locations.

A variant of the TSP with Dubins maneuvers [5] is known as the Dubins Traveling Salesman Problem (DTSP) [6]. Contrary to the DTSP which aims to minimize the total travel cost, the DOP allows to address the limited travel budget, and thus respects a practical deployment of UAVs with limited operational time. Therefore, we propose to directly solve the DOP, and our proposed solution is based on the Variable Neighborhood Search (VNS) metaheuristic for combinatorial optimization [7], which has been deployed to the OP in [8].

The paper is organized as follows. An overview of related work on the EOP and DTSP is presented in the next section. In Section III, the proposed DOP is formally introduced. Section IV presents the proposed direct solution of the DOP. Evaluation results together with the report on the method experimental deployment in a real-scale outdoor scenario are presented in Section V. Section VI concludes the paper.

Manuscript received: September, 10, 2016; Revised December, 15, 2016; Accepted January, 19, 2017.

This paper was recommended for publication by Editor Jonathan Roberts upon evaluation of the Associate Editor and Reviewers' comments. The presented work has been supported by the Czech Science Foundation (GAČR) under research project No. 16-24206S and research project No. 17-16900Y. Access to computing and storage facilities the National Grid Infrastructure MetaCentrum provided under the programme CESNET LM2015042 is greatly appreciated.

Authors are with the Czech Technical University, Faculty of Electrical Engineering, Technicka 2, 166 27, Prague, Czech Republic {penicrob|faiglj|vanapet1|saskam1}@fel.cvut.cz

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

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/LRA.2017.2666261

II. Related Work

The introduced Dubins Orienteering Problem (DOP) builds on the existing approaches for the Euclidean Orienteering Problem (EOP) [3] and Dubins Traveling Salesman Problem (DTSP) [9]. Therefore an overview of the existing approaches is presented in this section. Both the EOP [10] and DTSP [6] can be used for planning UAV missions; however, the EOP produces unfeasible paths for the Dubins vehicle, and the DTSP does not respect the travel budget.

The Euclidean OP has been studied since 1984 when Tsiligirides proposed two heuristics [11]. The first S-algorithm uses Monte Carlo method for picking the best solution from a large number of randomly generated paths with probabilities based on the reward per additional distance to the target location. The second D-algorithm uses a method for vehicle-scheduling with one depot by Wren and Holiday [12]. Tsiligirides further proposed a route-improvement algorithm that improves an initial route by using target insertion, target exchange, and 2-Opt operations [11].

A Four-Phase heuristic for the OP [1] uses insertion, improvement, and deletion phases to iteratively improve the path. In the insertion phase, new target locations are introduced to the path while using additional reward per distance and relaxed budget constraint. The second phase is based on 2-Opt and 3-Opt improvement operations. The deletion phase removes a target location with the minimal reward per distance and continues to the first phase with decreased relaxation of the travel budget. The fourth phase is the maximal insertion, and it follows after the iteration of previous three phases is terminated.

Chao et al. (1996) proposed a fast and effective heuristic for the OP in [13]. The heuristic considers only the target locations inside an ellipse with the foci in origin and ending locations with the major axis length equal to the travel budget. Using the most distant target locations from foci, a number of paths are generated during initialization with a greedy algorithm. The highest reward path $path_{op}$ is then improved by the Two-point exchange, i.e., by one-point move and 2-Opt operations, that systematically exchanges the target locations between $path_{op}$ and set of alternative paths $path_{nop}$ formed from unused target locations.

The Variable Neighborhood Search (VNS) metaheuristic [7] has been used to solve OP by Sevkli et al. (2006) [8]. This VNS-based method utilizes a predefined neighborhood structure, namely target insert/exchange and path insert/exchange operations. Using these four structures, the VNS algorithm iteratively performs *shake* and *local search* procedures. During the *shake* procedure, the currently best achieved solution is randomly changed to escape from a local minimum. In the *local search* procedure, the changed solution is searched within a smaller neighborhood structure to obtain a possibly better solution than the current best one.

Regarding the DTSP, the most relevant methods are the sampling based approaches [9], [14], [15] that allow a combinatorial optimization by using a discrete set of possible headings at the target locations. The DTSP stands to determine the minimum length path to visit all the target locations and satisfies the minimum curvature constraint of Dubins vehicle. The sampling based methods use a uniform sampling of the vehicle heading angle at each target location. The problem is then considered as the Generalized Asymmetric TSP that is further transformed and solved as the Asymmetric TSP (ATSP) [16], e.g., using Lin-Kernighan algorithm [17].

The closest existing problem formulation to the proposed DOP is the OP for kinodynamic vehicles outlined in [18]. Their solution of the Stochastic TSP and OP for kinodynamic vehicle is based on dividing the configuration space into cells with an equal volume, and merging the cells with no or few target locations into larger ones. In the TSP, the vehicle traverses each cell and collects the target locations inside by making small deviations from a fixed path that goes through all cells. For the OP, the vehicle selects a TSP sub-path with the highest reward. Even though the algorithm provides a possible approach to the DOP, it is useful mainly for the stochastic version of the OP where the target locations are randomly placed. In such a case, the algorithm provides an approximation of the optimal trajectory with a high probability. Moreover, the algorithm does not directly maximize the collected reward as the herein proposed method; it rather selects a part of the TSP path with the maximal reward and length below or equal to the budget.

The proposed solution of the introduced Dubins Orienteering Problem (DOP) is based on the VNS technique already deployed for the Euclidean OP in [8], which is actually one of the best performing methods for the EOP. The developed algorithm for the DOP is therefore compared with existing approaches for the EOP proposed by Chao et al. [13], Four-phase heuristic [1], and the original VNS-based method [8]. This comparison is done for the existing datasets by Tsiligirides [11] and two problem instances by Chao et al. [13]. Further experimental evaluation is performed for a practical scenario with a real UAV, see Section V.

III. Problem Statement

The motivation for the proposed Dubins Orienteering Problem (DOP) is in data collection scenarios for multirotor Unmanned Aerial Vehicles (UAVs) with limited operational time, where each of the target location requested to be visited has assigned a particular reward value, and the vehicle needs to follow a curvature-constrained path. The proposed solution can be however applied to any Dubins vehicle such as the fixed wings UAVs [19] or even the Ackermann vehicles. Hence, the objective is to find a data collection path for the Dubins vehicle that maximizes the sum of the collected rewards R such that the path length does not exceed the specified maximal travel budget T_{max}. The existing Euclidean OP [3] cannot be directly used in such scenarios as it produces unfeasible paths for the considered Dubins vehicle model and thus, it may lead to miss some of the target locations or violation of the budget constraint. The proposed DOP is a generalization of the Euclidean OP, and therefore, the EOP is formally introduced in the next section followed by its generalization for the Dubins vehicle in Section III-B.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/LRA.2017.2666261

A. Euclidean Orienteering Problem (EOP)

Having a set of target locations $S = \{s_1, \ldots, s_n\}$, the Orienteering Problem seeks to find a maximal reward subset $S_k \subseteq S$ and a path visiting S_k such that its length is limited by the given T_{max}. The origin and ending locations are given and denoted as s_1 and s_n. The subset selection in the problem, which determines the collected reward, is similar to the NP-hard Knapsack problem. The problem is also related to the NP-hard Traveling Salesman Problem (TSP) in finding a minimal-length path on S_k.

Each considered target location s_i is defined by its position denoted as $s_i \in \mathbb{R}^2$ (for simplicity and better readability) and its reward r_i. We assume that the reward of the origin and ending locations are zero $r_1 = r_n = 0$ and strictly positive for all other locations, i.e., $r_i > 0$ for $1 < i < n$. The EOP includes determination of k target locations defining the subset S_k and a sequence to their visits that can be described as a permutation $\Sigma = (\sigma_1, \ldots, \sigma_k)$, where $1 \leq \sigma_i \leq n$, $\sigma_i \neq \sigma_j$ for $i \neq j$ and $\sigma_1 = 1$, $\sigma_k = n$. For the Euclidean distance $\mathcal{L}_e(s_{\sigma_i}, s_{\sigma_j})$ between two locations s_{σ_i} and s_{σ_j}, the EOP can be formulated as the optimization problem:

$$\begin{aligned} &\underset{k,S_k,\Sigma}{\textit{maximize}}\ R = \sum_{i=1}^{k} r_{\sigma_i} \\ &\textit{subject to}\ \sum_{i=2}^{k} \mathcal{L}_e(s_{\sigma_{i-1}}, s_{\sigma_i}) \leq T_{max} \\ &\sigma_1 = 1, \sigma_k = n\ . \end{aligned} \tag{1}$$

B. Dubins Orienteering Problem (DOP)

The Dubins Orienteering Problem (DOP) is a generalization of the OP for the Dubins vehicle model to determine a feasible path over selected target locations S_k. The state of the Dubins vehicle $q = (x, y, \theta)$ consists of its position in plane $(x, y) \in \mathbb{R}^2$ and its heading $\theta \in \mathbb{S}^1$, i.e., $q \in SE(2)$. One of the specifics of this non-holonomic vehicle model is the minimal turning radius ρ that influences the length of the shortest path between two states. The kinematic model of Dubins vehicle with a constant forward velocity v and a control input u can be described as:

$$\dot{q} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = v \begin{bmatrix} \cos\theta \\ \sin\theta \\ \frac{u}{\rho} \end{bmatrix}, u \in [-1, 1]\ . \tag{2}$$

In [2], Dubins proved that for the model (2) the shortest path between two states consists only of straight line arc (S-segment) and arcs with the curvature ρ (turning left denoted as L-segment or right as the R-segment) and the optimal path is one of six possible maneuvers $\{LSL, LSR, RSL, RSR, LRL, RLR\}$ that are further denoted as Dubins maneuvers. For any two states q_i and q_j the Dubins maneuver together with its length $\mathcal{L}_d(q_i, q_j)$ can be determined analytically [2]; however, regarding the studied DOP, we need to determine the particular headings θ_i and θ_j of the vehicle at corresponding locations s_i and s_j, respectively.

Hence, each target location s_i is considered as the state $q_i = (s_i, \theta_i)$ in the DOP and in addition to the determination of the subset S_k of the k locations and the permutation $\Sigma = (\sigma_1, \ldots, \sigma_k)$, the DOP intends to find the corresponding heading angles $\Theta = (\theta_{\sigma_1}, \ldots, \theta_{\sigma_k})$. The Dubins Orienteering Problem for the model (2) can be then formulated as the optimization problem:

$$\begin{aligned} &\underset{k,S_k,\Sigma,\Theta}{\textit{maximize}}\ R = \sum_{i=1}^{k} r_{\sigma_i} \\ &\textit{subject to}\ \sum_{i=2}^{k} \mathcal{L}_d(q_{\sigma_{i-1}}, q_{\sigma_i}) \leq T_{max} \\ &\sigma_1 = 1, \sigma_k = n\ . \end{aligned} \tag{3}$$

In contrast to the Euclidean OP, the introduced DOP considers the Dubins vehicle model, and the path is constructed using the Dubins maneuvers between the adjacent target locations (states). Notice that the optimization problem (3) is not only over all possible subsets and respective permutations of the target locations (k, S_k, Σ), but also over all possible heading angles Θ at the target locations. This makes the problem computationally challenging as the already NP-hard EOP is extended to optimize over heading angles.

IV. Proposed Approach for the DOP

The proposed algorithm to solve the introduced Dubins Orienteering Problem (DOP) is based on the Variable Neighborhood Search (VNS) [7], which has been already deployed to the EOP in [8]. In contrast to the EOP, the DOP has to consider the heading angle at the target locations, which requires a new formulation of the solution search method. Therefore a brief overview of the VNS and the used approach for dealing with heading angles is provided prior detail description of the proposed VNS-based solution for the DOP.

The VNS is a metaheuristic proposed by Hansen and Mladenovic [7] for combinatorial optimization. The method operates on an initially defined Neighborhood structures $N(l_1, \ldots, l_{max})$, where l denotes the maximal distance between two solutions in the neighborhood. In the OP, the distance l is the number of different target locations inside the solution vector $(q_{\sigma_1}, \ldots, q_{\sigma_n})$. A set $N_l(x)$ contains all solutions in l distant neighborhood of the solution x. Particular Neighborhood structure is then expressed by an operation that changes the given solution within the desired distance.

Two main procedures are used in the VNS to search the solution space starting from an initial solution. In the *shaking* procedure, the incumbent solution x is randomly moved to different solution x' within the neighborhood. This is used to get farther from the current best solution which may be only a local minimum. Afterward, the *local search* procedure systematically searches for the best solution in the neighborhood of the solution x'. The solution from the *local search* becomes a new incumbent solution if it improves the current best solution. The procedures continue until stopping criterion is met, which is either a number of iterations, CPU time or maximal time between improvements.

For solving the DOP, we used the Randomized Variable Neighborhood Search (RVNS) variant of the VNS [7]. The RVNS algorithm uses a randomized *local search* procedure

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/LRA.2017.2666261

instead of the systematic approach used in the regular VNS. The randomized variant of the *local search* tries, during a predefined number of iterations, to randomly change the solution x' inside the Neighborhood structure to improve the solution by collecting more rewards. As it is shown for the EOP with the VNS [8], the RVNS is faster than the regular VNS and generates solutions that achieve the same rewards.

In the VNS, the Dubins Orienteering Problem is represented by a solution vector $(q_{\sigma_1}, \ldots, q_{\sigma_k}, q_{\sigma_{k+1}}, \ldots, q_{\sigma_n})$, where the first k target locations $(q_{\sigma_1}, \ldots, q_{\sigma_k})$ are within the budget constraint limit $\sum_{i=2}^{k} \mathcal{L}_d(q_{\sigma_{i-1}}, q_{\sigma_i}) \leq T_{max}$, $\sigma_1 = 1$ and $\sigma_k = n$. The remaining vector $(q_{\sigma_{k+1}}, \ldots, q_{\sigma_n})$ consists of all other target locations that are above the budget.

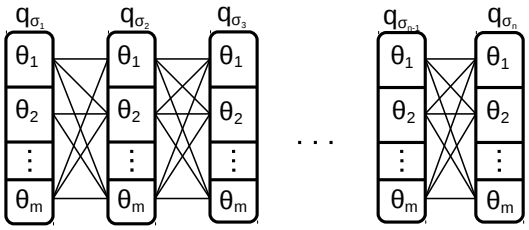


Fig. 2. Search graph of the DOP with m uniformly sampled heading angles at each target location q_{σ_i}, $0 \leq i \leq n$. For a particular selected sequence of the target locations $(q_{\sigma_1}, \ldots, q_{\sigma_n})$ a graph search over all heading values is performed to obtain headings providing the minimal path length.

Each state q_{σ_i} consists of the location $s_{\sigma i}$ and particular heading θ_{σ_i} that is selected from uniformly sampled angles from the interval $\theta \in \langle 0, 2\pi)$ into m samples $(\theta_1, \ldots, \theta_m)$.

The main difference of the proposed VNS-based DOP algorithm, compared to the existing variant for the EOP [8], is the determination of particular heading θ_{σ_i} at each target location. For a given number of samples m and a sequence of targets, the algorithm finds the shortest path by trying all possible combinations of sampled headings. The utilized search graph of the VNS DOP for a sequence of target locations $(q_{\sigma_1}, \ldots, q_{\sigma_n})$ is visualized in Fig. 2. A graph search is used to determine particular sequence of heading samples that produces path with the minimal length. A dynamic programming technique is utilized to store distances from the origin q_{σ_1} and ending q_{σ_k} locations to simplify further target location insertion/deletion.

In the proposed VNS method for the DOP, we utilize only a subset of reachable locations S_r such that $q_i \in S_r \Leftrightarrow (\mathcal{L}_d(q_1, q_i) + \mathcal{L}_d(q_i, q_n)) \leq T_{max}$ for any combination of sampled heading angles $(\theta_1, \theta_i, \theta_n)$. This selects all target locations that are reachable by the Dubins vehicle within the travel budget.

The initial solution x required for the VNS technique is generated using a greedy algorithm. For an initial zero reward Dubins path from q_1 to q_n, we iteratively add a new target location from S_r that minimizes additional distance per target reward as long as the length of the whole path is below T_{max}.

After an initial path P is found, the proposed VNS-based algorithm uses the following neighborhood structures in *shaking* and *local search* procedures to obtain solutions with higher rewards. The randomized *shaking* uses the structures:

- **Path Move** uses a randomly selected path $(q_{\sigma_i}, \ldots, q_{\sigma_j})$, where $1 < i < j < n$, from the solution vector $(q_{\sigma_1}, \ldots, q_{\sigma_n})$, and moves it to a randomly selected position $\sigma_o < i$ or $\sigma_o > j$. For the purpose of the VNS, this operation represents neighborhood $l = 1$ despite the fact that the number of different target locations is usually larger than one.
- **Path Exchange** also uses a randomly selected path $(q_{\sigma_i}, \ldots, q_{\sigma_j})$ from the solution vector, but exchange the path with a second random non-overlapping path $(q_{\sigma_o}, \ldots, q_{\sigma_p})$. The path exchange operation represents the neighborhood $l = 2$.

The *local search* procedure employs different and much closer neighborhoods. Unlike the *shaking*, the *local search* procedure uses an iterative search in the particular neighborhood such that it tries numerous operations on the same solution. For the RVNS, the *local search* tries random operations for a number of times that is equal to the square of the number of the target locations. This ensures that the neighborhood of solution x' from *shaking* is searched more deeply for local optima than in the *shaking* procedure. The procedure uses the following neighborhood structures.

- **One Point Move** corresponds to the $l = 1$ neighborhood in which only one randomly selected target is moved to a different position within the solution vector.
- **One Point Exchange** is a farther neighborhood $l = 2$ and it uses two randomly selected distinct targets from the solution vector and exchanges their positions.

In all four presented neighborhood structures, the operations also search through all sampled heading angles as described above, to minimize the solution path length for a particular sequence of the target locations.

The proposed VNS-based algorithm for the DOP is summarized in Algorithm 1. For brevity, the rewards collected by a path $P = P(k, S_k, \Sigma, \Theta)$ is $R(P) = \sum_{i=1}^{k} r_{\sigma_i}$, $\sigma_i \in \Sigma$, and the path length is $\mathcal{L}_d(P) = \sum_{i=2}^{k} \mathcal{L}_d(q_{\sigma_{i-1}}, q_{\sigma_i})$.

Algorithm 1: Variable Neighborhood Search for the DOP

Input : S – set of target locations
Input : T_{max} – maximal allowed budget
Input : m – number of heading values for each target
Output: P – found data collecting path

```
S_r ← getReachableLocations(S)
P ← createInitialPath(S_r,T_max)                    // greedy
while stopping condition is not met do
    l ← 1
    while l ≤ l_max do
        P' ← shake(P, l)
        P'' ← localSearch(P', l)
        if L_d(P'') ≤ T_max and R(P'') > R(P) then
            P ← P''
            l ← 1
        else
            l ← l + 1
```

The VNS-based DOP algorithm uses only two neighborhood structures for both *shaking* and *local search*, which means that the maximal neighborhood distance is $l_{max} = 2$.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/LRA.2017.2666261

An extension of the neighborhood structures to $l_{max} > 2$ is possible by concurrently moving more than two target locations in the *local search*.

Evaluation results of the proposed DOP method are presented in the next section together with the comparison to the existing algorithms for the EOP. Besides, results from the real practical experiments with UAV are presented as well.

V. Results

The proposed method for the Dubins Orienteering Problem (DOP) has been evaluated on five existing datasets from the literature [20] and also in real data collection scenario with Unmanned Aerial Vehicle (UAV). Using the existing datasets, the proposed VNS-based method is compared with existing Euclidean Orienteering Problem (EOP) approaches as to the best of our knowledge there is no existing solution for the introduced DOP. The maximal achieved rewards for particular non-zero turning radii are presented alongside to show the influence of increasing turning radius on the collected reward. Furthermore, the real experiment with hexarotor UAV is presented. The results show practical applicability of the introduced DOP and the proposed VNS-based method for robotic data collection planning.

A. Results on datasets and existing EOP approaches

A comparison of the proposed DOP method with solutions for the EOP, namely with the heuristic proposed by Chao et al. [13], 4-phase heuristic by Ramesh et al. [1], and VNS-based algorithm by Sevkli et al. [8] has been performed. Abbreviation of the methods and used existing benchmarks are in Table I. Results for the proposed method are presented for multiple representative turning radii $\rho \in \{0, 0.3, 0.5, 0.7, 0.9, 1.1, 1.3\}$, where $\rho = 0$ is a solution of the EOP.

TABLE I
Abbreviation related with the results

Set 1, **Set 2**, **Set 3**	Test instances created by Tsiligirides [11].
Set 64, **Set 66**	Test instances proposed by Chao [13].
4Phase	Four-Phase heuristic for EOP by Ramesh et al. [1].
Chao	Fast and effective heuristic by Chao et al. [13].
VNS	VNS-based algorithm by Sevkli et al. [8].
VNS DOP	Proposed Dubins Orienteering VNS method.

The utilized VNS is a stochastic procedure, and therefore, each scenario has been solved 10 times for each budget T_{max}. The results were computed using a single core of Intel i7 3.4GHz CPU. A single sample of the heading angle $m = 1$ has been used for the DOP problems with $\rho = 0$ as the heading angle does not influence the distance between the target locations. For $\rho > 0$, equidistant sampling of the heading angle into $m = 16$ values has been utilized. The stopping criterion is the maximal number of 10 000 iterations with the maximal 3 000 iterations without improvement.

Results for Tsiligirides datasets **Set 1**, **Set 2**, **Set 3** are presented in Tables II, III and IV, respectively. Tables V and VI show results for Chao datasets **Set 64** and **Set 66**. The presented results are the maximal achieved collected rewards R from the all 10 runs for the particular problem and budget.

TABLE II
Results comparison for Set 1

$\mathbf{T_{max}}$	Chao	4Phase	Proposed VNS-based DOP						
			ρ=0.0	ρ=0.3	ρ=0.5	ρ=0.7	ρ=0.9	ρ=1.1	ρ=1.3
5	10	10	10	10	10	0	0	0	0
10	15	15	15	15	15	15	15	15	15
15	45	45	45	45	45	45	45	40	35
20	65	65	65	65	60	60	60	60	50
25	90	90	90	90	85	85	85	85	75
30	110	110	110	110	110	105	105	105	95
35	135	135	135	135	135	130	130	120	120
40	155	150	155	155	155	150	145	140	140
46	175	175	175	175	175	170	170	165	160
50	190	180	190	185	185	185	175	175	165
55	205	205	205	200	200	195	195	185	185
60	225	225	220	220	220	215	215	205	205
65	240	240	240	240	235	235	235	225	220
70	260	260	260	260	255	250	250	240	235
73	265	265	265	265	265	260	260	250	240
75	270	275	270	270	265	265	260	255	245
80	280	280	280	280	275	275	270	265	255
85	285	285	285	285	285	280	275	270	265

TABLE III
Results comparison for Set 2

$\mathbf{T_{max}}$	Chao	4Phase	Proposed VNS-based DOP						
			ρ=0.0	ρ=0.3	ρ=0.5	ρ=0.7	ρ=0.9	ρ=1.1	ρ=1.3
15	120	120	120	120	120	115	115	115	95
20	200	200	200	190	190	180	175	165	135
23	210	210	210	205	200	200	200	200	160
25	230	230	230	230	220	220	205	200	165
27	230	230	230	230	230	230	230	220	180
30	265	260	265	260	255	255	240	230	225
32	300	300	300	290	290	275	275	260	240
35	320	320	310	320	315	310	300	285	285
38	360	385	360	350	345	340	330	325	310
40	395	395	395	385	375	375	365	355	335
45	450	450	450	440	440	420	410	395	370

TABLE IV
Results comparison for Set 3

$\mathbf{T_{max}}$	Chao	4Phase	Proposed VNS-based DOP						
			ρ=0.0	ρ=0.3	ρ=0.5	ρ=0.7	ρ=0.9	ρ=1.1	ρ=1.3
15	170	170	170	170	160	160	160	150	140
20	200	200	200	190	190	180	180	180	180
25	260	260	260	260	260	260	250	250	230
30	320	320	320	320	320	320	320	310	300
35	390	390	390	380	380	360	380	380	360
40	430	430	430	430	420	420	420	420	400
45	470	470	470	470	460	450	450	450	440
50	520	520	520	520	510	510	470	500	490
55	550	550	550	550	540	540	530	530	520
60	580	580	580	580	570	560	560	560	550
65	610	610	610	610	600	590	590	590	580
70	640	640	640	640	630	610	610	600	610
75	670	670	670	670	650	650	640	630	630
80	710	710	700	700	690	680	680	670	670
85	740	740	740	730	730	700	700	710	710
90	770	770	770	760	760	740	750	710	730
95	790	790	790	790	780	780	770	760	750
100	800	800	800	800	790	790	790	780	770
105	800	800	800	800	800	800	800	800	790
110	800	800	800	800	800	800	800	800	800

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/LRA.2017.2666261

TABLE V
RESULTS COMPARISON FOR SET 64

T_{max}	Chao	VNS	Proposed VNS-based DOP						
			ρ=0.0	ρ=0.3	ρ=0.5	ρ=0.7	ρ=0.9	ρ=1.1	ρ=1.3
15	96	96	96	96	96	96	96	96	96
20	294	294	294	294	294	294	252	252	252
25	390	390	390	390	384	366	360	300	300
30	474	474	474	468	468	468	468	408	390
35	570	576	576	570	570	564	546	498	492
40	714	714	714	696	696	690	672	582	582
45	816	816	816	798	792	780	756	642	636
50	900	900	900	888	882	876	834	708	714
55	984	984	984	978	960	960	924	804	786
60	1044	1062	1062	1044	1026	1026	1008	834	834
65	1116	1116	1116	1098	1098	1098	1080	918	900
70	1176	1188	1188	1170	1170	1134	1134	990	960
75	1224	1236	1236	1230	1206	1194	1194	1038	1014
80	1272	1272	1272	1272	1260	1254	1224	1074	1080

TABLE VI
RESULTS COMPARISON FOR SET 66

T_{max}	Chao	VNS	Proposed VNS-based DOP						
			ρ=0.0	ρ=0.3	ρ=0.5	ρ=0.7	ρ=0.9	ρ=1.1	ρ=1.3
5	10	10	10	10	10	10	0	0	0
10	40	40	40	40	40	40	40	40	40
15	120	120	120	100	100	100	100	95	95
20	195	205	205	205	200	195	195	195	170
25	290	290	280	290	280	280	280	275	260
30	400	400	400	400	380	370	370	370	370
35	460	465	465	465	465	460	455	450	445
40	575	575	575	570	570	570	545	540	535
45	650	650	650	645	650	650	645	640	640
50	730	730	730	725	725	710	710	695	690
55	825	825	825	825	825	800	820	795	790
60	915	915	915	895	895	895	890	890	860
65	980	980	980	980	930	925	950	945	945
70	1070	1070	1070	1065	1030	1070	1070	1070	1035
75	1140	1140	1140	1140	1120	1110	1080	1085	1090
80	1215	1215	1215	1195	1190	1170	1175	1165	1155
85	1270	1270	1270	1270	1245	1260	1245	1235	1200
90	1340	1340	1340	1320	1320	1305	1295	1295	1295
95	1380	1395	1395	1395	1390	1370	1370	1360	1320
100	1435	1465	1465	1445	1445	1435	1420	1420	1390
105	1510	1520	1520	1495	1505	1495	1485	1470	1445
110	1550	1560	1550	1550	1550	1545	1545	1530	1505
115	1595	1595	1590	1580	1580	1580	1575	1555	1550
120	1635	1635	1625	1625	1625	1610	1600	1595	1575
125	1655	1670	1670	1655	1655	1645	1640	1640	1620
130	1680	1680	1680	1680	1675	1675	1670	1670	1655

Presented results show that the proposed VNS-based DOP algorithm provides competitive results to the existing EOP approaches for the turning radius $\rho = 0$. Nevertheless in some test instances for $\rho = 0$ the DOP does not provide the best known results due to the fact that the most rewarded solutions are in terms of number of different nodes very far from the previously found result with a slightly lower budget. However, the proposed algorithm solves the DOP, which is not possible by existing methods for the EOP. For increasing ρ, the collected reward decreases for almost all problem instances. This indicates that increasing turning radius results in longer paths, and thus solutions provided by the EOP approaches would violate the budget constraint for Dubins vehicle. The computational time to find the maximal achieved rewards and the number of iterations needed to obtain the solutions of EOP and DOP using the proposed VNS-based algorithm are shown in Fig. 3.

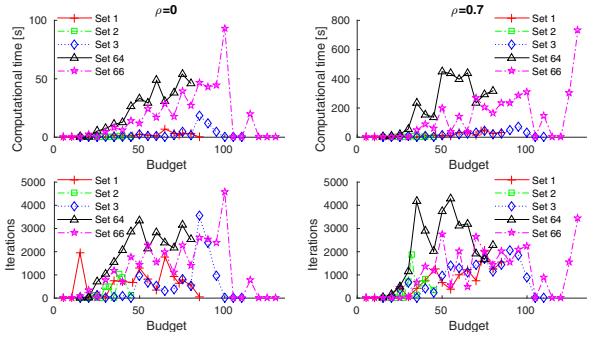


Fig. 3. Computational time and number of iterations for the EOP (DOP with $\rho = 0$), on the left, and the DOP with $\rho = 0.7$, on the right.

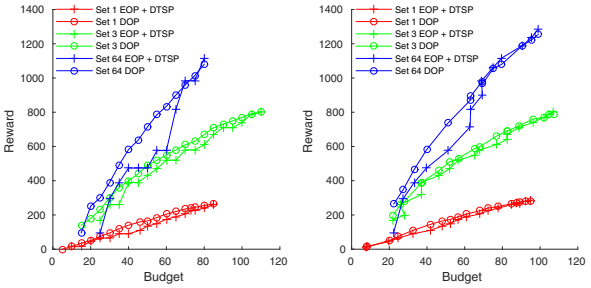


Fig. 4. A comparison of the proposed DOP algorithm with the subset selection by the EOP and finding the path as a solution of the DTSP. The same parameters $m = 16$ and $\rho = 1.3$ are used in both cases. Two strategies are considered for the EOP+DTSP approach. The results on the left are obtained for an iterative decrease of the budget for the EOP such that the solution of the DTSP meets the original travel budget. For the results on the right, the length of the path obtained by the EOP+DTSP is considered as a new travel budget in the proposed DOP algorithm.

A further comparison of the proposed direct solution of the DOP with existing approaches for the EOP is based on a straightforward combination of solving the EOP and Dubins Traveling Salesman Problem (DTSP). This naive approach is based on finding the subset of target locations S_k, with the highest collected reward, by solving the EOP. The sampling-based solution of the DTSP [9] is then used to find the data collecting path for the subset S_k with respect to the sampling of the heading angle m. The results are shown in Fig. 4, where the plot on the left shows that by using a smaller budget for the EOP and afterward the found S_k in the DTSP leads (in most cases) to lower rewards than a direct solution of the DOP. On the other hand, the right plot in Fig. 4 shows that in most cases (especially for lower budgets) the rewards collected by the solution of the DOP is higher. These results support suitability of the proposed algorithm for the introduced Dubins Orienteering Problem. Hence, it is not beneficial to solve the DOP by a separate selection of the target locations, e.g., by solving the EOP, and consecutive path planning for the Dubins vehicle. Solving the EOP may provide equally rewarded paths with multiple different subsets of the target locations. However, some of the subsets can be connectable in the consequent DTSP respecting the budget constraint, but some may not.

The proposed VNS DOP algorithm uses m sampled head-

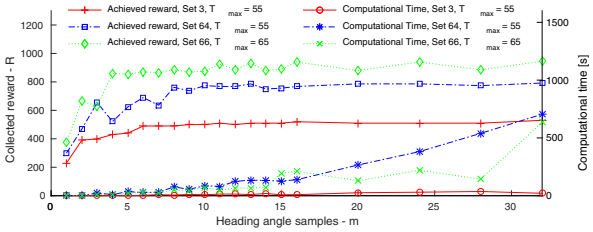


Fig. 5. Sums of collected rewards R and computational time for different heading sample rate m with $\rho = 1.2$.

ing values at each target location. A particular number m influences the path length and higher number of samples may almost always produce shorter paths and thus, a high reward collected for a given travel budget T_{max}. An influence of m on the sum of the collected rewards R and the computational time on m for the selected problems is shown in Fig. 5. The results show that R tends to increase until $m = 12$. This is caused by the fact that the main objective of the DOP optimization is the sum of collected rewards R and the path length is not important as far as it is shorter than T_{max}.

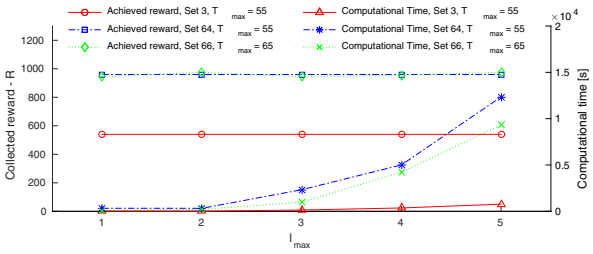


Fig. 6. Computational time and collected rewards of the proposed VNS-based DOP algorithm for increasing value of the maximal neighborhood distance l_{max} for particular problems.

The proposed VNS-based DOP uses the maximal neighborhood distance $l_{max} = 2$ but the value can be increased by concurrently moving $l_{max} > 2$ target locations in the *local search* procedure. Fig. 6 with the computational times and collected reward for different l_{max} shows that the solution convergence is slower for increased l_{max} because a single iteration lasts longer and the randomized RVNS algorithm does not benefit from the enlarged neighborhood distance.

B. Real experiments

The proposed method has been experimentally evaluated in the real data collection scenario with a hexarotor UAV. [1] The UAV is requested to visually inspect as many high rewarded target locations as possible during the length-limited flight. The considered scenario consists of 20 target locations, where a particular colored object with marked reward is located. The objects are placed in the area of approximately 100×50 m large. Fig. 7 shows the colored target object with the used hexarotor UAV, originally developed for multi-robot applications [21]. The considered travel budget is $T_{max} = 150$ m for

[1]We refer to http://mrs.felk.cvut.cz/icra17dop for more information about the experiment.

Fig. 7. Hexarotor UAV during the visual data collection of the colored target object with displayed reward.

which the UAV has to visit the locations of the objects and maximize the collected reward.

Although the hexarotor UAV can drive through a path from the Euclidean OP, in certain cases, it is then required to decelerate during the sharp turns. Therefore, the hexarotor UAV modeled as the Dubins vehicle with a smooth path over the target locations allow using constant speed trajectories. Moreover, the Dubins model respects the real constraints of the UAV such as the maximal speed and acceleration. This allows to the used onboard trajectory controller [22] to precisely navigate through the trajectory without missing the target location which can happen for the path produced by solving the related EOP.

The crucial parameter of the Dubins vehicle is the minimal turning radius ρ that is computed from the desired constant velocity v_c and the maximal acceleration of the UAV a_{max}. The equation of circular motion with constant speed $\rho = v_c^2/a_{max}$ is used to get the radius, which produces the maximal allowed acceleration during the circular parts of the path. The constant velocity $v_c = 4$ m.s^{-1} and the maximal allowed acceleration $a_{max} = 2.6$ m.s^{-2} has been used and the considered turning radius ρ is $\rho = 6.15$ m.

Paths found by the proposed DOP algorithm for the turning radius $\rho = 0$ and $\rho = 6.15$ m are shown in Fig. 8. A solution is found within a second using the same parameters as in Section V-A. The particular rewards of the found solutions are $R = 71$ and $R = 65$, for $\rho = 0$ and $\rho = 6.15$ m respectively, with total path lenghts of 149.0 m and 148.4 m. Although the solution for $\rho = 0$ provides a higher reward, the path is not feasible for the constant speed motion, and the onboard controller has to violate the planned path. This causes cutting of sharp turns to fulfill the schedule of the plan as it is shown for the "`EOP path traveled by UAV`" curve in Fig. 8. On the other hand, a solution of the DOP with $R = 65$ respects the maximal acceleration with the desired constant speed of the vehicle and all target objects have been successfully captured.

VI. CONCLUSIONS

This paper introduces a generalization of the Orienteering Problem to the Dubins vehicle that is called the Dubins Orienteering Problem (DOP). We propose a novel Variable Neighborhood Search (VNS) based method for solving this challenging problem. A sampling based approach is used

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/LRA.2017.2666261

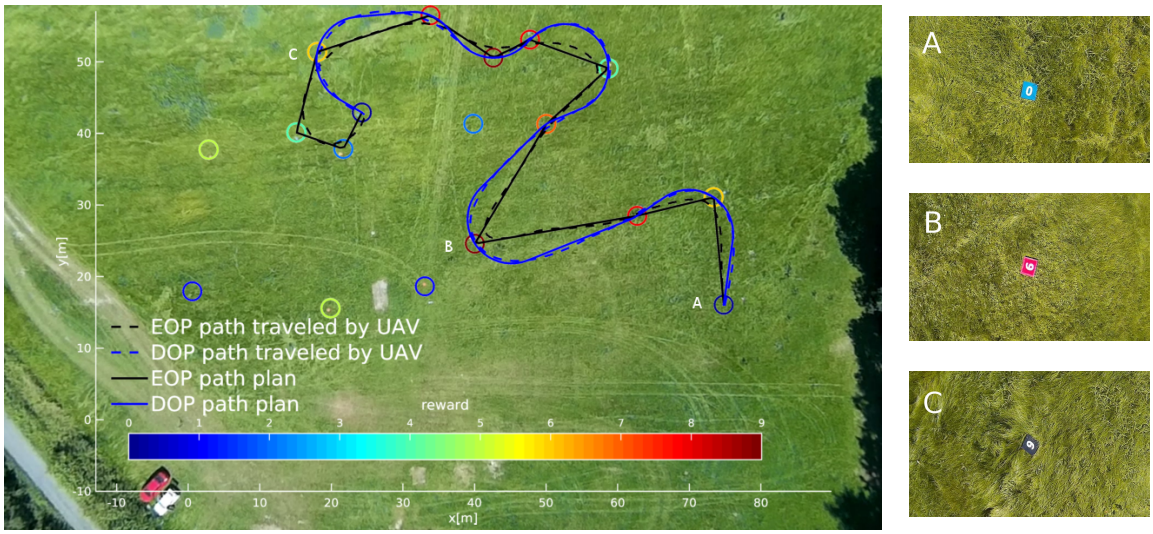


Fig. 8. Plots of the UAV position for solutions of the data collection scenario as the DOP and EOP.

to search for an appropriate sequence of heading angles at the target locations. The presented results indicate that for zero turning radius, the proposed DOP solver is competitive to existing methods for the Euclidean OP. Results for non-zero turning radius show that the collected reward decreases with the increasing radius. Moreover, the presented results demonstrate that a solution of the DOP as a combination of the Euclidean OP and consecutive Dubins Traveling Salesman Problem is not plausible. We also show that the sampling based approach to heading angles is viable as the prime objective of the DOP is to maximize the collected reward and a higher number of samples does not necessarily increase the quality of solution (the collected reward). Finally, results from the real deployment of the proposed approach further demonstrate a necessity of the proposed direct solution of the Dubins Orienteering Problem. For future work, we intend to investigate the OP for other more complex maneuvers such as splines, and to extend the DOP for possible data collection within proximity of the target locations.

REFERENCES

[1] R. Ramesh and K. M. Brown, "An efficient four-phase heuristic for the generalized orienteering problem," *Computers & Operations Research*, vol. 18, no. 2, pp. 151–165, Feb. 1991.
[2] 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*, vol. 79, no. 3, pp. 497–516, Jul. 1957.
[3] P. Vansteenwegen, W. Souffriau, and D. V. Oudheusden, "The orienteering problem: A survey," *European Journal of Operational Research*, vol. 209, no. 1, pp. 1 – 10, Feb. 2011.
[4] Y. Lin and S. Saripalli, "Path planning using 3d dubins curve for unmanned aerial vehicles," in *ICUAS*, May 2014, pp. 296–304.
[5] J. Faigl and P. Váňa, "Self-organizing map for the curvature-constrained traveling salesman problem," in *ICANN*, Sep. 2016, pp. 497–505.
[6] K. Savla, E. Frazzoli, and F. Bullo, "Traveling salesperson problems for the dubins vehicle," *IEEE Transactions on Automatic Control*, vol. 53, no. 6, pp. 1378–1391, July 2008.
[7] P. Hansen and N. Mladenovi, "Variable neighborhood search: Principles and applications," *European Journal of Operational Research*, vol. 130, no. 3, pp. 449–467, May 2001.
[8] Z. Sevkli and F. E. Sevilgen, "Variable neighborhood search for the orienteering problem," in *21th International Symposium on Computer and Information Sciences (ISCIS)*, 2006, pp. 134–143.
[9] J. Ny, E. Feron, and E. Frazzoli, "On the dubins traveling salesman problem," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 265–270, Jan. 2012.
[10] J. Yu, M. Schwager, and D. Rus, "Correlated orienteering problem and its application to persistent monitoring tasks," *IEEE Transactions on Robotics*, vol. 32, no. 5, pp. 1106–1118, Oct. 2016.
[11] T. Tsiligirides, "Heuristic methods applied to orienteering," *The Journal of the Operational Research Society*, vol. 35, no. 9, pp. 797–809, Sep. 1984.
[12] A. H. Anthony Wren, "Computer scheduling of vehicles from one or more depots to a number of delivery points," *Operational Research Quarterly (1970-1977)*, vol. 23, no. 3, pp. 333–344, Sep. 1972.
[13] I.-M. Chao, B. L. Golden, and E. A. Wasil, "A fast and effective heuristic for the orienteering problem," *European Journal of Operational Research*, vol. 88, no. 3, pp. 475–489, Feb. 1996.
[14] P. Oberlin, S. Rathinam, and S. Darbha, "A transformation for a heterogeneous, multiple depot, multiple traveling salesman problem," in *ACC*, June 2009, pp. 1292–1297.
[15] J. T. Isaacs, D. J. Klein, and J. P. Hespanha, "Algorithms for the traveling salesman problem with neighborhoods involving a dubins vehicle," in *ACC*, June 2011, pp. 1704–1709.
[16] C. E. Noon and J. C. Bean, "An efficient transformation of the generalized traveling salesman problem," *INFOR: Information Systems and Operational Research*, vol. 31, no. 1, pp. 39–44, 1993.
[17] K. Helsgaun, "An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic," *European Journal of Operational Research*, vol. 126, no. 1, pp. 106–130, 2000.
[18] A. Adler and S. Karaman, "The stochastic traveling salesman problem and orienteering for kinodynamic vehicles," in *ICRA*, May 2016, pp. 2788–2795.
[19] P. B. Sujit, S. Saripalli, and J. B. Sousa, "Unmanned aerial vehicle path following: A survey and analysis of algorithms for fixed-wing unmanned aerial vehicless," *IEEE Control Systems*, vol. 34, no. 1, pp. 42–59, Feb. 2014.
[20] "The Orienteering Problem: Test Instances – Department of Mechanical Engineering, University of Leuven," citied on 2016-08-20. [Online]. Available: http://www.mech.kuleuven.be/en/cib/op#section-2
[21] M. Saska, V. Vonásek, T. Krajník, and L. Přeučil, "Coordination and navigation of heterogeneous mav–ugv formations localized by a hawk-eye-like approach under a model predictive control scheme," *The International Journal of Robotics Research*, vol. 33, no. 10, pp. 1393–1412, July 2014.
[22] T. Baca, G. Loianno, and M. Saska, "Embedded model predictive control of unmanned micro aerial vehicles," in *MMAR*, Aug. 2016, pp. 992–997.

Chapter 4

Data Collection Planning with Non-zero Sensing Distance for a Budget and Curvature Constrained Unmanned Aerial Vehicle

This chapter presents the second core publication [2c] with VNS-based method for the Dubins Orienteering Problem with Neighborhoods (DOPN) published in the Autonomous Robots journal.

[2c] **R. Pěnička**, J. Faigl, M. Saska, and P. Váňa, "Data collection planning with non-zero sensing distance for a budget and curvature constrained unmanned aerial vehicle," *Autonomous Robots*, vol. 43, no. 8, pp. 1937 –1956, 2019

The manuscript introduces an extended version of a solution approach for the DOPN initially proposed in a conference paper [9a]. The DOPN is a generalization of the DOP where each target location has a disk-shaped neighborhood [8a], where the data can be collected without visiting a precise position of the targets. The generalization can be motivated by, e.g., data collection from Wireless Sensor Network where the measured data can be retrieved from the sensors placed in the environment by the UAV using wireless communication [29]. Another motivation scenario is data collection with UAV equipped with a long-range sensor, e.g., a wide field-of-view camera that does not require to visit the target locations precisely and thus can save the travel cost, which in turn allows to increase the amount of collected data.

In the conference paper [9a], the initial solution of the DOPN is proposed by a straightforward extension of the VNS for the DOP. The considered circular neighborhoods are equidistantly sampled on their border, and the DOPN is solved similarly to the DOP in [1c] with both heading and neighborhood statically sampled. In the related core publication [2c], the continuous optimization of finding positions within the neighborhoods and vehicle heading angles is significantly improved. The proposed VNS-based method uses the same *shaking* and *local search* procedures as for the DOP in [1c]; however, it uses a low-dense initial equidistant sampling. Additionally, new VNS operators are proposed to perform local continuous optimization of both the heading angle and neighborhood position samples to shorten the current solution. The continuous optimization operators are based on the LIO [56] technique. The Waypoint Shake operator performed in the *shaking* procedure randomizes the heading angles and neighborhood positions, while in the *local search*, the Waypoint Improvement operator performs the LIO. Such optimized heading and neighborhood samples are iteratively added to the initially low-dense graph of samples, which is then used by the combinatorial VNS operators.

The computational results on existing datasets show significant improvement of the solution quality compared to the static high-dense sampling of the neighborhood and heading angles in [9a]. Furthermore, the computational time required to achieve a certain solution quality is decreased when the continuous optimization operators are used. The proposed method is also shown to produce better solutions for tested instances than the SOM-based approach [10a]. Finally, the proposed method is experimentally verified in a visual data collection scenario. A real hexarotor UAV with a high-resolution wide field of view camera was employed in an outdoor environment to visually inspect targets in predefined positions.

The contribution of the author of this thesis on the manuscript is 65 %, with co-authors giving feedback to improve the method and the manuscript.

This is a post-peer-review, pre-copyedit version of an article published in Autonomous Robots.
The final authenticated version is available online at: http://dx.doi.org/10.1007/s10514-019-09844-5

Data Collection Planning with Non-zero Sensing Distance for a Budget and Curvature Constrained Unmanned Aerial Vehicle

Robert Pěnička · Jan Faigl · Martin Saska · Petr Váňa

Received: 24 June 2018 / Accepted: 15 February 2019

Abstract Data collection missions are one of the many effective use cases of Unmanned Aerial Vehicles (UAVs), where the UAV is required to visit a predefined set of target locations to retrieve data. However, the flight time of a real UAV is time constrained, and therefore only a limited number of target locations can typically be visited within the mission. In this paper, we address the data collection planning problem called the Dubins Orienteering Problem with Neighborhoods (DOPN), which sets out to determine the sequence of visits to the most rewarding subset of target locations, each with an associated reward, within a given travel budget. The objective of the DOPN is thus to maximize the sum of the rewards collected from the visited target locations using a budget constrained path between predefined starting and ending locations. The variant of the Orienteering Problem (OP) addressed here uses curvature-constrained Dubins vehicle model for planning the data collection missions for UAV. Moreover, in the DOPN, it is also assumed that the data, and thus the reward, may be collected from a close neighborhood sensing distance around the target locations, e.g., taking a snapshot by an onboard camera with a wide field of view, or using a sensor with a long range. We propose a novel approach based on the Variable Neighborhood Search (VNS) metaheuristic for the DOPN, in which combinatorial optimization of the sequence for visiting the target locations is simultaneously addressed with continuous optimization for finding Dubins vehicle waypoints inside the neighborhoods of the visited targets. The proposed VNS-based DOPN algorithm is evaluated in numerous benchmark instances, and the results show that it significantly outperforms the existing methods in both solution quality and computational time. The practical deployability of the proposed approach is experimentally verified in a data collection scenario with a real hexarotor UAV.

Keywords Unmanned Aerial Vehicles · Nonholonomic Motion Planning · Data Collection Planning · Orienteering Problem

1 INTRODUCTION

Unmanned Aerial Vehicles (UAV) are effective systems for long-range data collection (Ergezer and Leblebicioğlu 2014) or for information gathering scenarios (Nguyen et al. 2016), where a UAV has to gather data from specified locations in the environment. Such a scenario consists of a UAV equipped with an onboard sensor that is required to reach particular target locations and measure or collect the desired data. For example, in a Wireless Sensor Network (WSN), the sensors are placed in the environment, and the UAV can be used for retrieving the measured data from the sensor units by wireless communication with a limited range (Jawhar et al. 2014; Wang et al. 2015). Hence, the objective of data collection planning can be to minimize the required time to retrieve the requested data (i.e., to

This work was supported by the Czech Science Foundation under research project No. 16-24206S, No. 17-16900Y, and No. 19-20238S. The authors acknowledge the support of the OP VVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics". Access to computing at National Grid Infrastructure MetaCentrum, provided under the CESNET LM2015042 programme, is greatly appreciated. The support of CTU in Prague grant No. SGS17/187/13 is also gratefully acknowledged.

Robert Pěnička · Jan Faigl · Martin Saska · Petr Váňa
Czech Technical University, Faculty of Electrical Engineering, Technicka 2, 166 27, Prague, Czechia
E-mail: {penicrob | faiglj | saskam1 | vanapet1}@fel.cvut.cz

minimize the length of the data collection path) or/and to maximize the information collected by a single path.

Data collection planning can be formulated as a variant of the Traveling Salesman Problem (TSP) (Oberlin et al. 2010), where a path visiting all the given locations with minimal length is to be found. However, the required visits to all locations may not be possible with the budget limitation of a real vehicle (limited flight time).

Nowadays, the typical flight time of a small UAV is limited to tens of minutes, and the time is further decreased if the UAV is equipped with an additional payload, e.g., onboard sensors. Therefore, the Orienteering Problem (OP) (Tsiligirides 1984) formulation seems to be more suitable for data collection planning with a limited travel budget. Rather than minimizing the path length as in the TSP, the OP set out to find a path maximizing the sum of the rewards collected from a selected subset of target locations that can be reached using the given travel budget.

In this work, we consider that the data collecting vehicle with a budget constraint has to follow a curvature-constrained path, and thus we model the UAV as Dubins vehicle (Dubins 1957). Dubins vehicle can be used for modeling car-like robots (Tokekar et al. 2014), fixed-wing aerial vehicles (Lugo-Cárdenas et al. 2014) or Vertical Take-Off and Landing (VTOL) multirotor UAVs traversing the planned path at a constant speed (Pěnička et al. 2017a).

For Dubins vehicle, the TSP becomes the Dubins Traveling Salesman Problem (DTSP) (Savla et al. 2005), where it is required to find not only the optimal sequence for the visits to all target locations, but also optimal heading angles of the vehicle at the locations, as they greatly influence the final path length. Since each heading angle can be arbitrarily selected from 0 to 2π, the problem becomes computationally demanding due to the required non-linear continuous optimization of the additional dimension of the heading angles.

For a limited travel budget and Dubins vehicle, the OP becomes the Dubins Orienteering Problem (DOP), which was introduced and solved by a Variable Neighborhood Search (VNS) based approach in (Pěnička et al. 2017a). In the DOP, it is required to search over all possible heading angles at the target locations to find the most rewarding curvature-constrained path within the limited budget. Note that both the OP and the DOP are NP-hard similarly to the TSP and the DTSP (Le Ny et al. 2007).

In data collection planning, the solution quality, i.e., the path length in the (D)TSP or the sum of the rewards collected in the (D)OP, can be increased by introducing a non-zero sensing distance in which the data can be collected from the particular target locations. An extension of the DTSP for the non-zero sensing distance is called the Dubins Traveling Salesman Problem with Neighborhoods (DTSPN) (Obermeyer 2009; Isaacs et al. 2011; Váňa and Faigl 2015). In this paper, we consider a similar extension of the DOP to the Dubins Orienteering Problem with Neighborhoods (DOPN), initially introduced in (Pěnička et al. 2017b). Although exploiting the neighborhood in most cases increases the quality of the solutions (regarding the collected rewards), solving the DOPN is more challenging due to the additional determination of the most suitable waypoint locations to retrieve the rewards within the neighborhood of target locations. The DOPN thus includes both a combinatorial part and a continuous optimization part. Determining the subset of target locations and determining the sequence for visiting them are the combinatorial parts of the DOPN. The continuous optimization part involves determining the waypoint locations within the neighborhood of the target locations and the determining the waypoint heading angles of Dubins vehicle at the selected waypoint locations. An illustration of the DOPN solution from the experimental verification of the proposed method with a hexarotor UAV is shown in Fig. 1.

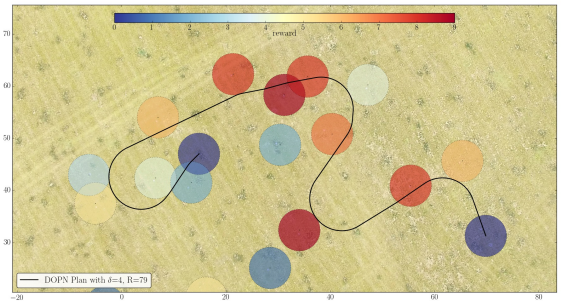


Fig. 1: A snapshot of the workspace for experimental verification of the proposed Dubins Orienteering Problem with Neighborhoods taken by a UAV flying 100 m above the ground. The solution of the DOPN used in the real experiment with a hexarotor UAV is calculated using the proposed Variable Neighborhood Search method with target neighborhood radius $\delta = 4\,\mathrm{m}$ and budget constraint $T_{max} = 150\,\mathrm{m}$.

The novel method for the DOPN is based on the Variable Neighborhood Search (VNS) metaheuristic (Mladenović and Hansen 1997). It consists of both combinatorial and continuous optimization operators to solve the DOPN. Initially, low-density equidistant sampling of both the waypoint heading angles and the waypoint locations within the neighborhoods is considered, in order to create waypoint graph for the combinatorial optimization to maximize the sum of the collected

rewards. The particular waypoint samples of the solutions for a given sequence of visited target locations are selected such that the path length is minimized. An initial greedy solution is then found by adding target locations that maximize the reward per tour prolongation, while the maximally allowed budget is still fulfilled. The VNS method afterward uses a set of neighborhood operators to randomly change and locally improve the best found solution. The proposed VNS consists of the combinatorial optimization operators extended from the VNS-based solution to the original OP in (Sevkli and Sevilgen 2006) and also utilized for solving the DOP in (Pěnička et al. 2017a). However, the herein VNS-based DOPN solver contains novel continuous optimization operators to minimize the path length over the selected sequence of target locations by optimizing both the heading angles and the waypoint locations within the neighborhoods of the target locations. The proposed operators shorten the solution found on the low-density sampled waypoint graph, and update the locally optimized values to the graph for combinatorial optimization. The path length is optimized to allow addition of previously unvisited target locations while satisfying the budget constraint.

A preliminary version of this work appears in (Pěnička et al. 2017b), where the DOPN is addressed by a purely sampling-based approach. This paper is considered to make the following contributions. The method introduced here significantly improves the solution quality and decreases the overall required computational time, which allows onboard online planning and anytime behavior. The proposed method combines combinatorial optimization and continuous optimization in a single VNS-based framework, which outperforms the previous purely combinatorial sampling-based solution (Pěnička et al. 2017b) and also the competitive Self-Organizing Map (SOM) based solution (Faigl and Pěnička 2017). The initial low-density waypoint sampling allows us to obtain high quality initial solutions ($\approx 90\%$ of the best-known rewards) within a few seconds, and due to the continuous optimization of the waypoints, the solution quality is improved above the so far best-known solutions created by dense waypoint sampling with required initialization in tens of minutes. The performance and quality improvements are mainly caused by the proposed tight coupling between combinatorial optimizations and continuous optimization in a single algorithm, which is also considered as one of the main contributions of our work. Furthermore, the designed VNS-based algorithm minimizes the path length in addition to the main OP objective of maximizing the sum of the collected rewards, which can be useful when all target locations can be feasibly collected within the defined budget. Last but not least, the experimental verification in the data collection scenario demonstrates the practical usefulness of the addressed problem and the proposed method.

The remainder of this paper is organized as follows. An overview of related work is presented in the next section. A formal definition of the DOPN is introduced in Section 3, and the novel VNS-based approach is proposed in Section 4. Section 5 shows the computational results and the experimental verification in a real data collection scenario. The conclusion and future work are outlined in Section 6.

2 Related Work

The Dubins Orienteering Problem with Neighborhoods belongs to a wider class of orienteering problems (Gunawan et al. 2016), where the objective is to find a limited length path between a starting location and an ending location which maximizes the sum of the rewards collected from a subset of the specified target locations. Therefore, this section presents an overview of existing approaches for the Orienteering Problem and relevant variants for UAVs. The DOPN is also related to the Traveling Salesman Problem (TSP) and its variants involving Dubins vehicle and neighborhoods; therefore a brief overview of relevant solutions of the TSP is provided in this section.

The Euclidean version of the OP, further denoted as the EOP, was introduced by Tsiligirides (Tsiligirides 1984) in 1984, together with the deterministic D-algorithm and stochastic S-algorithm approaches for the EOP. The S-algorithm is based on the Monte-Carlo method, which creates multiple feasible paths and selects the best solution according to the reward. The D-algorithm is based on the method for the vehicle routing problem (Wren and Holliday 1972). Furthermore, Tsiligirides created three OP benchmark instances (Vansteenwegen 2018), further denoted as Set 1, Set 2 and Set 3, with up to 33 target locations.

Since the first deterministic and stochastic algorithms for the OP, a large number of solutions for the EOP and other variants of the OP have been proposed (Vansteenwegen et al. 2011; Gunawan et al. 2016) with results that outperform the first solutions. The OP can be solved optimally using the Branch and Bound algorithm (Ramesh et al. 1992) or by the Branch and Cut (Fischetti et al. 1998) algorithm; however, the optimal solution of the EOP requires significant computational resources, and the solutions are provided in several minutes or hours for instances with tens of target locations.

For the Dubins Orienteering Problem or its variant with neighborhoods, additional waypoint sampling is required for each target location. Hence it is optimally solvable only for a given, rather low, sampling density with reasonable computational resources. Therefore, numerous heuristic solutions for the EOP, such as the approaches in (Ramesh and Brown 1991; Chao et al. 1996a; Schilde et al. 2009; Sevkli and Sevilgen 2006), have been proposed, with results that can achieve a solution close to the optimal one within a fraction of the computational time required for the optimal solution. The Fast and Effective heuristic for the EOP by Chao et al. (Chao et al. 1996a) considers only target locations reachable within the prescribed budget (i.e., target locations inside the respective ellipse around the prescribed starting and ending locations). This reduces the number of target locations in solutions with low budgets. The heuristic by Chao et al. uses a set of operators consisting of two-point exchange and one-point movement together with the 2-Opt operation to find high-quality EOP solutions. Furthermore, two symmetrical benchmark sets were created in (Chao et al. 1996a), the diamond shaped Set 64 and the square shaped Set 66 with up to 66 target locations.

The OP has also been proposed for path and data collection planning for UAVs. A variant of the OP, called the Correlated Orienteering Problem (COP) (Yu et al. 2016), introduced for persistent monitoring and data collection tasks with UAVs, proposed a variant of the OP where the rewards of target locations are correlated on the basis of their mutual distances. The COP is motivated by the correlation in sensory measurements of neighboring target locations, and its solution can be found optimally using mixed integer quadratic programming for a small number of target locations. A version of the COP involving Dubins vehicle has been proposed recently by Tsiogkas and Lane (2018).

Thakur et al. (2013) proposed a variant of the Team Orienteering Problem (TOP) (the multi-vehicle variant of the OP proposed by Chao et al. (1996b)) for Dubins vehicle in environments with obstacles. However, the definition of the problem proposed in (Thakur et al. 2013) consists of a given set of waypoints for Dubins vehicle and does not consider an arbitrary heading angle at the target locations or the non-zero sensing distance, as in the DOPN. An optimal multilevel graph search technique is proposed for optimizing the TOP on a given set of Dubins vehicle waypoints for up to 15 target locations. The multi-robot variant of the OP is also proposed in (Jorgensen et al. 2018) for so-called Team Surviving Orienteers (TSO), where the budget is replaced by the constraining probabilities that each robot survives to its destination.

The proposed DOPN method is based on the Variable Neighborhood Search (VNS) (Mladenović and Hansen 1997) metaheuristic by Hansen and Mladenović for combinatorial optimization applicable to numerous problems (Hansen and Mladenović 2001). The VNS employs predefined neighborhood operators used for iterative improvement of the initial solution inside the *shaking* and *local search* procedures. The first VNS-based approach to the EOP (Sevkli and Sevilgen 2006) uses neighborhood structures that motivate the combinatorial optimization part of the proposed solution of the DOPN. The VNS-based method for the EOP randomly changes the current best solution by either path move operator or path exchange operator in the *shaking* procedure to get from the possible local maximum. Then, the method tries to improve the randomly changed path by multiple one point moves or exchanges in the *local search* procedure in order to find a more rewarded path than the incumbent solution.

In our previous work (Pěnička et al. 2017a), the DOP was introduced together with the VNS-based method to solve it. The method uses similar neighborhood structures as the VNS method for the EOP (Sevkli and Sevilgen 2006). However, to tackle the continuous optimization problem of finding a suitable path for curvature-constrained Dubins vehicle, equidistant sampling of the heading angle at the target locations was proposed. The VNS-based method then searches for the most rewarding path, together with the appropriate sequence of sampled heading angles to fit the path length within the budget constraint. The DOPN and its heuristic VNS-based solution was introduced in (Pěnička et al. 2017b) with a straightforward extension of the pure sampling-based approach by additional sampling of visit positions in the circular neighborhood of each target location. In this paper, the solution of the DOPN is further improved by a combination of combinatorial optimization of the DOPN with continuous optimization of the waypoint samples in a single VNS-based algorithm. Furthermore, the deployment of the proposed method is shown in an experimental verification with a hexarotor UAV.

The first approach addressing the generalization of the OP to the Euclidean variant of the Orienteering Problem with Neighborhoods (OPN) was proposed in (Best et al. 2016), and was further improved in (Faigl et al. 2016). The multi-robot variant of the OPN for active perception has been studied in (Best et al. 2018). The approach is based on unsupervised learning of the Self-Organizing Map (SOM) for the Prize-Collecting Traveling Salesman Problem with Neighborhoods (PC-TSPN) (Faigl and Hollinger 2014), i.e., a variant of the TSP that combines maximization of the

rewards (prizes) and minimization of the path length. The approach has been further extended to variants with multiple vehicles in the OP (Faigl 2017) and also multi-vehicle PC-TSPN (Faigl and Hollinger 2018). The SOM has also been applied to the DTSP and DTSPN in (Faigl and Váňa 2017). Recently, the SOM-based approach has been adopted for solving the Close Enough Dubins Orienteering Problem (CEDOP) (Faigl and Pěnička 2017), which is the DOPN with name emphasized usage of disk-shaped neighborhoods. The VNS-based solution of the DOPN proposed here significantly outperforms the SOM-based approach for CEDOP, both in the maximal achievable solution quality and also regarding computational time.

The proposed DOPN is also related to existing approaches to the DTSP (Cohen et al. 2017) and the DTSPN (Váňa and Faigl 2015). The most relevant approaches are sampling-based variants of the DTSP, where the heading angles at the target locations are sampled, and the problem is transformed to the Asymmetric TSP (ATSP) (Noon and Bean 1993), which can be solved optimally for the specified sampling. A similar approach can be used for the DTSPN (Obermeyer et al. 2010), where both the heading angles and the positions within the neighborhood are sampled. The problem is then transformed into the Generalized TSP (GTSP) and further to the ATSP, which can be solved, e.g., by the LKH solver (Helsgaun 2000). The solutions of sampling based methods, however, can be further improved by employing the Dubins Touring Problem (DTP) (Faigl et al. 2017), which sets out to find the optimal heading angles of Dubins vehicle for a given sequence of target locations in order to minimize the path length in the DTSP. For the DTSPN, the DTP can be further extended to the Dubins Touring Regions Problem (DTRP), recently addressed as the Generalized Dubins Interval Problem (Váňa and Faigl 2018), where both the heading angles of Dubins vehicle and the visit position inside the neighborhoods of target locations are optimized for a given sequence of target locations. The proposed VNS-based solution of the DOPN uses the adopted version of the Local Iterative Optimization (LIO) procedure (Váňa and Faigl 2015) (originally designed for the DTRP) in continuous optimization VNS operators. It iteratively optimizes individual heading angles and neighborhood positions at each target location to minimize the required path length. The related DTSPN and its DTRP subproblem, however, does not contain subset selection with maximization of the collected rewards, and the budget constraint, as in the DOPN, which is formally introduced in the next section.

3 Problem Statement

In this section, we formally define the DOPN. The problem studied here consists of two main optimization parts. The first part is the combinatorial optimization part of the OP, which sets out to maximize the sum of the collected rewards by selecting a subset of the target locations such that the path length visiting them is within the specified travel budget. The second part is the continuous optimization of the DTRP which, for a given sequence of target locations themselves, sets out to find appropriate waypoint heading angles of Dubins vehicle and also the waypoint locations themselves in the neighborhoods of the selected target locations. Both parts have to be addressed at the same time, as the OP subset selection influences the continuous DTRP optimization, which on the other hand influences the path length constrained by the combinatorial OP. The addressed DOPN is therefore incrementally formulated from the OP and the DTRP in the following subsections.

3.1 Orienteering Problem (OP)

The OP assumes a given set of target locations to be visited $S = \{s_1, \cdots, s_n\}$, where each target location $s_i = (t_i, r_i)$ consists of its position in the plane $t_i \in \mathbb{R}^2$ and the associated reward r_i. The reward of all target locations is expected to be strictly positive $r_i \in \mathbb{R}_{>0}$, with the exception of the predefined starting location s_1 and ending location s_n with zero rewards $r_1 = r_n = 0$. Furthermore, the problem is constrained by the given maximal allowed travel budget T_{max}, i.e., the path length of the vehicle is limited by this value.

The objective of the OP is to maximize the sum of the collected rewards $R = \sum_{r_i \in S_k} r_i$ by selecting a subset of k target locations $S_k \subseteq S$. However, the length of the tour to visit all the locations of subset S_k is constrained by T_{max}, and therefore, the path length has to be taken into account during the selection of S_k. The path can be described as a sequence of target location indexes Σ_k, in which the path visits the selected target locations $\Sigma_k = (\sigma_1, \cdots, \sigma_k)$, with $1 \leq \sigma_i \leq n$, $\sigma_i \neq \sigma_j$ for $i \neq j$, $s_{\sigma_h} \in S_k$ where $h \in (1, \ldots, k)$ and $\sigma_1 = 1$, $\sigma_k = n$. Using the predefined starting and ending locations in the permutation ($\sigma_1 = 1$, $\sigma_k = n$), the solution of the OP is determined by searching over all possible values of k, S_k, and Σ_k. In the ordinary OP (Gunawan et al. 2016), the Euclidean distance $\mathcal{L}_e(s_{\sigma_i}, s_{\sigma_j})$ is used as the travel cost between two target locations s_{σ_i} and s_{σ_j}. Having these preliminaries, the OP can be formulated as the optimization problem:

Problem 1 (Orienteering Problem (OP))

$$\begin{aligned} &\text{maximize}_{k,S_k,\Sigma_k}\ R = \sum_{i=1}^{k} r_{\sigma_i} \\ &\text{subject to} \\ &\sum_{i=2}^{k} \mathcal{L}_e(t_{\sigma_{i-1}}, t_{\sigma_i}) \leq T_{max} \\ &\sigma_1 = 1, \sigma_k = n\ . \end{aligned} \tag{1}$$

3.2 Dubins Touring Regions Problem (DTRP)

In the DTRP, Dubins vehicle model is utilized to plan a curvature-constrained data collection path. Dubins (1957) showed that the shortest path between two configurations of Dubins vehicle can be found by a closed-form expression, and the path is one of six possible maneuvers of CSC or CCC type, where 'C' stands for turning right or left and 'S' means going straight. A configuration of Dubins vehicle $q = (p, \theta)^T = (x, y, \theta)^T$ can be described by its position $p = (x, y)$ in the plane, i.e., $p \in \mathbb{R}^2$, and the vehicle heading angle θ, $\theta \in \mathbb{S}^1$. The kinematic model of Dubins vehicle shown in (2) uses a constant forward velocity v_c and a control input u, which steers the vehicle. The minimal turning radius ρ of Dubins vehicle is assumed to be constant.

$$\dot{q} = \begin{bmatrix} \dot{p}^T \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = v_c \begin{bmatrix} \cos\theta \\ \sin\theta \\ \frac{u}{\rho} \end{bmatrix}, u \in [-1, 1] \tag{2}$$

In multi-goal path planning with curvature-constrained Dubins vehicle formulated as the DTSP or the DOP and their variants with neighborhoods, the important issue of the continuity of heading angles has to be solved. The analytical solution of optimal Dubins maneuvers (Dubins 1957) provides the shortest path between two target locations with known heading angles. However, the heading angles have to be appropriately found to connect multiple Dubins maneuvers into a path of minimal length over multiple target locations with a priori unknown heading angles. For a given sequence of waypoint locations, the problem of determining the optimal heading values is called the Dubins Touring Problem (DTP) (Faigl et al. 2017). For the purposes of the OP, we can consider a variant of the DTP in which the target locations in S_k are visited in the sequence defined by $\Sigma_k = (\sigma_1, \cdots, \sigma_k)$ with specified starting and ending locations $\sigma_1 = 1$ and $\sigma_k = n$, respectively. The problem is then to find a vector of the waypoint heading angles $\Theta_k = (\theta_{\sigma_1}, \cdots, \theta_{\sigma_k})$ that connects Dubins maneuvers at the target locations. The solution of the DTP minimizes the sum of the length of Dubins maneuvers, where $\mathcal{L}_d(q_{\sigma_i}, q_{\sigma_j})$ denotes the length of the shortest Dubins maneuver (Dubins 1957) between configurations q_{σ_i} and q_{σ_j}.

The DTRP additionally requires to find the waypoint locations within a disk-shaped neighborhood of each target location. The non-zero sensing distance in the DTRP is denoted as the neighborhood radius δ defining a δ-radius disk centered at the respective target location. The same neighborhood radius is used for all the target locations in the given sequence Σ_k, with the exception of the starting s_1 and ending s_k locations, which are assumed to have a zero neighborhood radius due to the vehicle taking off and landing at these locations. The DTRP extends the DTP to a variant where an additional vector of the waypoint locations $P_k = (p_{\sigma_1}, \cdots, p_{\sigma_k})$ has to be found. Each $p_{\sigma_i} \in \mathbb{R}^2$ defines the location within the δ neighborhood of the target location $s_{\sigma_i} = (t_{\sigma_i}, r_{\sigma_i}) \in S_k$ such that $\|p_{\sigma_i}, t_{\sigma_i}\| \leq \delta$ for $i \in (2, k-1)$ and $\|p_{\sigma_i}, t_{\sigma_i}\| = 0$ for $i = 1, k$. The DTRP sets out to minimize the length $\mathcal{L}(\Theta_k, P_k)$ of the Dubins tour over the given sequence of targets Σ_k by optimizing both the vector of the waypoint heading angles Θ_k and the vector of the waypoint locations P_k that are inside the neighborhoods of the particular selected targets. This type of continuous optimization problem is complex, as any change of, e.g., heading angle θ_{σ_i} at a single target location, influences not only the optimal location p_{σ_i} of the same waypoint, but also other adjacent waypoint heading angles and locations. The same applies to changes in the waypoint locations P_k. The DTRP can be summarized as the following optimization problem.

Problem 2 (Dubins Touring Regions Problem (DTRP))

$$\begin{aligned} &\text{minimize}_{\Theta_k,P_k}\ \mathcal{L}(\Theta_k, P_k) = \sum_{i=2}^{n} \mathcal{L}_d(q_{\sigma_{i-1}}, q_{\sigma_i}) \\ &\text{subject to} \\ &\quad q_{\sigma_i} = (p_{\sigma_i}, \theta_{\sigma_i}),\ p_{\sigma_i} \in P_k,\ \theta_{\sigma_i} \in \Theta_k,\ i \in (1, k)\ , \\ &\quad \|p_{\sigma_i}, t_{\sigma_i}\| \leq \delta\ ,\ i \in (2, k-1)\ , \\ &\quad \|p_{\sigma_1}, t_{\sigma_1}\| = 0\ ,\ \|p_{\sigma_k}, t_{\sigma_k}\| = 0\ , \\ &\quad \sigma_1 = 1\ ,\ \sigma_k = n\ . \end{aligned} \tag{3}$$

3.3 Dubins Orienteering Problem with Neighborhoods

The DOPN combines combinatorial OP reward maximization with continuous path length minimization of the DTRP. However, both optimization problems combined in the DOPN have to be addressed simultaneously, due to their mutual influence. The DOPN can

be therefore expressed in a single optimization formulation:

Problem 3 (Dubins Orienteering Problem with Neighborhoods (DOPN))

$$\text{maximize}_{k,S_k,P_k,\Sigma_k,\Theta_k}\ R = \sum_{i=1}^{k} r_{\sigma_i}$$

subject to

$$\begin{aligned} &\sum_{i=2}^{k} \mathcal{L}_d(q_{\sigma_{i-1}}, q_{\sigma_i}) \leq T_{max}\ , \\ &q_{\sigma_i} = (p_{\sigma_i}, \theta_{\sigma_i}),\ p_{\sigma_i} \in P_k,\ \theta_{\sigma_i} \in \Theta_k,\ i \in (1,k)\ , \\ &\|p_{\sigma_i}, t_{\sigma_i}\| \leq \delta\ ,\ i \in (2, k-1)\ , \\ &\|p_{\sigma_1}, t_{\sigma_1}\| = 0\ ,\ \|p_{\sigma_k}, t_{\sigma_k}\| = 0\ , \\ &\sigma_1 = 1\ ,\ \sigma_k = n\ . \end{aligned} \tag{4}$$

4 Proposed Approach for the DOPN

The proposed approach for the DOPN is a novel variant of the Variable Neighborhood Search (VNS) metaheuristic, which combines combinatorial optimization and continuous optimization. The preliminary VNS-based method for the DOPN proposed in (Pěnička et al. 2017b) is purely sampling-based combinatorial optimization, which requires significantly longer computational times and achieves lower quality solutions. In this paper, we propose the VNS-based method, which additionally contains continuous optimization to improve the solution quality and to reduce the computational burden. The method proposed here uses low-density initial sampling for solving the DTRP subproblem (sampling the waypoint heading angles and the waypoint locations); however, it also employs continuous optimization of the waypoints. This kind of waypoint optimization can shorten the actual tour to cover the same subset of target locations, and thus it potentially allows visits to additional as yet unvisited target locations, without violating the travel budget constraint. Optimized waypoints that shorten the actual path are therefore added to the initial sampled waypoints to be used further for OP optimization.

The proposed method for the DOPN is based on the VNS metaheuristic (Mladenović and Hansen 1997), which has been introduced for combinatorial optimization in various problems (Hansen and Mladenović 2001) and its principles are also applicable for continuous optimization (Mladenović et al. 2008). VNS uses *shake* and *local search* procedures to iteratively improve the best achieved incumbent solution. Both procedures use the l_{max} predefined operators in the context of the VNS described as neighborhood structures $N_l, l = 1, \ldots, l_{max}$, where, in each VNS iteration, the neighborhood N_l is gradually increased when no better solution is found. The *shake* procedure uses the incumbent solution and randomly changes it using one of its operators to get from possible local optima. The randomly changed incumbent solution is then used by the *local search* procedure in an attempt to increase the quality of the solution above the incumbent solution.

The VNS-based algorithm for the DOPN uses equidistant initial sampling of the waypoints in the δ-radius neighborhood disk centered at the respective target locations $s_{\sigma_i} \in S$. Each waypoint consists of the waypoint location p_{σ_i} on the circumference of the neighborhood circle and also the heading angle of Dubins vehicle θ_{σ_i} at the waypoint location. The initial sampling uses o equidistantly placed waypoint locations along the circumference of the δ-radius circle. Each such waypoint location is described throughout the VNS-based algorithm by its directional angle $\langle 0, 2\pi)$ from the respective target location. This allows the waypoint location to be described by only one parameter and, like the description by two parameters (x, y), does not restrict solutions of the DOPN. Zero neighborhood radius is used for both the starting locations and the ending location specified by the DOPN, as the exact start and end position of the vehicle is considered, and therefore the $o = 1$ location sample is used. The heading angle is similarly sampled into m values from the interval $\langle 0, 2\pi)$ for each of the o waypoint location samples. The sampling approach requires $(o \cdot m)$ samples per target location, which is sufficient for the initial solution of the DOPN (Pěnička et al. 2017b). However, a high sampling rate, which is needed for finding high-quality solutions, is very computationally demanding, and most of the waypoint samples are never used in the improvements to the solution. Therefore, we propose to use low-density sampling of the waypoints from the initialization, together with the online addition of the optimized waypoints, which shorten the current paths, to the set of initial waypoint samples. DOPN paths are then created on the optimized waypoint samples where the appropriate waypoints, i.e., vectors Θ_k and P_k, are selected for the target sequence Σ_k using the shortest path in the graph of samples between the starting and ending target locations.

The neighborhood operators used for combinatorial optimization inside the VNS algorithm for the DOPN, namely **Path Move**, **Path Exchange**, **One Point Move**, and **One Point Exchange**, were introduced for the Euclidean OP in (Sevkli and Sevilgen 2006). The modified version of the same operators was also used in the initial solution of the purely sampling-based DOPN in (Pěnička et al. 2017b). The novel VNS *shake* proce-

dure for the DOPN consists of the following $l = 1, \ldots, 3$ neighborhood operators: **Path Move** and **Path Exchange**, which are further described in detail in Section 4.1, and **Waypoint Shake**, which is described in Section 4.3. The particular l for the individual operators of the *shake* procedure are:

- **Waypoint Shake** ($l = 1$);
- **Path Move** ($l = 2$);
- **Path Exchange** ($l = 3$).

The *local search* procedure consists of three operators, **One Point Move** and **One Point Exchange**, which are discussed in Section 4.2, and **Waypoint Improvement**, which is described in Section 4.3. The particular l for the individual operators of the *local search* procedure are:

- **Waypoint Improvement** ($l = 1$);
- **One Point Move** ($l = 2$);
- **One Point Exchange** ($l = 3$).

The proposed continuous optimization of the waypoints is performed by a combination of the Waypoint Shake operator in the *shake* procedure and the Waypoint Improvement operator in the *local search* procedure. The operators randomly change the waypoint of the current solution of the DOPN, and then improve the waypoints by iterative usage of local improvements. The continuous optimization operators ($l = 1$) are prioritized in order to shorten any newly found solution (see Algorithm 1) and thus to allow the combinatorial operators ($l = 2, 3$) to add previously unvisited target locations within the same budget. Local optimization of the waypoint is also performed during the *local search* One Point Move and One Point Exchange operators, when a new unvisited target location is added to the path. The improvement ratio α_{imp} defines the minimal collected reward $R_{imp} = \alpha_{imp} R_{init}$ when the newly-added target location is optimized for its waypoint samples. Value R_{init} denotes the sum of the rewards collected by the initial greedy solution of the DOPN. This immediate shrinking of the path allows more unvisited target locations to be added within the same travel budget, and at the same time, improvement ratio α_{imp} ensures that only waypoints of promising paths are improved. Ratio α_{imp} thus represents a tradeoff between exploration and exploitation. While low α_{imp} attempts waypoint improvement for all new target location additions made by the local search, a high value of α_{imp} (up to the point where $\alpha_{imp} R_{init}$ is equal to the current maximal reward) tends to exploit (improve) only the best found solution. Having high α_{imp} can thus lead to an even better solution being missed by not continuously optimizing the waypoints of promising solutions. On the other hand, optimizing the waypoints of low-quality solutions is more computationally demanding, mainly due to the large number of additional waypoint samples that are never used in further solutions. The influence of ratio α_{imp} is shown in Section 5.1.

The internal representation of the DOPN solution in the designed VNS-based method consists of the vector $v = (s_{\sigma_2}, \ldots, s_{\sigma_{k-1}}, s_{\sigma_{k+1}}, \ldots, s_{\sigma_n})$, where the first $k-2$ elements are the selected target locations of set S_k, together with the starting $s_{\sigma_1} = s_1$ and ending $s_{\sigma_k} = s_n$ target locations ordered according to Σ_k. The rest of the vector elements are the unvisited target locations $(s_{\sigma_{k+1}}, \ldots, s_{\sigma_n})$. Any solution of the DOPN is describable only by v on existing waypoint samples, as the appropriate waypoints Θ_k and P_k at the target locations are selected from the samples waypoint graph in such a way that the path over Σ_k is minimal. During combinatorial optimization by the operators Path Move, Path Exchange, One Point Move, and One Point Exchange, the whole solution vector v is used, such that the same operators can change the order of the visited target locations, and new unvisited targets can also be introduced to the solution path.

The proposed VNS-based method for the DOPN is summarized in Algorithm 1. The method starts with the *getReachableLocations* procedure, which filters out all target locations unreachable within the budget to reduce the number of target locations considered to be visited by the travel budget T_{max}. The reachable set of target locations S_r then contains $s_i \in S_r$ such that $\mathcal{L}_e(s_1, s_i) + \mathcal{L}_e(s_i, s_1) - 2\delta \leq T_{max}$. Note that the Euclidean distance with subtracted neighborhood radius is used as the lower bound on the required distance to the target location. This can add some unreachable target locations for Dubins vehicle; however, it does not require to determine the waypoint location and the heading angle that visits the neighborhood of the target location.

Using the set of reachable target locations S_r, the initial solution is created by a method denoted as *createInitialPath*. This method greedily adds target locations into the initial path between the starting location and the ending location with respect to the additional reward per length increase of the data collection path. The initial solution then consists of all such added target locations that fit within the budget constraint T_{max}. Afterward, the VNS uses the neighborhood operators in the *shake* and *local search* procedures, which are described in detail in the following subsections, to improve the incumbent solution P, either by increasing the sum of the collected rewards or by shrinking the length of the equally rewarded solution. The termination condition for the proposed method can be the number of per-

Algorithm 1: VNS method for the DOPN

Input : S – Set of the target locations
Input : T_{max} – Maximal allowed travel budget
Input : o – Initial number of position waypoints for each target
Input : m – Initial number of heading values for each waypoints
Input : α_{imp} – Local waypoint improvement ratio
Input : l_{max} – Maximal neighborhood number
Output: P – Found data collecting path defined by k, S_k, Σ_k, Θ_k, and P_k

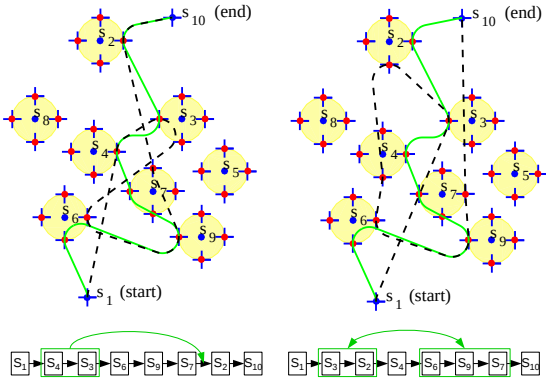
$S_r \leftarrow$ getReachableLocations(S, T_{max})
$P \leftarrow$ createInitialPath(S_r,T_{max}) // greedy
while *Stopping condition is not met* **do**
  $l \leftarrow 1$
  while $l \leq l_{max}$ **do**
    $P' \leftarrow$ shake(P, l)
    $P'' \leftarrow$ localSearch(P', l, α_{imp})
    if $\mathcal{L}_d(P'') \leq T_{max}$ ***and*** $[R(P'') > R(P)$ ***or*** $[R(P'') == R(P)$ ***and*** $\mathcal{L}_d(P'') < \mathcal{L}_d(P)]]$ **then**
      $P \leftarrow P''$
      $l \leftarrow 1$
    else
      $l \leftarrow l + 1$

formed iterations, or the number of iterations without any improvement, or the elapsed computational time, or a targeted sum of collected rewards. For brevity, the solution DOPN path (defined by k, S_k, Σ_k, Θ_k, and P_k) is denoted P, and the sum of the rewards collected by the vehicle traveling along path P is denoted $R(P)$ and its length is denoted as $\mathcal{L}_d(P)$.

4.1 Combinatorial *shake* Operators

The combinatorial part of the *shake* procedure consists of two operators, **Path Move** and **Path Exchange**. Both operators are intended to randomly change the currently best achieved incumbent solution to escape from possible local optima. Changes are made to the underlying sequence Σ_k and subset selection S_k by random reordering of the solution vector v, which internally represents the DOPN solution. Corresponding waypoints of the target locations for the reordered solution are selected from the existing graph of waypoint samples to minimize the overall length of the solution. A DOPN solution is selected from the first $k-2$ target locations in vector v that fit within the budget constraint between the starting location and the ending location.

Operator **Path Move** ($l = 2$), illustrated in Fig. 2a, randomly selects a part of the existing solution and moves it into a different randomly selected place within the solution vector. The operator is implemented by selecting three random indexes inside the solution vector, e.g., $i_1 \in \langle 2, n-1\rangle$, $i_2 \in \langle i_1+1, n-1\rangle$, $i_3 < i_1$ or $i_3 > i_2$, and $i_{1\ldots3} \neq k$. The DOPN solution vector $v = (s_{\sigma_2}, \ldots, s_{\sigma_n})$ of the initial incumbent solution is then changed, e.g., for the case of $i_3 > i_2$, into $v = (s_{\sigma_2}, \ldots, s_{\sigma_{i_1-1}}, s_{\sigma_{i_2+1}}, \ldots, s_{\sigma_{i_3}}, s_{\sigma_{i_1}}, \ldots, s_{\sigma_{i_2}}, s_{\sigma_{i_3+1}}, \ldots, s_{\sigma_n})$. Note that the operator can change not only the used part of the solution, the part until index $k-1$ that fits within T_{max} between the starting and ending locations, but it can also change the order of the unused target locations. The same property applies to all the other operators for combinatorial optimization.



(a) **Path Move** (b) **Path Exchange**

Fig. 2: **Path Move** and **Path Exchange** operators with a random change of the initial incumbent solution of the DOPN (dashed black) into a shorter solution (green) by changing the sequence of target locations and selecting the optimal waypoint samples. The combinatorial *shake* operators are shown with waypoint sampling, which consists of $o = 4$ number of neighborhood position samples and $m = 4$ heading samples of Dubins vehicle in each position sample.

The **Path Exchange** operator ($l = 3$) randomly selects two non-overlapping parts of the existing solution and switches their position inside the solution. Such a random exchange can be realized by selecting four feasible random indexes $i_1 \in \langle 2, n-1\rangle$, $i_2 \in \langle i_1+1, n-1\rangle$, $i_3 \in \langle i_2+1, n-1\rangle$, and $i_4 \in \langle i_3+1, n-1\rangle$ with $i_{1\ldots4} \neq k$. The initial solution $v = (s_{\sigma_2}, \ldots, s_{\sigma_n})$ is then modified by the operator into $v = (s_{\sigma_2}, \ldots, s_{\sigma_{i_1-1}}, s_{\sigma_{i_3}}, \ldots, s_{\sigma_{i_4}}, s_{\sigma_{i_2+1}}, \ldots, s_{\sigma_{i_3-1}}, s_{\sigma_{i_1}}, \ldots, s_{\sigma_{i_2}}, s_{\sigma_{i_4+1}}, \ldots, s_{\sigma_n})$. An example of the operator is shown in Fig. 2b.

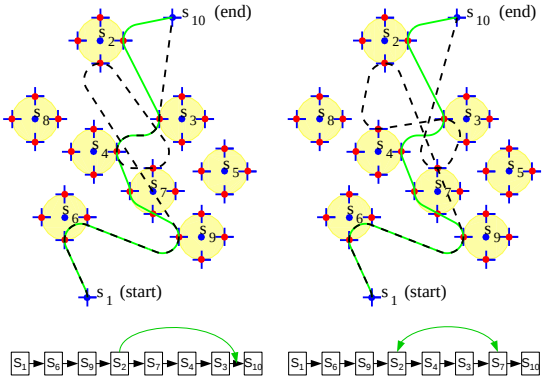
4.2 Combinatorial *local search* Operators

The *local search* operators for combinatorial optimization of the DOPN are **One Point Move** and **One Point Exchange**. The proposed method for the DOPN is based on the Randomized Variable Neighborhood

Search (RVNS), a variant of the VNS where the *local search* procedure is randomized. In the ordinary VNS, the *local search* uses a systematic local search in the solution space; however, in the proposed RVNS-based variant, the solution space is searched by numerous random operations. In both combinatorial local search operators, each random operator is tested for a number of times equal to the square of the number of reachable target locations. Every such random operation that increases the quality of the solution, i.e., increases the sum of the collected reward or decreases the path length for the same reward, is applied, and the operators continue with testing other random changes. Using *local search* randomized optimization, the solution created in the *shake* procedure is deeply searched for local optima in pursuit of a solution that improves the current best incumbent solution. Both the One Point Move operator and the One Point Exchange operator use the graph of the existing waypoint samples, and for any testing sequence Σ_k, represented by solution vector v, they select the waypoint samples that minimize the total path length.

For solutions with promising rewards (i.e., with the sum of the rewards equal to or higher than $R_{imp} = \alpha_{imp} R_{init}$, where R_{init} is the reward of the initial greedy solution created by *createInitialPath*), the operators also perform local optimization of the waypoint samples. When a new target location s_{σ_i} is added into the existing solution with a reward equal to or higher than R_{imp}, the currently selected waypoint heading sample θ_{σ_k} and the waypoint location sample inside the target neighborhood p_{σ_i} are optimized by a hill climbing method, similar to the method used in Waypoint Improvement, introduced in Section 4.3 for decreasing the length of the data collection path. Additionally, the waypoint samples of the adjacent target locations (in the current solution) are optimized, and if the path length after adding the new target location meets the budget constraint the solution is modified, and the optimized samples are inserted into the global graph of the waypoint samples. In this manner, the *local search* operators can fit more target locations within the same budget T_{max}, even with low initial sampling density determined by o and m.

The **One Point Move** operator ($l = 2$) shown in Fig. 3a randomly selects one target location within solution vector v and moves it into a different randomly chosen position inside v. By selecting two random indexes i_1 and i_2, $i_1 \neq i_2$, $i_1 \neq k$, $i_2 \neq k$, without loss of generality $i_1 < i_2$, inside $v = (s_{\sigma_2}, \ldots, s_{\sigma_n})$, the solution of a single move operation is $v = (s_{\sigma_2}, \ldots, s_{\sigma_{i_1-1}}, s_{\sigma_{i_1+1}}, \ldots, s_{\sigma_{i_2-1}}, s_{\sigma_{i_1}}, s_{\sigma_{i_2}}, \ldots, s_{\sigma_n})$. If such a move operation improves the quality of the solution,



(a) **One Point Move** (b) **One Point Exchange**

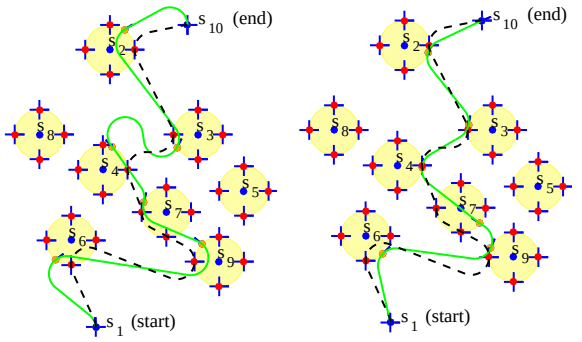
Fig. 3: *Local search* operators **One Point Move** and **One Point Exchange** with waypoint sampling $o = 4$ and $m = 4$. The combinatorial optimization operators randomly move one target location within the solution in the case of One Point Move or randomly exchange two target locations in the solution sequence by the One Point Exchange operator.

the change is applied, and further random One Point Move operations are tested.

The **One Point Exchange** operator ($l = 3$) illustrated in Fig. 3b is similar to the first $l = 2$ operator; however, instead of moving one target location within the solution v, the operator exchanges two randomly selected target locations. The operator can be realized by selecting two random indexes i_1 and i_2, $i_1 \neq i_2$, $i_1 \neq k$, $i_2 \neq k$, within the existing solution $v = (s_{\sigma_2}, \ldots, s_{\sigma_n})$ and by exchanging the target location with the selected indexes $v = (s_{\sigma_2}, \ldots, s_{\sigma_{i_1-1}}, s_{\sigma_{i_2}}, s_{\sigma_{i_1+1}}, \ldots, s_{\sigma_{i_2-1}}, s_{\sigma_{i_1}}, s_{\sigma_{i_2+1}}, \ldots, s_{\sigma_n})$. Like the One Point Move operator, the One Point Exchange operator tests numerous such operations and applies those that improve the quality of the solution.

4.3 Continuous Optimization Operators for the DOPN

This section presents two novel operators used for continuous optimization of the underlying DTRP to minimize the required path length for visiting a selected sequence of target locations. Minimization of the path length is motivated by the idea of fitting additional target locations that slightly violate the budget constraint T_{max}; however, the path length can fulfill T_{max} after optimizing the waypoint samples. Two proposed operators: **Waypoint Shake** inside the *shake* procedure, and **Waypoint Improvement** in the *local search*, are used within the proposed VNS algorithm as the first Neighborhood operators $l = 1$.



(a) **Waypoint Shake** (b) **Waypoint Improvement**

Fig. 4: **Waypoint Shake** and **Waypoint Improvement** with $o = 4$ samples of waypoint locations and $m = 4$ samples of heading angles at each waypoint location. Waypoint Shake randomly changes the current waypoint samples used in the incumbent solution. Waypoint Improvement optimizes the waypoint samples to minimize the length of the solution.

The **Waypoint Shake** operator randomly changes the waypoints currently used by the incumbent solution within the interval $\langle 0, 2\pi)$ for the heading angle and within the δ-radius circle for the waypoint location inside the target neighborhoods. Note that the waypoint location on the δ-radius circle can also be described by the angle within the interval $\langle 0, 2\pi)$, and it can therefore be changed and optimized in a similar way as the heading angle. The operation corresponds to a random change of vectors Θ_k and P_k that describe a solution of the DTRP, and similarly to the other *shake* operators it is intended to get the solution from possible local optima. Waypoint Shake is illustrated in Fig. 4a.

The **Waypoint Improvement** operator is a procedure which utilizes continuous local optimization of both the waypoint locations and the corresponding heading angles to improve the solution produced by Waypoint Shake. This continuous optimization enables solutions to be found closer to the optimum because the configurations are no longer selected from a discrete set of initial samples. This problem is formalized as the DTRP, as introduced in Section 3.2.

Although the sequence of visits to the targets is given, the DTRP remains challenging due to $2k + 2$ continuous variables, where k is the number of currently selected targets to be visited. One variable defines the location of the waypoint on the boundary of the respective target neighborhood, and the other variable defines the waypoint heading angle. The DTRP is addressed by dividing the problem into smaller optimization sub-problems, where each variable is treated separately. The modification of a single variable influences not more than two adjacent Dubins maneuvers, which makes the optimization very fast. However, the variables are mutually affected, and the local optimizations of a single variable are therefore repeated several times. This method has been adopted from the LIO procedure (Váňa and Faigl 2015), originally designed for the DTSPN.

5 Results

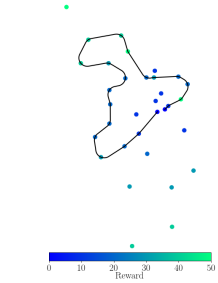
The proposed VNS-based solution to the DOPN has been evaluated on benchmark datasets for the regular OP from the literature, and has also been verified experimentally in a data collection scenario with a real UAV. The computational results on the OP datasets show that the proposed solution of the DOPN increases the so far best achieved collected rewards (Pěnička et al. 2017b) in numerous benchmark instances. The proposed approach also outperforms the only other existing solution of the DOPN, which is based on the Self-Organizing Map (SOM) (Faigl and Pěnička 2017). Moreover, in comparison to the preliminary purely sampling-based approach (Pěnička et al. 2017b), the computational time is significantly decreased by the continuous optimization and solutions with $\approx 90\%$ of the maximally collected rewards are found within several seconds. The experimental verification with the hexarotor UAV shows the benefit of using the neighborhood distance δ in a data collection scenario with a wide field of view camera.

5.1 Computational Results

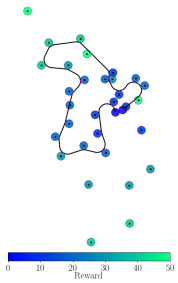
The VNS-based DOPN method has been evaluated on two existing benchmark[1] groups for the ordinary OP (Vansteenwegen 2018). The first group consists of **Set 3**, created by Tsiligirides (1984) with up to 34 randomly placed target locations and with various instances for different budget constraints T_{max}. The second group consists of two larger sets, **Set 64**, with a diamond-shaped structured placement, and **Set 66**, with a square-shaped structured placement, with up to 66 target locations proposed by Chao et al. (1996a). Example solutions of the DOPN for all benchmark sets used here are presented in Fig. 5, showing the benefits of using a non-zero neighborhood radius for maximizing the collected reward.

The computational times reported in this section have been achieved using a single core of the Intel i7

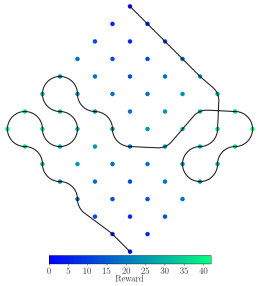
[1] Available online `https://www.mech.kuleuven.be/en/cib/op/#OP`



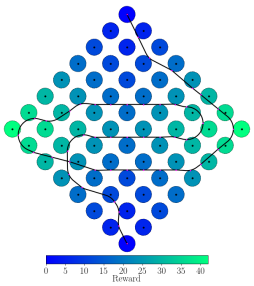
(a) Set 3, $\delta = 0$, $R = 510$



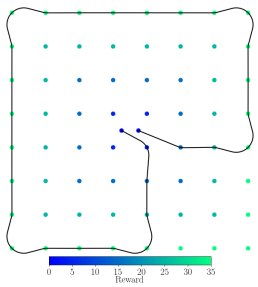
(b) Set 3, $\delta = 0.5$, $R = 570$



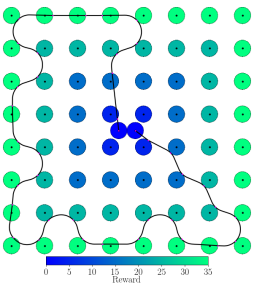
(c) Set 64, $\delta = 0$, $R = 894$



(d) Set 64, $\delta = 0.5$, $R = 1116$



(e) Set 66, $\delta = 0$, $R = 890$



(f) Set 66, $\delta = 0.5$, $R = 1055$

Fig. 5: Solutions of the DOPN for Set 3, Set 64, and Set 66, using $T_{max} = 50$, 55, and 60, respectively. All solutions are shown for the turning radius $\rho = 1.0$ and the neighborhood distance $\delta = 0$ (i.e., a solution of the DOP) on the left and $\delta = 0.5$ on the right. The indicated sum of the collected rewards R is shown to increase in all three instances, in comparison with the rewards of $\delta = 0$, with the VNS-base solution of the DOPN for $\delta = 0.5$.

3.4GHz CPU and C++ implementation of the proposed algorithm. The VNS is a stochastic method, and thus each benchmark instance has been solved 10 times to obtain meaningful statistical results. The initial waypoint sampling of the possible vehicle headings and the waypoint locations are $o = 8$ and $m = 8$. For instances with the zero neighborhood radius $\delta = 0$, only $o = 1$ has been used, and for the zero turning radius of Dubins vehicle $\rho = 0$, the algorithm automatically uses $m = 1$ samples for the heading angle at the target locations. The local waypoint improvement ratio α_{imp} used for defining the minimal collected reward in which the *local search* combinatorial operators start the local optimization of the waypoints during the target location addition has been set to $\alpha_{imp} = 0.95$. The termination criterion used in the computation of the presented results is a combination of the maximum of 10 000 iterations together with the maximal number of 5 000 iterations without any improvement.

Table 1: Maximal Collected Rewards for Set 3

T_{max}	δ=0			δ=0.5			δ=1		
	ρ=0	ρ=0.5	ρ=1	ρ=0	ρ=0.5	ρ=1	ρ=0	ρ=0.5	ρ=1
15	170	160	160	180	180	180	210	210	***200**
20	200	190	180	250	240	230	300	290	***290**
25	260	260	250	320	320	310	370	370	360
30	320	320	**320**	380	370	370	450	450	**450**
35	390	380	380	450	450	440	**510**	500	***490**
40	430	430	*420	500	500	480	570	570	***550**
45	470	460	***460**	550	550	***540**	600	600	***590**
50	*520	520	***510**	580	570	*570	630	630	***620**
55	550	550	530	620	620	600	670	670	***660**
60	580	580	560	650	650	630	710	710	***700**
65	610	600	590	680	670	***660**	750	740	***730**
70	640	630	***620**	720	710	***700**	790	780	***760**
75	670	660	***650**	750	740	**730**	800	800	***790**
80	**710**	690	**680**	790	780	***760**	800	800	800
85	740	730	***710**	800	800	*790	800	800	800
90	770	760	740	800	800	800	800	800	800
95	790	780	770	800	800	800	800	800	800
100	800	800	790	800	800	800	800	800	800
105	800	800	800	800	800	800	800	800	800
110	800	800	800	800	800	800	800	800	800

Table 2: Maximal Collected Rewards for Set 64

T_{max}	δ=0			δ=0.5			δ=1		
	ρ=0	ρ=0.5	ρ=1	ρ=0	ρ=0.5	ρ=1	ρ=0	ρ=0.5	ρ=1
15	96	96	96	204	204	**204**	***312**	312	***306**
20	294	294	252	432	426	***384**	576	570	*552
25	390	384	**342**	564	558	***510**	744	738	***714**
30	474	468	420	714	696	***630**	***954**	948	***936**
35	576	570	**516**	**894**	852	***774**	***1170**	1146	***1128**
40	714	696	624	1068	1026	***900**	***1296**	1272	***1242**
45	816	792	**708**	1164	1134	***990**	1344	1344	**1326**
50	900	882	*798	1248	1212	***1026**	1344	1344	*1344
55	984	972	894	1320	1296	***1116**	1344	1344	1344
60	1062	1044	**954**	1344	1344	**1188**	1344	1344	1344
65	1116	1098	**1020**	1344	1344	***1236**	1344	1344	1344
70	1188	1170	**1092**	1344	1344	***1290**	1344	1344	1344
75	1236	1218	**1134**	1344	1344	***1308**	1344	1344	1344
80	1284	1266	***1176**	1344	1344	***1344**	1344	1344	1344

One of the most important factors that shows the performance of the OP solver, or the DOPN solver in our case, is the maximally achievable sum of the collected rewards. The maximal rewards for various budget constraints T_{max} and for the neighborhood distance $\delta \in \{0, 0.5, 1.0\}$ and turning radii $\rho \in \{0, 0.5, 1.0\}$ are shown in the tables. In particular, the results for Set 3, Set 64, and Set 66 are presented in Table 1, Table 2, and Table 3, respectively. Due to the computational requirements for computing all the instances for various δ and ρ, the results for Tables 1–3 have been obtained with a grid of Xeon CPUs running at 2.2 GHz to 3.4 GHz. The solutions with an improved maximal collected reward with respect to the previously best found

Table 3: Maximal Collected Rewards for Set 66

T_{max}	δ=0			δ=0.5			δ=1		
	ρ=0	ρ=0.5	ρ=1	ρ=0	ρ=0.5	ρ=1	ρ=0	ρ=0.5	ρ=1
5	10	10	0	20	15	0	35	25	0
10	40	40	40	70	70	55	105	100	90
15	120	100	***100**	160	160	***160**	***225**	220	***205**
20	*205	200	195	265	265	***260**	***385**	370	***360**
25	280	280	275	**400**	380	***375**	540	540	540
30	400	380	370	***500**	495	***475**	685	685	*655
35	*465	465	455	605	595	***590**	870	845	835
40	**575**	570	***545**	**735**	710	***680**	**985**	965	***960**
45	650	650	*645	**840**	815	***775**	1135	1100	***1130**
50	730	710	**710**	920	920	***865**	1275	1240	*1235
55	**825**	825	820	***1050**	1015	***950**	1390	1370	***1380**
60	915	895	**890**	**1165**	1120	***1055**	**1555**	1500	***1485**
65	980	950	955	**1265**	1205	***1155**	1620	1605	***1590**
70	**1070**	1050	1070	**1360**	1315	***1260**	**1680**	1650	***1620**
75	1140	1090	**1120**	**1450**	1410	***1325**	1680	1680	***1680**
80	1215	1185	**1175**	1535	1490	***1375**	1680	1680	*1680
85	1270	1255	***1240**	1605	1570	***1410**	1680	1680	1680
90	1340	1310	1295	1635	1620	***1500**	1680	1680	1680
95	1395	1390	1365	1680	1665	***1565**	1680	1680	1680
100	**1465**	1445	1420	1680	1680	***1605**	1680	1680	1680
105	1520	1505	***1480**	1680	1680	***1670**	1680	1680	1680
110	1550	1550	**1535**	1680	1680	***1680**	1680	1680	1680
115	1595	1580	1565	1680	1680	***1680**	1680	1680	1680
120	1625	1625	**1610**	1680	1680	1680	1680	1680	1680
125	1670	1655	1640	1680	1680	1680	1680	1680	1680
130	1680	1680	1670	1680	1680	1680	1680	1680	1680

solutions provided by the purely sampling VNS-based solution of the DOPN (Pěnička et al. 2017a) without continuous optimization are displayed in bold font. In all cases, the maximal collected rewards are the same as, or better than, the previously best found solutions.

The instances with significantly better rewards using significance level $\alpha = 0.05$ are denoted by '*' based on a t-test comparison between the proposed method and the purely sampling VNS-based solution. As is shown in Tables 1–3, the maximal collected reward is improved mainly for non-zero neighborhood distances δ and turning radii ρ. This is caused by the fact that the improvements are mainly due to the continuous optimization, which is only effective when at least $\delta > 0$ or $\rho > 0$. Furthermore, the longer δ and ρ are, the greater their influence on the path length (continuously optimized in the proposed approach), and thus on the maximally achievable reward. Increasing the turning radius enlarges the limited path length and thus decreases the collected reward. Larger neighborhood radius, on the other hand, can increase the collected reward due to the distance savings. Both effects can be observed in Tables 1–3 by comparing $\rho = 0$ and $\rho = 1$ for the turning radius, and $\delta = 0$ and $\delta = 1$ for the neighborhood radius. The results for $\rho = 0.5$ do not contain any improvement, as this particular turning radius has no previously known best found solutions (Pěnička et al. 2017a). Furthermore, the maximal reward cannot be improved for a large number of instances where the created path visits all possible target locations. This is mainly noticeable for $\delta > 0$ and higher budgets T_{max}, where from a certain budget the maximally collected reward does not increase. Significantly better results (based on the t-test comparison) are in most cases in the same instances where the maximal collected reward is improved. However, for several maximal reward improvements, the newly found maximal reward tends to be an outlier, and the reward is not systematically higher. On the other hand, several instances have significantly better rewards without improving the maximal score. This is caused by the closeness of the average reward of the proposed method to the known maximum, together with a small standard deviation in comparison with the purely sampling VNS-based solution. We refer to an enlarged variants[2] of Tables 1–3, which contain the average rewards and the standard deviations of all instances.

The continuous optimization of the waypoint samples is one of the main improvements of the proposed DOPN algorithm in comparison to the previous algorithm proposed in (Pěnička et al. 2017b). Waypoint optimization is used in two main parts of the proposed VNS-based algorithm. The first part is used in the combinatorial *local search* operators One Point Move and One Point Exchange while testing an additional target insertion into a solution with the minimal reward of $R_{imp} = \alpha_{imp} R_{init}$. The waypoint samples of the inserted and adjacent target locations are locally optimized to shrink the length of the solution below T_{max}. The second waypoint optimization is through the Waypoint Shake and Waypoint Improvement operators, which randomly change the waypoint samples and optimize the randomly changed waypoints to minimize the length of the incumbent solution. Fig. 6 shows a comparison of the solution quality as the average sum and as the maximal sum of the collected rewards over the computational time. The solution obtained for 'High sampling DOPN' uses $o = 12$ and $m = 12$ samples and the 'Low waypoint sampling' solution uses $o = 4$ and $m = 4$ samples. Both use only combinatorial optimization with sampled waypoints, as proposed in (Pěnička et al. 2017b). The reward improvement for the solutions denoted as 'Local optimization *local search*' uses local waypoint improvement during the combinatorial *local search* operators, together with low initial waypoint sampling. The solution denoted as 'With Waypoint Improvement' shows the reward improvement with waypoint optimization employed both in local waypoint improvement during combinatorial *local search* and by using the Waypoint Shake and Waypoint Improvement operators. Low waypoint sampling $o = 4$ and $m = 4$ is also used for the solution with both waypoint optimizations.

[2] `https://archive.org/download/vns-dopn/results.pdf`

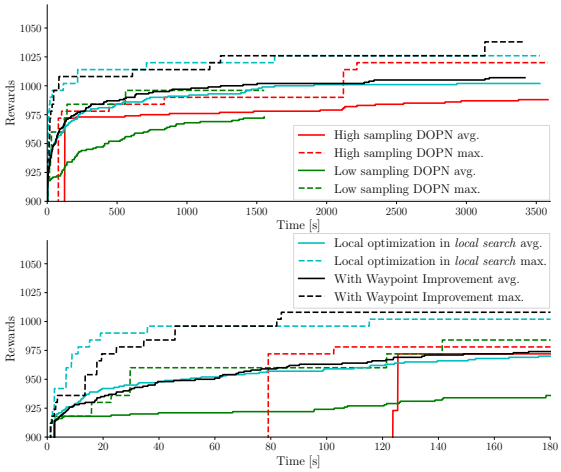


Fig. 6: Comparison of the average sum and the maximal sum of the collected rewards over time for high $o = m = 12$ and low $o = m = 4$ dense initial sampling of the DOPN without and with waypoint optimization. The solution denoted as 'Local optimization in *local search*' uses waypoint optimization only in the combinatorial *local search* operators and the 'With Waypoint Improvement' solution uses additionally the Waypoint Shake and Waypoint Improvement operators. The algorithm performance is shown for Set 64 with $T_{max} = 50$, $\rho = 1.0$ and $\delta = 0.5$. The upper plot shows the reward progress over one hour of maximal computational time, while the lower plot shows a detail of the initial 180 seconds of computation.

The comparison shows that waypoint continuous optimization increases the maximal achieved sum of the collected rewards within the maximal one hour of computational time, which was used in these tests as an additional stopping criterion. The maximum sum of the achieved reward is (similarly to the results in Table 1–3) considered as one of the most important aspects of the OP solver that shows the limiting extreme-most performance of the proposed method. The initial solution for low-density sampling together with waypoint optimization (denoted as 'With Waypoint Improvement') provides solutions with more than 90% of the best solutions within a few seconds only. The initial solution for a high sampling solution, however, takes approximately two minutes to compute, and in that time its maximally achieved reward is outperformed by the initially low sampling solution with both waypoint optimizations. The high sampling approach has a bigger average reward than the proposed method shortly after initialization (between time 125 s and 149 s) due to the quality of the initial solution, which influences the average. However, the high sampling approach is limited by static samples, which also limit the maximally achievable reward during the computation and also result in a lower average reward than that of the proposed method after 167 s of calculation.

The local optimization in *local search* is highly influenced by the selection of the improvement ratio α_{imp}, which determines how rewarded (compared to the reward of initial solution) a solution has to be in order to perform the local waypoint optimization of newly added target locations. Fig. 7 shows a comparison of the selected values of α_{imp} with the average and maximal sum of the collected rewards over the computational time.

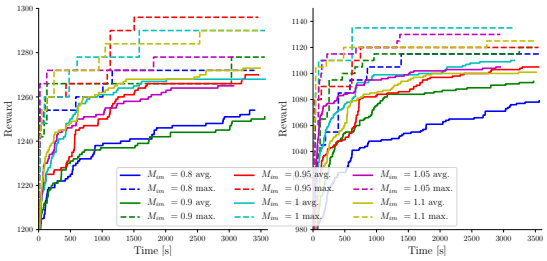


Fig. 7: Comparison of the average and maximal sum of the collected rewards for improvement ratio $\alpha_{imp} \in \{0.8, 0.9, 0.95, 1.0, 1.05, 1.1\}$ over the computational time for Set 64, on the left, with $T_{max} = 55$, $\rho = 0.5$, $\delta = 0.5$ and for Set 66, on the right, with $T_{max} = 60$, $\rho = 0.5$, $\delta = 0.5$.

The comparison shows that higher $\alpha_{imp} \in \{1.05, 1.1\}$ tends to optimize only the current incumbent solution which is demonstrated by the faster initial growth of the average rewards. In the same time, however, the higher α_{imp} prohibits continuously optimizing enough promising solutions in order to find even better than the currently best found solution, which is reflected in the fact that it cannot find the best known solutions. On the other hand, the lower $\alpha_{imp} \in \{0.8, 0.9\}$ slows down the computation of local search operators (by performing waypoint optimization on more non-promising solutions), which results in both lower average rewards and lower maximal rewards. Maximal rewards are achieved by either $\alpha_{imp} \in \{0.95, 1.0\}$ based on the dataset instance being solved.

To the best of our knowledge, the only other existing algorithm for solving the DOPN is the SOM-based approach for the so-called Close Enough Orienteering Problem, which is in fact the DOPN emphasizing the usage of circular neighborhoods. A comparison of the proposed VNS-based and SOM-based algorithms for various neighborhood radii is presented in Fig. 8 for 20 runs of VNS per instance, and for 80 runs in the case of the SOM-based algorithm. The results show that for all tested neighborhood distances, the proposed method produces solutions with similar or significantly

better results than the SOM-based algorithm (Faigl and Pěnička 2017).

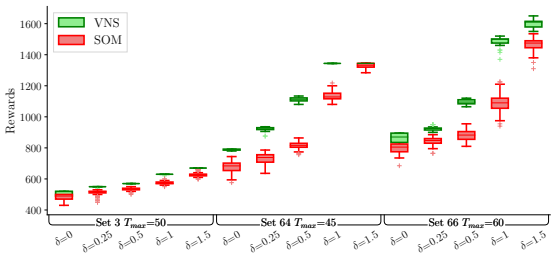


Fig. 8: Comparison of the proposed VNS-based solution and the SOM-based solution (Faigl and Pěnička 2017) of DOPN for selected neighborhood distances. The collected rewards are shown for Dubins vehicle with turning radius $\rho = 0.5$ for all Set 3, Set 64, and Set 66.

The performance of the proposed algorithm is significantly influenced by the initial waypoint sampling density defined by the number of Dubins vehicle heading samples m at each of the o waypoint location samples. Waypoint sampling mainly influences the computational time and the maximally achievable sum of the collected rewards. Using high sampling density such as $o = 12$ and $m = 12$ requires much more computational time in the local search One Point Move and One Point Exchange operators for selecting the samples with the shortest path for a given sequence of target locations. However, with high sampling density, the quality of the solution as the sum of the collected rewards is higher than for low-density sampling. For the newly proposed VNS-based solution of the DOPN with optimization of the waypoint samples, it is possible to achieve the same solution quality with low initial sampling through optimization of the samples. The initial sampling therefore mainly influences the evolution of the solution quality, i.e., the maximal sum and the average sum of the collected rewards, over the computational time, as shown in Fig. 9.

The comparison of the initial waypoint sampling shows that both for very low sampling $o = m = 1$ and for very high sampling $o = m = 16$, the average and maximal collected rewards are below other medium sampling densities. The highest maximal and average rewards are collected with waypoint sampling $o = m = 8$, with similar results for $o = m = 4$. However, the computational time required for creating the initial solution with $o = m = 8$ is approximately 17 s, while for a lower sampling density $o = m = 4$, it is within 1.5 s.

Fig. 9 also shows the computational requirements of the proposed VNS-based solution for the DOPN, and can be used for a comparison of the computational time

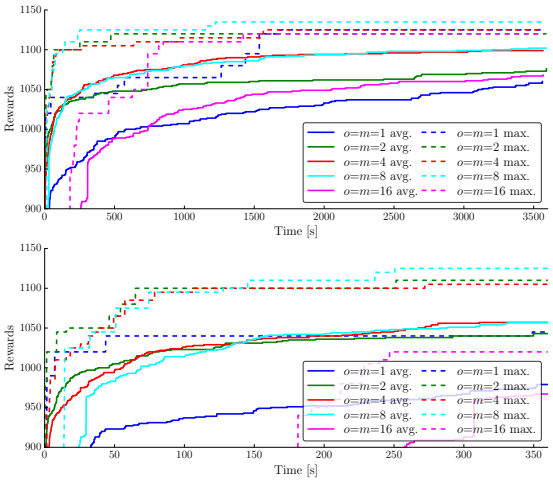


Fig. 9: Comparison of different initial waypoint sampling densities for Set 66 with $T_{max} = 60$, $\rho = 0.5$, and $\delta = 0.5$. In the upper plot, the average and maximal collected rewards are shown over one hour of the maximal computational time. The lower plot shows a detail of the average and maximal rewards within the initial 360 seconds of computation.

for achieving a certain sum of collected rewards. The SOM-based solution (Faigl and Pěnička 2017) requires, for Set 66, $T_{max} = 60$, $\delta = 0.5$ and $\rho = 0.5$, an average computational time of 33.8 s with the maximal achieved rewards $R = 955$ and average $\overline{R} = 880$. For the same configuration of the problem, Fig. 9 shows that in 33.8 s the average sum of the collected rewards is 970 for $o = m = 8$, 985 for $o = m = 4$, and 997 for $o = m = 2$. The maximal collected rewards at the same time are 1045 for $o = m = 8$, 1040 for $o = m = 4$, and 1050 for $o = m = 2$. The proposed VNS-based solution outperforms the state-of-the-art algorithm not only in the maximally achievable sum of collected rewards but also regarding the computational time required for achieving a certain sum of collected rewards. This is very important for a real deployment of the method, as it is demonstrated in the following section.

5.2 Experimental Verification

The proposed method was experimentally verified in a visual data collection scenario with a real hexarotor UAV in an outdoor environment. Although the Vertical Take-Off and Landing (VTOL) UAV does not necessarily have to be modeled as Dubins vehicle, the model is convenient for the VTOL UAV when traversing a curvature-constrained path at a constant speed. Constant speed flights of the VTOL can be beneficial for visual data collection missions, where additional vi-

sual information during flights between target location neighborhoods can be further used, and constant speed improves the quality of the images that are taken. From the selected constant speed v_c and the maximal acceleration a_{max} of the UAV, the minimal turning radius of Dubins vehicle can be computed using the equation of circular motion with constant speed $\rho = v_c^2/a_{max}$. Note that for the VTOL (not so much for the fixed-wing UAV), the solution quality can be improved by allowing acceleration to the maximal speed during straight line segments of Dubins maneuvers. This, however, is only a technical consideration, which does not require any change to the proposed algorithm. Considering these accelerated Dubins maneuvers would only require the time of flight cost instead of length cost, and would change the length-budget constraint to a time-budget constraint. We therefore consider for experimental verification the DOPN as defined in Problem 3 with standard Dubins maneuvers for VTOL, which also provides a better comparison of the results. We refer to Pěnička et al. (2017a) for a comparison of an experimental deployment of the DOP and the ordinary OP with a straight line trajectory and sharp turns. The advantage of using Dubins vehicle model for fast and reliable visits to multiple locations to be scanned by the onboard camera was also demonstrated by our team in the third challenge of the Mohamed Bin Zayed International Robotics Challenge (MBZIRC) competition, which also motivated the research presented here. The effectiveness of fast flying using Dubins vehicle model (e.g., demonstrated in the MBZIRC challenge in the solution of the DTSP) resulted in the best performance among all 143 competing teams.

The used hexarotor UAV was designed for the MBZIRC competition[3] and was built on the DJI hexacopter F550 frame with E310 DJI motors and with the PixHawk Autopilot low-level flight controller (Meier et al. 2012). The low-level localization of the PixHawk Autopilot is realized as a combination of the standard GPS with a compass and with the accelerometers and gyroscopes at the lowest level. To increase the localization precision, the system uses PRECIS-BX305 (Tersus-GNSS 2018) RTK GPS with centimeter accuracy and also the *TeraRanger One* laser rangefinder for measuring the distance from the ground. The onboard Intel NUC-i7 mini PC provides enough resources for calculating the plan for the addressed data collection scenario formulated as the DOPN. In addition, the onboard computer realizes the Model Predictive Control (MPC) trajectory controller (Báča et al. 2016) for trajectory tracking, UAV localization estimation, and sensor fusion. For a visual information gathering task, a high-resolution wide field of view Mobius ActionCam camera was used. The hardware components are summarized in Fig. 10.

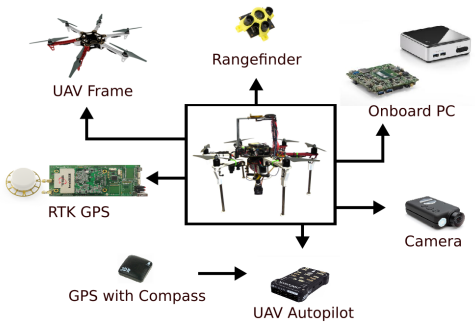


Fig. 10: Hardware components of the hexarotor UAV for experimental verification of the data collection scenario formulated as the Dubins Orienteering Problem with Neighborhoods.

The results from the experimental verification of the proposed DOPN method in a realistic data collection scenario using the onboard camera are shown in Fig. 11. During the experiment, constant vehicle speed $v_c = 4\,\mathrm{ms}^{-1}$ was used together with maximal acceleration $a_{max} = 2.6\,\mathrm{ms}^{-2}$, which resulted in $\rho = 6.15\,\mathrm{m}$ turning radius of Dubins vehicle. The scenario for the experimental verification consists of 19 target locations, including the starting and ending locations, with the budget constraint set to $T_{max} = 150\,\mathrm{m}$. The planned trajectories for the various neighborhood radii $\delta = \{0, 3, 6\}$ m together with the real flown trajectories of the UAV are depicted in Fig. 11.

The real trajectories deviate slightly from the planned trajectory due to the strong wind conditions and the tightly set maximal acceleration of the vehicle. However, the presented camera images show the target markers placed throughout the experimental area, which were taken at the respective waypoints of the targets, and thus the mission was successfully fulfilled.

The sum of the collected rewards for the increasing neighborhood radii shows the main benefits of using the VNS-based solver for the DOPN with non-zero turning radii, where the collected reward increases with each incrementation of the neighborhood distance. The onboard camera images in Fig. 11 show that the high neighborhood radius of 6 m, with a high collected reward, is usable for a visual data collection scenario of this kind. The complete set of radii used during the experimental verification, together with the corresponding sum of the collected rewards and the path lengths, is shown in Table 4.

[3] See `http://mrs.felk.cvut.cz/mbzirc` for examples of the experimental deployment of the system.

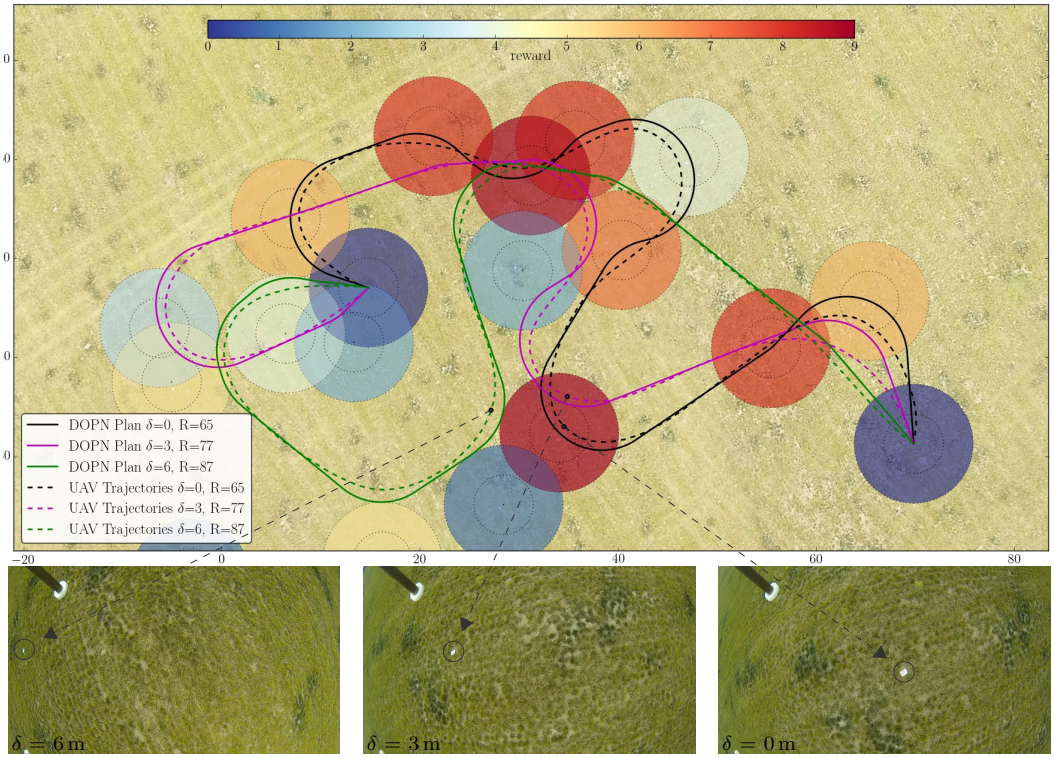


Fig. 11: Snapshots from the experimental verification of the proposed VNS-based algorithm for the DOPN in a data collection scenario with various neighborhood distances δ. We refer to `https://youtu.be/zPXZahW33-w` for supporting video material from the experimental verification of the method.

Table 4: Collected reward and path length from the real experiment for various neighborhood radius δ

δ [m]	0	1	2	3	4	5	6	7
Reward R	65	67	71	77	79	85	87	88
Length [m]	148.0	143.3	146.9	148.0	148.6	147.9	139.0	143.7

6 CONCLUSIONS

In this paper, we introduce a novel approach for curvature-constrained data collection planning with UAVs that is formulated as the Dubins Orienteering Problem with Neighborhoods (DOPN). The DOPN sets out to find a path for Dubins vehicle that maximizes the sum of the collected rewards by visiting a subset of the given target locations with prescribed starting and ending locations and a constrained travel budget. The DOPN uses a predefined circular neighborhood at each target location, motivated by remote data collection from the target locations to save the required travel cost, and thus to increase the sum of the collected rewards within the same budget constraint.

The proposed Variable Neighborhood Search-based method uses a set of neighborhood operators that perform a combinatorial optimization to maximize the sum of the collected rewards by selecting a subset of target locations to be visited, and also by determining the sequence of the visits. The proposed method employs initial low-density waypoint sampling consisting of sampling both Dubins vehicle headings and waypoint locations within the neighborhood of each target location, to quickly determine an initial data collection path by a greedy maximization of reward per tour prolongation. The continuous optimization employed in the proposed VNS neighborhood operators is used for the optimizing the initial waypoint samples to minimize the length of the Dubins path visiting the neighborhoods of the selected target locations. The proposed waypoint optimization increases the sum of the collected rewards by adding unvisited target locations within the prescribed budget constraint.

The computational results show that the proposed VNS-based algorithm is a viable method for solving the DOPN. The continuous optimization employed in the novel approach significantly improves the required computational time, and also improves the best-known solutions in several benchmark instances. The method also outperforms the only other existing SOM-based DOPN approach in both solution quality and computational time. Finally, the experimental verification of the pro-

posed method with a real hexarotor UAV demonstrates the deployment of the proposed solution of the DOPN in a data collection scenario with an onboard camera. The solutions found for the experimental deployment also show the benefits of using a non-zero neighborhood distance on the sum of the collected rewards.

For our future work, we intend to extend the proposed approach to a variant of multi-UAV data collection scenarios and to employ more complex maneuvers, such as cubic splines, which are suitable for the non-constant forward velocity of the VTOL UAV, e.g., as in Faigl and Váňa (2018) for solving TSP-like scenarios.

References

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

Best G, Faigl J, Fitch R (2016) Multi-robot path planning for budgeted active perception with self-organising maps. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp 3164–3171

Best G, Faigl J, Fitch R (2018) Online planning for multi-robot active perception with self-organising maps. Autonomous Robots 42(4):715–738

Chao IM, Golden BL, Wasil EA (1996a) A fast and effective heuristic for the orienteering problem. European Journal of Operational Research 88(3):475–489

Chao IM, Golden BL, Wasil EA (1996b) The team orienteering problem. European Journal of Operational Research 88(3):464–474

Cohen I, Epstein C, Isaiah P, Kuzi S, Shima T (2017) Discretization-based and look-ahead algorithms for the dubins traveling salesperson problem. IEEE Transactions on Automation Science and Engineering 14(1):383–390

Dubins LE (1957) On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. American Journal of Mathematics 79(3):497–516

Ergezer H, Leblebicioğlu K (2014) 3d path planning for multiple uavs for maximum information collection. Journal of Intelligent & Robotic Systems 73(1):737–762

Faigl J (2017) On self-organizing maps for orienteering problems. In: 2017 International Joint Conference on Neural Networks (IJCNN), pp 2611–2620

Faigl J, Hollinger GA (2014) Unifying multi-goal path planning for autonomous data collection. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp 2937–2942

Faigl J, Hollinger GA (2018) Autonomous data collection using a self-organizing map. IEEE Transactions on Neural Networks and Learning Systems 29(5):1703–1715

Faigl J, Pěnička R (2017) On close enough orienteering problem with dubins vehicle. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp 5646–5652

Faigl J, Váňa P (2017) Unsupervised learning for surveillance planning with team of aerial vehicles. In: International Joint Conference on Neural Networks (IJCNN), pp 4340–4347

Faigl J, Váňa P (2018) Surveillance planning with bézier curves. IEEE Robotics and Automation Letters 3(2):750–757

Faigl J, Pěnička R, Best G (2016) Self-organizing map-based solution for the orienteering problem with neighborhoods. In: IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp 1315–1321

Faigl J, Váňa P, Saska M, Báča T, Spurný V (2017) On solution of the dubins touring problem. In: European Conference on Mobile Robotics (ECMR), pp 151–156

Fischetti M, González JJS, Toth P (1998) Solving the orienteering problem through branch-and-cut. INFORMS Journal on Computing 10(2):133–148

Gunawan A, Lau HC, Vansteenwegen P (2016) Orienteering problem: A survey of recent variants, solution approaches and applications. European Journal of Operational Research 255(2):315–332

Hansen P, Mladenović N (2001) Variable neighborhood search: Principles and applications. European Journal of Operational Research 130(3):449–467

Helsgaun K (2000) An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic. European Journal of Operational Research 126(1):106–130

Isaacs JT, Klein DJ, Hespanha JP (2011) Algorithms for the traveling salesman problem with neighborhoods involving a dubins vehicle. In: American Control Conference (ACC), pp 1704–1709

Jawhar I, Mohamed N, Al-Jaroodi J, Zhang S (2014) A framework for using unmanned aerial vehicles for data collection in linear wireless sensor networks. Journal of Intelligent & Robotic Systems 74(1):437–453

Jorgensen S, Chen RH, Milam MB, Pavone M (2018) The team surviving orienteers problem: routing teams of robots in uncertain environments with survival constraints. Autonomous Robots 42(4):927–952

Le Ny J, Frazzoli E, Feron E (2007) The curvature-constrained traveling salesman problem for high point densities. In: IEEE Conference on Decision and

Control, pp 5985–5990

Lugo-Cárdenas I, Flores G, Salazar S, Lozano R (2014) Dubins path generation for a fixed wing uav. In: International Conference on Unmanned Aircraft Systems (ICUAS), pp 339–346

Meier L, Tanskanen P, Heng L, Lee GH, Fraundorfer F, Pollefeys M (2012) Pixhawk: A micro aerial vehicle design for autonomous flight using onboard computer vision. Autonomous Robots 33(1):21–39

Mladenović N, Hansen P (1997) Variable neighborhood search. Computers & Operations Research 24(11):1097–1100

Mladenović N, Dražić M, Kovačevic-Vujčić V, Čangalović M (2008) General variable neighborhood search for the continuous optimization. European Journal of Operational Research 191(3):753 – 770

Nguyen JL, Lawrance NRJ, Fitch R, Sukkarieh S (2016) Real-time path planning for long-term information gathering with an aerial glider. Autonomous Robots 40(6):1017–1039

Noon CE, Bean JC (1993) An efficient transformation of the generalized traveling salesman problem. INFOR: Information Systems and Operational Research 31(1):39–44

Oberlin P, Rathinam S, Darbha S (2010) Today's traveling salesman problem. IEEE Robotics Automation Magazine 17(4):70–77

Obermeyer KJ (2009) Path planning for a uav performing reconnaissance of static ground targets in terrain. In: AIAA Guidance, Navigation, and Control Conference

Obermeyer KJ, Oberlin P, Darbha S (2010) Sampling-based roadmap methods for a visual reconnaissance uav. In: AIAA Guidance, Navigation, and Control Conference

Pěnička R, Faigl J, Váňa P, Saska M (2017a) Dubins orienteering problem. IEEE Robotics and Automation Letters 2(2):1210–1217

Pěnička R, Faigl J, Váňa P, Saska M (2017b) Dubins orienteering problem with neighborhoods. In: International Conference on Unmanned Aircraft Systems (ICUAS), pp 1555–1562

Ramesh R, Brown KM (1991) An efficient four-phase heuristic for the generalized orienteering problem. Computers & Operations Research 18(2):151–165

Ramesh R, Yoon YS, Karwan MH (1992) An optimal algorithm for the orienteering tour problem. ORSA Journal on Computing 4(2):155–165

Savla K, Frazzoli E, Bullo F (2005) On the point-to-point and traveling salesperson problems for dubins' vehicle. In: American Control Conference (ACC), vol 2, pp 786–791

Schilde M, Doerner KF, Hartl RF, Kiechle G (2009) Metaheuristics for the bi-objective orienteering problem. Swarm Intelligence 3(3):179–201

Sevkli Z, Sevilgen FE (2006) Variable neighborhood search for the orienteering problem. In: International Symposium on Computer and Information Sciences (ISCIS), pp 134–143

Tersus-GNSS (2018) PRECIS-BX305 GNSS RTK Board. URL `https://www.tersus-gnss.com`, (cited on 2018-07-21)

Thakur D, Likhachev M, Keller J, Kumar V, Dobrokhodov V, Jones K, Wurz J, Kaminer I (2013) Planning for opportunistic surveillance with multiple robots. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp 5750–5757

Tokekar P, Karnad N, Isler V (2014) Energy-optimal trajectory planning for car-like robots. Autonomous Robots 37(3):279–300

Tsiligirides T (1984) Heuristic methods applied to orienteering. The Journal of the Operational Research Society 35(9):797–809

Tsiogkas N, Lane DM (2018) Dcop: Dubins correlated orienteering problem optimizing sensing missions of a nonholonomic vehicle under budget constraints. IEEE Robotics and Automation Letters 3(4):2926–2933

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

Váňa P, Faigl J (2018) Optimal solution of the generalized dubins interval problem. In: Robotics: Science and Systems (RSS)

Vansteenwegen P (2018) The Orienteering Problem: Test Instances – Department of Mechanical Engineering, University of Leuven. URL `http://www.mech.kuleuven.be/en/cib/op/#OP`, cited on 2018-07-21

Vansteenwegen P, Souffriau W, Oudheusden DV (2011) The orienteering problem: A survey. European Journal of Operational Research 209(1):1–10

Wang C, Ma F, Yan J, De D, Das SK (2015) Efficient aerial data collection with uav in large-scale wireless sensor networks. International Journal of Distributed Sensor Networks 11(11)

Wren A, Holliday A (1972) Computer scheduling of vehicles from one or more depots to a number of delivery points. Operational Research Quarterly (1970-1977) 23(3):333–344

Yu J, Schwager M, Rus D (2016) Correlated orienteering problem and its application to persistent monitoring tasks. IEEE Transactions on Robotics PP(99):1–13

Chapter 5

Variable Neighborhood Search for the Set Orienteering Problem and its application to other Orienteering Problem variants

The third core publication of this thesis is the manuscript [3c] published in the European Journal of Operational Research concerning the Set Orienteering Problem (SOP).

[3c] **R. Pěnička**, J. Faigl, and M. Saska, "Variable neighborhood search for the set orienteering problem and its application to other orienteering problem variants," *European Journal of Operational Research*, vol. 276, no. 3, pp. 816 –825, 2019

The paper introduces the VNS-based method for the SOP [26]. The SOP is a recently proposed generalization of the OP where the target locations (also denoted as customers or nodes) are grouped in clusters, and the profit associated with each cluster is collected by visiting at least one node in the cluster. The objective of the SOP is to maximize the collected reward clusters for a limited budget path, together with the determination of individual nodes to be visited within the clusters.

For the data collection planning for aerial vehicles, the SOP can be seen as a discrete generalization of all the studied variants of the OP. In the sampling-based approaches to the (D)OP(N), the sampled heading angles or the neighborhood positions of a single target are the nodes of one cluster. For the SOP, the selection of a single heading and neighborhood position sample in (D)OP(N) is sufficient to collect the reward from the target (cluster).

In the paper, we introduce a novel ILP formulation for the SOP to find the optimal solution for small and medium-sized problems. Besides, the proposed VNS-based method for the SOP is a generalized variant of the algorithm for DOP [1c] and DOPN [2c]. We employ similar *shaking* and *local search* procedures; however, we further extend the *local search* procedure by two main aspects. The procedure uses a dynamic programming technique to store the shortest paths from the starting and ending clusters to each vertex in the current solution. Furthermore, a fast-to-find lower bound solution (in length) is used to filter out over-budget solutions. Therefore, the one cluster modifications of the current solution done in the *local search* procedure can be evaluated faster.

The computational results of the introduced ILP formulation show significantly better computational times in the CPLEX solver than the existing formulation [26]. The VNS-SOP method also performs faster than the existing matheuristic based on tabu search for the SOP [26]. Furthermore, the VNS-SOP is able to improve solutions of several large SOP instances over the best-known solutions. Finally, we show the application of the SOP on sampling-based DOP and OPN. For most instances both the DOP and OPN are, for a given sampling, solved optimally using the introduced ILP formulation in the CPLEX. The proposed VNS-SOP is shown to produce optimal solutions of the DOP and OPN when compared with the ILP solutions.

The author of this thesis contributed 70 % on this core publication. The co-authors contributed by giving valuable feedback about the method and the manuscript.

This is the author's accepted version of an article that has been published in European Journal of Operational Research.
The final version of paper is available at https://doi.org/10.1016/j.ejor.2019.01.047

Variable Neighborhood Search for the Set Orienteering Problem and its application to other Orienteering Problem variants

Robert Pěnička*, Jan Faigl, Martin Saska

Czech Technical University, Faculty of Electrical Engineering,
Technicka 2, 166 27, Prague, Czech Republic

Abstract

This paper addresses the recently proposed generalization of the Orienteering Problem (OP), referred to as the Set Orienteering Problem (SOP). The OP stands to find a tour over a subset of customers, each with an associated profit, such that the profit collected from the visited customers is maximized and the tour length is within the given limited budget. In the SOP, the customers are grouped in clusters, and the profit associated with each cluster is collected by visiting at least one of the customers in the respective cluster. Similarly to the OP, the SOP limits the tour cost by a given budget constraint, and therefore, only a subset of clusters can usually be served. We propose to employ the Variable Neighborhood Search (VNS) metaheuristic for solving the SOP. In addition, a novel Integer Linear Programming (ILP) formulation of the SOP is proposed to find the optimal solution for small and medium-sized problems. Furthermore, we show other OP variants that can be addressed as the SOP, i.e., the Orienteering Problem with Neighborhoods (OPN) and the Dubins Orienteering Problem (DOP). While the OPN extends the OP by collecting a profit within the neighborhood radius of each customer, the DOP uses airplane-like smooth trajectories to connect individual customers. The presented computational results indicate the feasibility of the proposed VNS algorithm and ILP formulation, by improving the solutions of several existing SOP benchmark instances and requiring significantly lower computational time than the existing approaches.

Keywords: Routing, Orienteering Problem, Variable Neighborhood Search

1. INTRODUCTION

The Set Orienteering Problem (SOP) belongs to a large class of routing problems with profits, where the objective is to find a tour that maximizes the collected profit for a given budget, or minimizes the tour length while ensuring at least a predefined profit, or the objective function combines profit maximization with tour length minimization (Feillet et al., 2005).

One of the well-studied routing problems with profits is the Orienteering Problem (OP), which was introduced into operational research by Tsiligirides (1984). The OP stands to find a tour with a limited length that maximizes the profit collected from a visited subset of the given nodes using predefined starting and ending depot locations. The OP thus combines two well-known combinatorial optimization problems, the Knapsack Problem (KP) and the Traveling Salesman Problem (TSP). While the Knapsack part of the problem addresses the maximization of the collected profit by selecting the subset of customers to be visited within the budget, the TSP part finds the sequence to visit selected customers and minimize the tour length in order to fit it within the budget.

The OP has multiple variants and generalizations, as it is shown in surveys by Vansteenwegen et al. (2011) and Gunawan et al. (2016). Among others, the recently introduced SOP proposed by Archetti et al. (2018) is a generalization of the OP where the customers are grouped in clusters. The profit is associated with the individual clusters, and it is collected by visiting at least one customer in the respective cluster. The SOP has been introduced together with a Mixed-Integer Programming (MIP) formulation and

*Corresponding author
Email addresses: penicrob@fel.cvut.cz (Robert Pěnička), faiglj@fel.cvut.cz (Jan Faigl), saskam1@fel.cvut.cz (Martin Saska)

This is the author's accepted version of an article that has been published in European Journal of Operational Research. The final version of paper is available at https://doi.org/10.1016/j.ejor.2019.01.047

a matheuristic solution algorithm.

Like the OP, the SOP is a combination of the Knapsack problem and the Generalized Traveling Salesman Problem (GTSP), studied in Laporte & Nobert (1983). In the GTSP, the customers are grouped in clusters as in the SOP, and the objective is to minimize the tour length for visiting at least one customer in each cluster. Therefore, the GTSP is an extension of the TSP in the same way as the SOP extends the OP.

The profit collection from clusters in the OP has been previously addressed by the Clustered Orienteering Problem (COP) in Angelelli et al. (2014). However, in the COP, all customers within the respective cluster have to be visited to collect the profit. The problem is solved by means of branch-and-cut and tabu search algorithms.

The Correlated Orienteering Problem (CorOP) by Yu et al. (2016) is also related to the SOP. In the CorOP, the profit collected from customers contains a part of the profit of neighboring customers based on the mutual spatial correlation between the visited customers and the neighboring customers. The CorOP thus creates spatially correlated clusters of the customers, where the profit consists of the individual visited customers together with the distance-weighted profit of the neighboring customers. Therefore, the CorOP can be seen as a hybrid combination of the COP and the SOP, as the profit gained from visiting individual customers consists of the portion of the otherwise unvisited neighboring customer's profit. However, the profit can be increased by visiting more customers within the cluster. The therein presented the exact solution of the CorOP is based on the Mixed-Integer Quadratic Programming Yu et al. (2016).

The applications of the SOP, originally introduced by Archetti et al. (2018), are in mass distribution, where the carrier chooses to serve only one customer within a cluster of customers that are afterward served by internal distribution within the cluster. However, applications of the SOP far exceed the application originally outlined. In fact, the SOP can be used for any applications of the GTSP, discussed in Laporte et al. (1996), where the salesman has a limited budget, and cluster profits can be used for prioritization. The travel guide problem is an example of such an application, where the guide aims to maximize the profit from visiting attractions in a limited time, but only one attraction of each kind (cluster) would bring the profit.

Similarly, the SOP can be used for a generalization of the traveling salesman with profit-rated customers and a limited budget, where multiple modes of transport are allowed, but with constrained transport changes. Each cluster then consists of multiple departure modes of transport from the individual customer, and the objective is to maximize the profit while using the most suitable transport option to fit within the time budget.

The SOP can also be applied to several other variants of the OP. The Orienteering Problem with Neighborhoods (OPN) is a generalization of the OP, introduced in Faigl et al. (2016), where the profit from each customer can be collected anywhere within the circular neighborhood of the customer location. The OPN can be addressed, like the SOP, by creating clusters of position samples on a circle around the original customer's locations (see Section 4.1). An application example of the OPN is in sensory network information retrieval, where standalone sensor units displaced throughout the environment can wirelessly communicate within a close distance radius, as discussed in Li et al. (2009). To maximize the profit from information collected within a given time, the data collecting vehicle can save travel costs by retrieving the measured information without visiting the precise position of the sensors.

The Dubins Orienteering Problem (DOP), proposed in Pěnička et al. (2017a), can be used for planning package delivery by dropping from an airplane. The DOP addresses the generalization of the OP for the Dubins vehicle model, introduced by Dubins (1957), where the modeled airplane cannot feasibly travel the tour created by the Euclidean OP with sharp turns between the target locations. The Dubins vehicle has to travel between the locations using a curvature-constrained path with a minimum turning radius. By sampling the heading angle of the Dubins vehicle at the target locations, the DOP can be addressed as the SOP presented here (see Section 4.1).

The contributions of this paper are as follows. We introduce a novel ILP formulation of the SOP to find the optimal solution of small to medium-sized problems within less computational time than the existing MIP formulation. The definition of the SOP extends the existing definition by allowing different starting and ending depot clusters, both of which can have multiple nodes. We propose an algorithm for the SOP based on the Variable Neighborhood Search (VNS) metaheuristic, and we show that its computational

This is the author's accepted version of an article that has been published in European Journal of Operational Research. The final version of paper is available at https://doi.org/10.1016/j.ejor.2019.01.047

times are about one order of magnitude shorter for small and medium-sized problems than the existing tabu search solution. The best-known solutions of several benchmark instances are improved by the proposed VNS algorithm for the SOP further denoted as VNS-SOP. Furthermore, we employed the studied SOP in a solution of other OP variants, such as the DOP and the OPN, both newly addressed as the sampling based SOP that is solved optimally. For reuse by the community and to accelerate research on SOP-related problems, both the ILP-based and VNS-SOP algorithms for the SOP are published as open-source software, together with benchmark datasets for comparison.

The remainder of this paper is organized as follows. The description and the formulation of the problem are presented in the next section. The VNS-SOP algorithm is introduced in Section 3. The computational results are presented in Section 4, and final conclusions are outlined in Section 5.

2. Problem description and formulation

The Set Orienteering Problem is a generalization of the OP where customers are grouped in clusters, and the objective is to find a tour with a predefined starting cluster and ending cluster, a restricted budget, and such that the tour maximizes the profit collected from the visited clusters. A cluster is visited when at least one customer belonging to the cluster has been visited. The herein presented SOP formulation builds upon the existing formulation by Archetti et al. (2018) that is extended by considering possibly two different starting and ending clusters, both with the possibility of having multiple nodes instead of a single depot cluster with one node, as in the original formulation.

The SOP can be defined on a directed graph $G = (V, A)$ with a set of vertices $V = \{v_0, \ldots, v_m\}$ and a set of arcs $A = \{a_{ij}\}$. For each pair of vertices v_i, v_j, there exists an arc a_{ij} with cost c_{ij}. The vertices are clustered into disjoint sets $s_0, \ldots, s_n$, with $S = \{s_0, \ldots, s_n\}$, $s_i \cap s_j = \emptyset$ for $i \neq j$, $0 \leq i, j \leq n$, and each vertex $v_{i=0,\ldots,m}$ is associated with exactly one set in S. All sets $s_{i=0,\ldots,n}$ have the associated profit $p_{i=0,\ldots,n}$ for visiting at least one vertex within the set. The starting set s_0 and the ending set s_n are for simplicity the first and the last sets, respectively, both associated with zero profit ($p_0 = p_n = 0$). The objective is to find a tour that maximizes the collected profit P such that its cost does not exceed the given budget $\mathrm{T}_{\max}$. Assuming that the triangle inequality holds for the arc costs, an optimal tour always includes one vertex per visited cluster (see Archetti et al. (2018)).

For instances with a common depot, as in the original SOP formulation Archetti et al. (2018), an additional copy of such a depot can be used as the ending set. Furthermore, the proposed formulation allows multiple vertices in both starting set ($|s_0| \geq 1$) and ending set ($|s_n| \geq 1$).

Any solution of the SOP can be described by a permutation Σ_k of set indexes, according to which the tour visits the individual sets $\Sigma_k = (\sigma_1, \ldots, \sigma_k)$ with $0 \leq \sigma_i \leq n$, $\sigma_i \neq \sigma_j$ for $i \neq j$ and $\sigma_1 = 0$, $\sigma_k = n$. Beside determining the permutation of the sets, the SOP also requires the vertices in the respective visited clusters to be found. The vertices are represented by their respective indexes $\Pi_k = (\pi_1, \ldots, \pi_k)$, $0 \leq \pi_i \leq m$ and $v_{\pi_i} \in s_{\sigma_i}$ for $i \in (1, \ldots, k)$. Using the above notation, the SOP can be defined as follows:

$$\begin{aligned} \underset{\Sigma_k, \Pi_k, k}{maximize}\, \mathrm{P} &= \sum_{i=1}^{k} p_{\sigma_i} \\ \text{s.t. } & \sum_{i=2}^{k} c_{\pi_{i-1},\pi_i} \leq \mathrm{T}_{\max} \ , \\ & v_{\pi_i} \in s_{\sigma_i} \quad \forall i = 1, \ldots, k \ , \\ & \sigma_1 = 0 \ , \ \sigma_k = n \ . \end{aligned} \tag{1}$$

The SOP can also be formulated as an Integer Linear Program. Unlike the MIP proposed by Archetti et al. (2018) for the SOP, the proposed formulation does not contain the binary variables of the vertices, and requires only two variable types. Furthermore, the proposed model uses subtour elimination constraints (SECs) like the COP formulation (see Angelelli et al. (2014)), instead of the connectivity cuts of the previous MIP formulation.

The decision variables used in the proposed ILP are:

- y_i: binary variable equal to 1 if at least one customer is visited in the set s_i and 0 otherwise;
- x_{ij}: binary variable equal to 1 if arc a_{ij} is traversed and 0 otherwise.

This is the author's accepted version of an article that has been published in European Journal of Operational Research. The final version of paper is available at https://doi.org/10.1016/j.ejor.2019.01.047

The proposed ILP formulation of the SOP is:

$$maximize \sum_{s_i \in S} p_i y_i, \tag{2}$$

$$\text{s.t.} \sum_{a_{ij} \in A} c_{ij} x_{ij} \leq \mathrm{T}_{\max}, \tag{3}$$

$$\sum_{v_i \in V \setminus \{s_q\}} x_{ij} = \sum_{v_i \in V \setminus \{s_q\}} x_{ji} \quad \forall s_q \in S \setminus \{s_0, s_n\} \, , \forall v_j \in s_q, \tag{4}$$

$$\sum_{v_i \in V \setminus \{s_q\}} \sum_{v_j \in s_q} x_{ij} = y_q \qquad \forall s_q \in S \setminus \{s_0\}, \tag{5}$$

$$\sum_{v_i \in V \setminus \{s_q\}} \sum_{v_j \in s_q} x_{ji} = y_q \qquad \forall s_q \in S \setminus \{s_n\}, \tag{6}$$

$$\sum_{v_i \in U} \sum_{v_j \in U} x_{ij} \leq \sum_{s_q \in U \setminus \{s_t\}} y_q \quad \forall U \subset S \setminus \{s_0, s_n\}, \forall s_t \in U, \tag{7}$$

$$y_0 = 1, y_n = 1, \tag{8}$$

$$y_q \in \{0,1\}, \quad s_q \in S, \qquad x_{ij} \in \{0,1\}, \quad a_{ij} \in A. \tag{9}$$

The objective function (2) calls for the maximization of the collected profit. The budget constraint (3) limits the total length of the arcs that are used. Constraints (4) ensure that each vertex, except for those in the starting and ending clusters, has the same number of entering and leaving arcs. Each visited cluster, except for the starting cluster, must have an entering arc. This is ensured by constraints (5). Similarly, constraints (6) ensure that one leaving arc must be selected for all visited clusters different from s_n. Constraints (7) are the SECs. Constraints (8) ensure that both the starting cluster and the ending cluster are visited. Finally, constraints (9) define the domains of the variables.

Both (1) and (2)-(9) aim at finding a permutation of a subset of the clusters and the vertices to visit inside the selected clusters at the same time. However, for the VNS-SOP algorithm, the problem can be partially separated into: (i) selection of the clusters to visit; (ii) determination of the order of visits to the selected clusters; and (iii) selection of the vertices to visit in the chosen clusters. For a given permutation of clusters Σ_k, the solution of (i) and (ii), the subproblem (iii) of selecting individual vertices Π_k within clusters can be addressed as finding the shortest path in a graph of the visited clusters, see Fig. 1a.

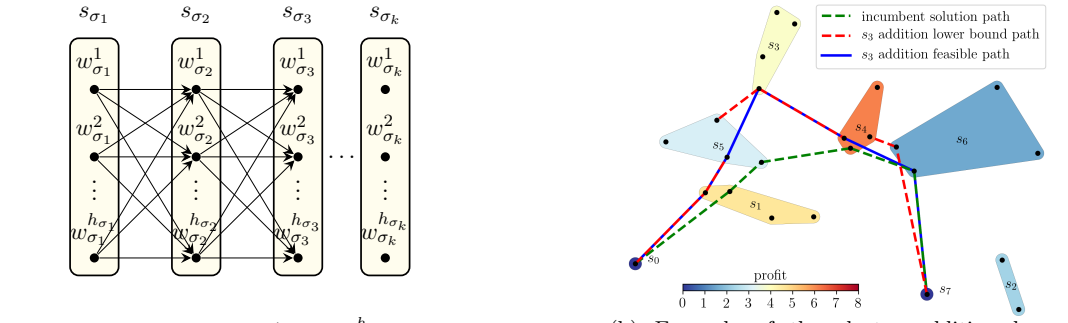
3. Variable neighborhood search algorithm for the SOP

The designed heuristic solution of the Set Orienteering Problem is based on the Variable Neighborhood Search metaheuristic proposed by Mladenović & Hansen (1997) for combinatorial optimization. The metaheuristic uses a greedy initial solution that minimizes the distance per additional profit gained by visiting a new, previously not visited cluster. Afterward, the VNS tries to improve the currently best incumbent solution by a set of predefined neighborhood operators. The VNS metaheuristic was introduced for the OP by Sevkli & Sevilgen (2006), and similar neighborhood operators have been further used for initial solutions of the DOP in Pěnička et al. (2017a) and the OPN in Pěnička et al. (2017b).

In both the SOP initialization procedure and the VNS-SOP algorithm itself, the solution of the SOP is represented only by a sequence of clusters Σ_k. For a given sequence Σ_k with k clusters, the resulting path can be found using a shortest path search in a search graph that is visualized in Fig. 1a as a path connecting the starting cluster $s_{\sigma_1} = s_0$ with the ending cluster $s_{\sigma_k} = s_n$. The graph contains only the arcs between adjacent clusters of Σ_k, and therefore, the shortest path contains exactly one vertex in each cluster of Σ_k and defines the vertices used for a given sequence. The shortest path is found by a dynamic programming breadth-first search storing the shortest path from the starting cluster s_{σ_1} to all vertices in s_{σ_l} iteratively for $l = 2, \ldots, k$ by using already-stored shortest paths to vertices in the preceding cluster and the corresponding arcs connecting adjacent clusters. The shortest path over Σ_k is then defined as the shortest path found among the vertices of ending cluster s_{σ_k}. The proposed algorithm therefore operates with the sequence of clusters and internally calculates the vertices Π_k within the visited clusters such that the overall path length is minimized.

The proposed VNS algorithm for the SOP, including the greedy initialization, consists largely of simple cluster sequence modifications, where a single cluster is added, moved or removed from an existing cluster sequence Σ_k. In the case of the cluster addition, the evaluation of the resulting path length requires only to calculate the connection from the previous to the following cluster in the sequence. However, the shortest path from the starting cluster to each vertex in the preceding cluster and also the shortest path from

This is the author's accepted version of an article that has been published in European Journal of Operational Research. The final version of paper is available at https://doi.org/10.1016/j.ejor.2019.01.047



(a) Σ_k with vertices $\{w^1_{\sigma_l}, \ldots, w^{h_{\sigma_l}}_{\sigma_l}\} \in s_{\sigma_l}, \forall l \in (1, \ldots, k)$.

(b) Example of the cluster addition lower bound.

Figure 1: Graph of cluster sequence Σ_k in (a) and the cluster addition lower bound in (b).

each vertex in the subsequent cluster to the ending cluster has to be known. Therefore, we propose to employ dynamic programming technique to store the shortest paths for each vertex (in the current cluster sequence Σ_k) from the starting and ending clusters to quickly evaluate the simple modifications without searching the shortest path in the whole graph.

The proposed VNS algorithm is further time optimized by using fast denial of the simple sequence modifications that produce solutions with over-budget length. For a typical SOP near-optimal solution, the total path length is close to the budget limit $\mathrm{T_{max}}$, and almost all modifications, such as cluster addition or movement, produce an over-budget solution. To quickly determine such cases, the lower bound distance between each cluster pair, i.e., the minimal-length arc between the cluster pair, is found and stored before the initial solution of the SOP is created. Then, e.g., for adding the cluster s_3 between the clusters s_5 and s_4, as shown in Fig. 1b, the lower bound is first tested to be within the budget while using the minimal-length arcs from s_5 to s_3 and from s_3 to s_4. The lower bound path further consists of the shortest paths to cluster s_5 from the starting cluster and the shortest path to the ending cluster from s_4, both found as the shortest distance stored by the dynamic programming technique for an incumbent solution among the vertices in s_5 and s_4, respectively. The proposed lower bound can be unfeasible, as it might use different vertices in the cluster being added, the previous and following clusters. However, the feasible solution cannot be shorter, and finding the lower bound is of low complexity, e.g., $\mathcal{O}(|s_5| + |s_4|)$. Thus, simple cluster operations can be found to produce over-budget solutions without searching the vertices to be used in the previous s_5, the newly added s_3, and the following cluster s_4 in the cluster sequence of the feasible solution with the complexity of, e.g., $\mathcal{O}(|s_5||s_3| + |s_3||s_4|)$.

Since both the proposed ILP-based and VNS-SOP solution algorithms for the SOP employ the greedy construction of the initial solution, the procedure is described first, followed by the introduction of the VNS metaheuristic for the SOP.

3.1. Initial solution construction procedure

The proposed construction procedure of the initial solution uses a greedy approach that minimizes the additional length of the path per additional profit. The initially empty sequence of clusters Σ_2 contains only the predefined starting and ending clusters, and the path uses the shortest arc between these depot clusters. Then, in each step of the construction procedure, a non-visited cluster s^*_i and a position j^* for $1 < j^* < k$ within the current sequence Σ_k is found, using the rule

$$s^*_i, j^* = \underset{i \notin \Sigma_k, i \in \Sigma_{k+1}, 1<j<k+1, \sigma_j = i}{\operatorname{argmin}} \frac{\mathcal{L}(\Sigma_{k+1}) - \mathcal{L}(\Sigma_k)}{p_i}. \tag{10}$$

The selection rule (10) uses the difference of the lengths $\mathcal{L}(\Sigma_k)$ and $\mathcal{L}(\Sigma_{k+1})$ of the shortest paths over the cluster sequences Σ_k and Σ_{k+1}, respectively. This requires an evaluation of multiple simple addition operations, and therefore, the cluster distance lower bounds and the dynamic programming technique with storing of the shortest distance to the terminating clusters are used. The initialization procedure terminates as soon as the budget limit does not allow any other non-visited cluster to be added.

This is the author's accepted version of an article that has been published in European Journal of Operational Research. The final version of paper is available at https://doi.org/10.1016/j.ejor.2019.01.047

3.2. Variable Neighborhood Search algorithm

The Variable Neighborhood Search algorithm consists of two main procedures, the *shake* procedure and the *local search* procedure, which iteratively try to improve the best found incumbent solution. The shake procedure uses random changes of the incumbent solution to get away from a possible local maximum. The local search procedure then extensively searches around the randomly created solution to find a possibly better solution than the actual incumbent. The VNS thus optimizes the incumbent solution using a combination of the shake and local search procedures with the predefined operators in variably large solution space neighborhoods.

In order to uniquely represent any solution of the SOP inside the VNS, the solution is represented by a vector $u = (s_{\sigma_2}, \ldots, s_{\sigma_{k-1}}, s_{\sigma_{k+1}}, \ldots, s_{\sigma_n})$ of all clusters, including the not visited clusters, with the exception of the depot clusters s_{σ_1} and s_{σ_k}. Only the first $k-2$ clusters $(s_{\sigma_2}, \ldots, s_{\sigma_{k-1}})$ are feasibly visited between the starting cluster $s_{\sigma_1} = s_0$ and the ending cluster $s_{\sigma_k} = s_n$ within the $\mathrm{T_{max}}$ budget limit forming the solution sequence Σ_k. The individual visited vertices Π_k in the respective visited clusters Σ_k are always calculated using the breadth-first search for the shortest path over Σ_k in the graph Fig. 1a. The number of visited clusters in the solution vector u is maximized, i.e., we select the largest k possible for the given u and $\mathrm{T_{max}}$ such that the ending cluster $s_{\sigma_k} = s_n$ is reached within the budget.

VNS-SOP algorithm is summarized in Alg. 1. It starts with the construction of the initial solution and then tries to improve the solution until the stopping criteria are met. A combination of the maximal computational time together with the limited number of iterations and the number of iterations without improvement is used as the stopping rule. In each iteration, the shake procedure is applied, followed by the local search procedure varying the neighborhood operators based on the variable l (with $1 \leq l \leq l_{max} = 2$). When the profit $P(u'')$ of the solution u'' found by the local search exceeds the profit of the incumbent solution $P(u)$, and its length $\mathcal{L}(u'')$ is within the budget, the incumbent solution is changed to the newly-found solution. The algorithm applies all neighborhood operators during a single iteration and thus increases the size of the examined solution space neighborhood.

Algorithm 1: Variable Neighborhood Search for the SOP

Input : S - customer sets, $\mathrm{T_{max}}$ - maximal allowed budget
Output: u - solution path

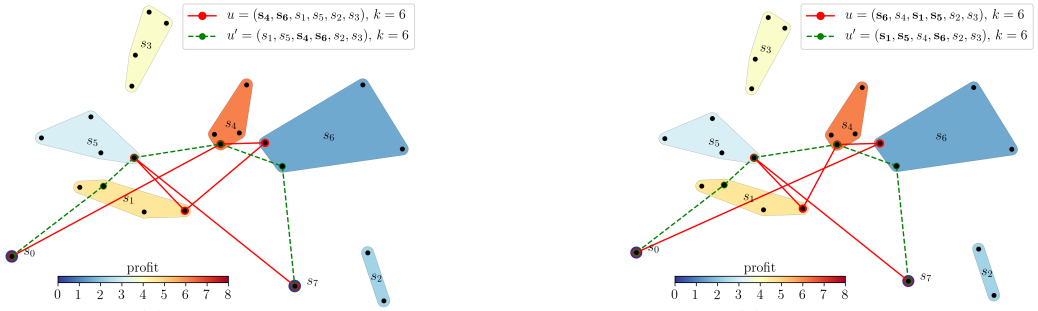
```
u ← createInitialSolution(S,T_max)
while stopping conditions not met do
    l ← 1
    while l ≤ l_max do
        u' ← shake(u, l)
        u'' ← local_search(u', l)
        if L(u'') ≤ T_max and P(u'') > P(u) then
            u ← u''
            l ← 1
        else
            l ← l + 1
```

Shake procedure

The shake procedure of the employed VNS randomly changes the actual incumbent solution to get away from the possible local maximum. It consists of two random operators that modify the solution vector u. By changing u, the operators can change the order of traversing the clusters defining Σ_k and can also add some previously not visited clusters σ_i, $i > k$ to Σ_k. The **Path move** operator and the **Path exchange** operator move or exchange large parts of the solution vector u, and thus create a new solution u' away from the original incumbent. A detail description of the operators follows, and an example of the operators is shown in Fig. 2.

- The **Path move** ($l = 1$) operator randomly selects a part of the solution vector u and moves it to a randomly selected position. This can be done by selecting three random positions inside u, e.g., $1 < i_1 < i_2 < i_3 \leq n$, $i_{1\ldots3} \neq k$, and moving the sequence of clusters $s_{\sigma_j}, i_1 \leq j \leq i_2$ further in u after the cluster $s_{\sigma_{i_3}}$. Alternatively, with the same probability as moving the cluster sequence further in u, a cluster sequence $s_{\sigma_j}, i_2 \leq j \leq i_3$ is moved before $s_{\sigma_{i_1}}$.
- The **Path exchange** ($l = 2$) operator exchanges two randomly selected non-overlapping parts of the solution vector. Similarly to the path move, the path exchange can be implemented using four randomly selected positions within u, e.g., $1 < i_1 < i_2 < i_3 < i_4 \leq n$, $i_{1\ldots4} \neq k$. Afterwards, the cluster sequence $s_{\sigma_j}, i_1 \leq j \leq i_2$ is exchanged with the sequence $s_{\sigma_h}, i_3 \leq h \leq i_4$.

This is the author's accepted version of an article that has been published in European Journal of Operational Research. The final version of paper is available at https://doi.org/10.1016/j.ejor.2019.01.047



(a) Path move (b) Path exchange

Figure 2: Examples of the shake operators that (a) move the clusters (s_4, s_6) after s_5; and (b) an exchange of (s_6) with (s_1, s_5).

Local search procedure

The VNS local search procedure is used for an extensive search around the randomly created solution vector u' produced by the shake procedure. A close neighborhood of the solution u' is searched using the operators **One cluster move** and **One cluster exchange** to find a better solution.

The implemented local search procedure originates from the randomized variant of the VNS (RVNS) in which the local search operators are applied randomly to the solution vector instead of being applied according to deterministic rules as in the regular VNS. Both operators test simple modifications of the solution vector u', where only one (One cluster move) or two (One cluster exchange) clusters are moved within u'. Each operator tries n^2 such random modifications, and only those not worsening the quality of the solution are applied to u' before examining further modifications. Each operator thus implements a hill climbing paradigm guaranteeing that no decrease occurs in the solution quality.

The two local search operators examine numerous cluster sequence modifications to improve the solution. By employing the dynamic programming technique, with storing the shortest paths inside the solution u', the evaluation of n^2 such modifications is significantly speeded up. Furthermore, each modification of this type is examined in advance to check whether its lower bound does not produce a solution with an over-budget length. The local search operators illustrated in Fig. 3 are as follows:

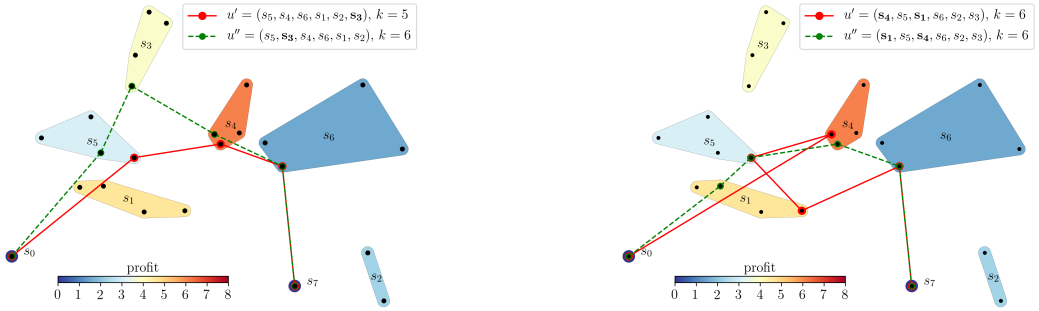
- The **One cluster move** $(l = 1)$ operator repeatedly tries modifications where one random cluster within the solution vector is moved into a different randomly selected position. The modification can be realized by selecting two random positions $1 < i_1 < i_2 \leq n$, $i_{1,2} \neq k$, within the solution vector u'. Afterwards, one cluster is moved either $s_{\sigma_{i_1}}$ after $s_{\sigma_{i_2}}$ or $s_{\sigma_{i_2}}$ before $s_{\sigma_{i_1}}$ with the equal probability. Only modifications not decreasing the solution quality are applied to u' before examining further modifications.
- The **One cluster exchange** $(l = 2)$ is similar to the previous local search operator; however, instead of moving one cluster, it exchanges two randomly selected distinct clusters within the solution vector. Using two random indexes $1 < i_1 < i_2 \leq n$, $i_{1,2} \neq k$, a single modification of this operator is made by exchanging clusters s_{i_1} and s_{i_2} in u'. The operator examines n^2 such exchange modifications and always applies only those that do not decrease the solution quality.

4. Computational tests

The proposed VNS-SOP algorithm and the novel ILP formulation have been evaluated on the existing SOP benchmark instances. Furthermore, the methods are tested on the instances of the OP with Neighborhoods (OPN) and the Dubins OP (DOP) addressed as a sampling-based SOP.

Both VNS-SOP and the ILP-based solution algorithms are implemented in C++, and the computational experiments have been performed on a standard PC equipped with an Intel Core i7, clocked at 3.40 GHz, and 16GB of RAM, by using a single core for each run. The ILP formulation (2)-(9) proposed to find the optimal solution of the SOP is solved by means of CPLEX 12.6.1. The subtour elimination constraints (7) are dynamically added to the formulation when found to be violated. The greedy construction procedure used for VNS-SOP is also used for creating an initial feasible solution for the CPLEX solver when addressing the ILP formulation (2)-(9). The

This is the author's accepted version of an article that has been published in European Journal of Operational Research. The final version of paper is available at https://doi.org/10.1016/j.ejor.2019.01.047



(a) One cluster move (b) One cluster exchange

Figure 3: Example of cluster sequence modifications made by the local search operator.

maximal computational time for the CPLEX solver has been set to 9 hours.

The stopping condition of VNS-SOP is a combination of the following three criteria: a) the maximum of 2000 iterations, b) 1000 iterations without an improvement, and c) the maximum computational time of 20 minutes. Each problem instance has been solved 20 times to obtain valid statistical results of VNS-SOP.

In the following section, we describe the benchmark datasets that are used. Then, we present the computational results obtained by applying VNS-SOP and by solving the ILP formulation (2)-(9) by means of the CPLEX solver on the GTSP dataset instances used for the SOP in Archetti et al. (2018). Finally, the computational results of the OPN and the DOP are presented, both addressed as a sampling-based SOP.

4.1. Test instances

The evaluated benchmark instances can be categorized into three types. The first is based on the dataset created for the GTSP by Fischetti et al. (1997). The other two datasets are based on the benchmark instances created by Tsiligirides (1984) for the OP that are used to generate test instances for the OPN and DOP with a predefined number of samples forming the clusters for the SOP from the original OP nodes. The OP datasets for the OPN and DOP are 100× scaled and use the rounded up distances between nodes instead of the exact Euclidean distances. The benchmark datasets are available online together with the implementations of the proposed methods[1].

[1] `https://github.com/ctu-mrs/vns-sop`

GTSP dataset

The GTSP dataset has been previously used for evaluating the performance of the matheuristic based on tabu search for the SOP (MASOP) proposed in Archetti et al. (2018), and it is therefore used for comparison with the SOP solvers proposed here. To modify the GTSP dataset for the SOP with a single depot, the authors of MASOP removed the first node in each dataset instance from its original GTSP cluster and added the node to a new depot cluster s_0. The budget limit T_{max} for individual instances is generated using the ω ratio of the GTSP*, the best known cost of the GTSP solution taken from Fischetti et al. (1997) [2]. Two types of cluster profit $\mathsf{p}_g \in \{g_1, g_2\}$ are considered in the dataset. The first, g_1, uses the cluster profit $p_i = |s_i|$ equal to the number of nodes in the respective cluster. The second type, g_2, uses the pseudo-random profit of each node j, with the exception of the depot node $j = 0$ with $p_0 = 0$, equal to $1 + (7141j) mod (100)$ with the consequent cluster profit summed from its respective nodes[3]. As we consider predefined starting and ending clusters in our SOP formulation, the original single depot cluster is duplicated and is used as both a starting cluster and an ending cluster.

[2] In Archetti et al. (2018), the solutions with $\omega = 1$ are expected to collect all clusters within the dataset instance. However, this is not feasible due to the newly created depot cluster, which necessarily adds a travel cost compared to the GTSP* solution. Furthermore, the SOP dataset uses rounded up 'CEIL_2D' edge costs, which is reasonable for the budget limited SOP, but it further increases the length of the shortest cycle over all clusters. The original GTSP dataset uses rounding to the nearest integer value 'EUC_2D'.

[3] The originally proposed g_2 rule for the SOP in Archetti et al. (2018) and previously also used for the COP in Angelelli et al. (2014) uses the profit formula $1 + (7141j + 73) mod (100)$. However, the dataset and its results presented for the SOP match with the formula $1 + (7141j) mod (100)$.

This is the author's accepted version of an article that has been published in European Journal of Operational Research. The final version of paper is available at https://doi.org/10.1016/j.ejor.2019.01.047

OPN dataset

The second benchmark instances use the Set 2 dataset with 21 nodes originally proposed by Tsiligirides (1984) for the OP. In the OPN proposed by Faigl et al. (2016), the profit of individual nodes can be collected within a circular neighborhood of each node with predefined neighborhood radius δ. The solution can be addressed using a sampling-based approach, where for each original node (except the starting and ending nodes) in the dataset, a cluster with o_n equidistantly sampled nodes is created on the δ-radius circle. For all nonterminating original OP nodes with the position $v_i = (x_i, y_i), i \in (2, \dots, n-1)$, the newly created clusters s_i have o_n sampled neighborhood nodes with the positions $(x_{i,j}, y_{i,j}), i \in (2, \dots, n-1), j \in (0, \dots o_n - 1)$. The positions of the neighborhood nodes are determined as:

$$(x_{i,j}, y_{i,j}) = (x_i, y_i) + \delta \left(\cos\left(\frac{2j\pi}{o_n}\right), \sin\left(\frac{2j\pi}{o_n}\right) \right). \tag{11}$$

The neighborhood radius used for generating the dataset is $\delta = 50$. Both terminating clusters contain only the original nodes. The created SOP dataset for the OPN thus consists of 21 clusters with $2 + 19o_n$ nodes. The dataset does not contain overlapping clusters although the original OPN can have overlapping δ-radius circles. The SOP dataset for the OPN is approximation of the original instances where more samples o_n better approximates the instances at the cost of the increased number of nodes.

DOP dataset

The second shown variant of the OP solvable as the SOP is the DOP introduced in Pěnička et al. (2017a). In the DOP, the airplane-like vehicle is approximated by the Dubins vehicle model proposed in Dubins (1957). A solution of the OP contains straight line segments between nodes with sharp turns, which are not feasible for the Dubins vehicle. In the DOP, the aerial vehicle has to turn with a given turning radius ρ. Dubins showed that for a curvature-constrained vehicle of this type, the optimal length maneuver between two locations with initial and final heading angles is one of the six possible Dubins maneuvers which satisfy the triangular inequality. The Dubins vehicle state $q = (x, y, \theta)$ can be described by its position in the plane $(x, y) \in \mathbb{R}^2$ and its heading angle $\theta \in S^1$, i.e., its state q belongs to the special Euclidean group $q \in SE(2)$. To solve the DOP, we have to consider the heading angle at each node to connect the consecutive Dubins maneuvers between nodes feasibly, and thus the selection of the heading angles is a part of the optimization due to their influence to the length of the respective Dubins maneuvers. Similarly to the OPN dataset, a sampling-based approach with heading angles at the given nodes can approximate the original DOP by creating clusters of the SOP. For all original nodes v_i, $i \in (1, \dots, n)$, the created clusters s_i contain o_h nodes with equidistantly sampled heading angle $\theta_{i,j}$ for $j \in (0, \dots o_h - 1)$. The individual nodes $q_{i,j}$ representing the Dubins vehicle states are

$$q_{i,j} = (x_{i,j}, y_{i,j}, \theta_{i,j}) = \left(x_i, y_i, \frac{2j\pi}{o_h} \right). \tag{12}$$

The minimal turning radius used in the created dataset is $\rho = 50$. The dataset for the DOP consists of asymmetric SOP instances, as the Dubins maneuver has a different length when the initial and final vehicle states of the maneuver are exchanged.

An example of the found solutions of the SOP on the GTSP, OPN and DOP test instances is shown in Fig. 4. Figure 4a shows the solution on the GTSP 11berlin52 dataset for $\omega = 0.6$ and $\mathsf{p}_g = g_1$. Figure 4b and 4c are example solutions of the OPN and DOP both with $\mathrm{T}_{\max} = 3000$ on the Set 2 dataset, using $o_n = 8, \delta = 50$ in the case of the OPN and $o_h = 8, \rho = 50$ for the DOP.

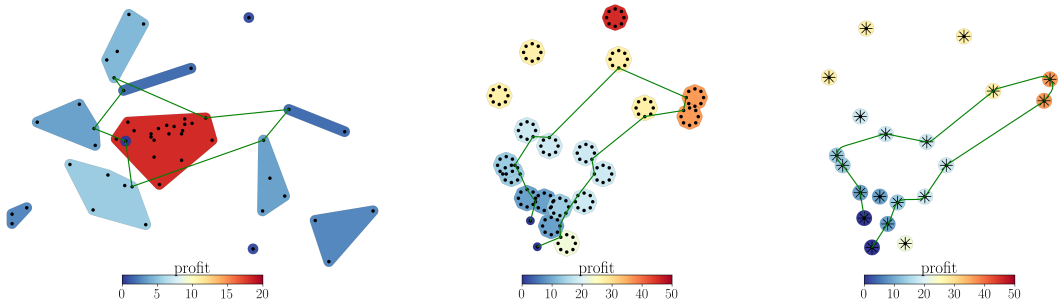
4.2. Computational results on the GTSP dataset

The proposed ILP formulation and VNS-SOP have been evaluated on the GTSP dataset instances, and have been compared with the existing MIP formulation and the matheuristic based on tabu search (MASOP), both proposed by Archetti et al. (2018).

The results shown in Table 1 concern small instances with up to 76 nodes and 16 clusters. Both cluster profit types $\mathsf{p}_g \in \{g_1, g_2\}$ are considered together with various ω ratios of the GTSP* solution and the corresponding budget limit $\mathrm{T}_{\max}$. For each method are reported the collected profit P and the computational time T in seconds. The collected profit of VNS-SOP has been identical in all runs, and the reported computational time is the average from 20 runs.

Table 1 shows that solving to optimality the ILP formulation (2)-(9) requires significantly less computational time than solving the MIP formulation proposed in Archetti et al. (2018). Furthermore, the computational times of VNS-SOP are about one order of magnitude lower than those of MASOP, while

This is the author's accepted version of an article that has been published in European Journal of Operational Research. The final version of paper is available at https://doi.org/10.1016/j.ejor.2019.01.047



(a) 11berlin52 (b) OPN Set 2 (c) DOP Set 2

Figure 4: Example solutions of the SOP on selected dataset instances.

Table 1: Comparison with exisitng methods on small GTSP dataset instances.

instance	p_g	ω	$\mathbf{T_{max}}$	MIP		MASOP		ILP		VNS-SOP	
				P	**T**	**P**	**T**	**P**	**T**	**P**	**T**
11berlin52	g_1	0.4	1616	37	47.07	37	1.75	37	1.08	37	0.11
11berlin52	g_2	0.4	1616	1829	65.96	1829	1.70	1829	1.18	1829	0.11
11berlin52	g_1	0.6	2424	43	777.88	43	2.40	43	4.24	43	0.16
11berlin52	g_2	0.6	2424	2190	1532.91	2190	2.64	2190	1.34	2190	0.15
11berlin52	g_1	0.8	3232	47	2648.04	47	7.17	47	4.63	47	0.19
11berlin52	g_2	0.8	3232	2384	3833.50	2384	6.61	2384	7.67	2384	0.19
11eil51	g_1	0.4	69	24	39.72	24	1.85	24	2.54	24	0.09
11eil51	g_2	0.4	69	1279	40.13	1279	1.97	1279	2.81	1279	0.09
11eil51	g_1	0.6	104	39	34.64	39	5.13	39	1.67	39	0.14
11eil51	g_2	0.6	104	1911	204.65	1911	4.74	1911	3.01	1911	0.14
11eil51	g_1	0.8	139	43	1586.67	43	2.30	43	16.51	43	0.18
11eil51	g_2	0.8	139	2114	1520.67	2114	1.93	2114	40.32	2114	0.20
14st70	g_1	0.4	126	33	9666.29	33	4.43	33	16.65	33	0.14
14st70	g_2	0.4	126	1672	4396.77	1672	4.35	1672	28.50	1672	0.15
14st70	g_1	0.8	252	65	18227.23	65	8.80	65	959.59	65	0.31
14st70	g_2	0.8	252	3355	30851.18	3355	7.89	3355	228.84	3355	0.33
16eil76	g_1	0.4	83	40	4987.09	40	3.88	40	86.18	40	0.19
16eil76	g_2	0.4	83	2223	4939.08	2223	4.73	2223	37.55	2223	0.20
16eil76	g_1	0.6	125	59	29565.85	59	2.40	59	64.31	59	0.31
16eil76	g_2	0.6	125	3119	21127.41	3119	6.28	3119	108.75	3119	0.32

the solution value is optimal for all the runs. We recall that in Archetti et al. (2018) the MIP formulation was solved by means of CPLEX 12.6 and the experiments were carried on a standard PC equipped with Intel Core i7 clocked at 2.80 GHz. Thus, the computational time improvement obtained can be only partially justified by the newer version of the CPLEX solver and the better PC used to perform our experiments. The significantly lower computational times suggest that solving the ILP formulation (2)-(9) and computing solutions by VNS-SOP are both themselves less computationally demanding. The ILP model (2)-(9) has fewer variables (no vertex variables) than the MIP formulation. Furthermore, the different SECs are added only when found to be violated, which can save the insertion of all SECs (especially when the lower bound is set to the CPLEX solver using the greedy initial feasible solution).

The results shown in Table 2 compare the performance of the proposed algorithms against that of MASOP for the budget ratio $\omega = 1$ on large instances with up to 1084 nodes. The large instances cannot be solved optimally using the ILP formulation within the given computational time, except four cases. For both profit types, the table shows the solution value P and the computational time T for the solution computed by MASOP. The computational results of VNS-SOP are reported with the maximal P and the average P_{avg} solution values, and also with the average computational time T. The results computed by the CPLEX solver when addressing the ILP formulation (2)-(9) are shown with the maximally achieved solution values during the optimization and with the percentage gap (or the computational time in seconds for the four optimal solutions found). The profits computed by VNS-SOP appear in bold or underlined when found to be larger or smaller, respectively, than those computed by MASOP.

Regarding the results presented in Table 2, VNS-SOP requires less computational time than MASOP for almost all instances with up to 654 nodes. VNS-SOP does not find the best known result for two instances with g_1 profit and for six instances with g_2. For both profit types, the unachieved best solutions occur for the largest instances with 493 nodes and more, where also the computational time is the

This is the author's accepted version of an article that has been published in European Journal of Operational Research. The final version of paper is available at https://doi.org/10.1016/j.ejor.2019.01.047

Table 2: Comparison on large GTSP dataset instances of the SOP with $\omega = 1$.

instance	g_1							g_2						
	MASOP		VNS-SOP			ILP		MASOP		VNS-SOP			ILP	
	P	**T**	**P**	**P_{avg}**	**T**	**P**	**T/gap**	**P**	**T**	**P**	**P_{avg}**	**T**	**P**	**T/gap**
16pr76	74	8.6	74	74.0	0.6	74	*1.4%*	3765	10.6	3765	3765.0	0.6	3765	26567.7
20kroA100	96	10.5	96	96.0	0.8	96	*3.1%*	4868	11.5	4868	4868.0	0.8	4868	*2.9%*
20kroB100	98	13.4	98	98.0	0.8	98	*1.0%*	4916	10.7	4916	4916.0	0.9	4916	*1.9%*
20kroC100	97	10.8	97	96.1	0.9	97	*2.1%*	4882	11.2	4882	4869.2	1.1	4882	*2.6%*
20kroD100	96	10.3	96	96.0	1.0	96	*3.1%*	4838	8.9	4838	4838.0	1.1	4838	*3.5%*
20kroE100	96	9.1	96	96.0	0.9	96	15536.1	4887	9.9	4887	4887.0	1.1	4887	18938.5
20rat99	93	7.6	93	92.9	1.2	85	*15.3%*	4721	8.1	4721	4721.0	1.0	4483	*11.7%*
20rd100	97	10.4	97	97.0	1.3	97	*2.1%*	4929	9.6	4929	4929.0	1.3	4929	*1.6%*
21eil101	97	8.8	**98**	97.8	1.2	98	*1.0%*	4953	20.1	**4993**	4957.0	1.1	4948	*2.1%*
21lin105	102	8.1	102	102.0	1.1	102	*2.0%*	5157	8.4	5157	5157.0	1.1	5101	*2.5%*
22pr107	101	8.6	101	101.0	0.6	101	*5.0%*	5109	8.3	5109	5105.4	0.7	5104	*5.1%*
25pr124	121	11.3	121	121.0	1.3	114	*7.9%*	6173	11.9	6173	6170.2	1.5	6159	*1.2%*
26bier127	125	16.0	125	125.0	1.8	125	*0.8%*	6314	16.2	6314	6314.0	1.8	6314	4967.7
26ch130	127	10.2	127	126.7	1.9	126	*2.4%*	6412	9.7	6412	6382.3	2.2	6412	*1.4%*
28pr136	134	10.2	134	134.0	2.1	134	*0.7%*	6841	12.2	6841	6831.4	1.8	6808	*0.6%*
29pr144	141	17.4	141	141.0	1.7	139	*2.9%*	7195	22.0	7195	7157.3	1.6	7137	*1.5%*
30ch150	144	9.6	**147**	146.7	2.1	134	*11.2%*	7315	12.3	**7394**	7378.2	1.9	6750	*11.6%*
30kroA150	145	11.3	145	144.7	2.2	140	*6.4%*	7361	13.8	7361	7356.6	2.3	7145	*5.4%*
30kroB150	148	15.2	148	148.0	2.5	148	*0.7%*	7445	15.1	7445	7445.0	2.6	7355	*2.4%*
31pr152	147	18.2	147	145.6	1.7	137	*10.2%*	7422	17.8	7422	7355.6	1.9	6545	*17.0%*
32u159	157	15.5	157	155.1	2.2	143	*10.5%*	7991	22.9	**8011**	7965.2	2.8	7666	*4.8%*
39rat195	189	13.0	189	188.9	5.2	164	*18.3%*	9558	11.0	9558	9546.3	4.7	8438	*16.9%*
40d198	196	36.8	196	195.2	5.7	171	*15.2%*	9934	25.7	**9938**	9926.3	6.7	8628	*15.9%*
40kroa200	198	22.8	198	198.0	3.7	189	*5.3%*	10010	24.5	10010	9976.0	3.6	9577	*5.0%*
40krob200	198	19.9	198	198.0	5.0	192	*3.6%*	9990	28.6	9990	9982.7	5.8	9869	*1.9%*
45ts225	221	34.7	221	220.8	7.3	185	*21.1%*	11187	26.2	**11225**	11158.4	7.8	9767	*15.8%*
45tsp225	219	16.7	**220**	219.1	6.1	186	*20.4%*	11103	16.4	**11124**	11063.9	7.1	9615	*17.6%*
46pr226	224	26.9	224	224.0	3.6	222	*1.4%*	11368	26.4	11368	11358.1	4.6	11222	*1.4%*
53gil262	258	27.2	258	254.2	8.3	215	*21.4%*	13050	25.9	13050	13003.6	7.8	10957	*20.4%*
53pr264	262	34.0	262	262.0	7.1	230	*14.3%*	13277	36.9	13277	13277.0	7.4	13277	*0.2%*
56a280	273	33.5	273	270.8	10.4	212	*31.6%*	13971	37.0	13971	13834.9	9.9	11996	*18.2%*
60pr299	296	31.3	296	295.0	12.0	270	*10.4%*	15005	36.8	15005	14974.4	12.1	12138	*24.5%*
64lin318	315	43.4	**316**	313.8	10.6	295	*7.5%*	16013	77.3	16013	15948.0	11.2	15170	*5.7%*
80rd400	397	76.7	**398**	394.3	28.2	342	*16.7%*	20055	48.1	**20140**	19942.2	29.4	17617	*14.4%*
84fl417	415	103.5	415	414.4	18.0	399	*4.3%*	21030	114.7	21030	20956.0	18.1	19766	*6.5%*
88pr439	437	158.0	437	432.4	33.9	415	*5.5%*	22110	132.8	22110	22032.2	33.1	21058	*5.3%*
89pcb442	440	129.5	440	438.2	38.6	361	*22.2%*	22300	95.0	22300	22116.3	36.2	19456	*14.7%*
99d493	490	120.8	490	487.1	67.3	462	*6.5%*	24827	153.1	**24840**	24708.3	66.3	23545	*5.6%*
115rat575	562	91.2	**563**	555.0	76.5	459	*25.1%*	28497	65.9	28361	28043.5	75.6	23192	*25.2%*
115u574	571	204.5	571	569.9	80.3	509	*12.6%*	28888	212.7	28888	28866.5	72.2	26118	*10.9%*
131p654	652	356.0	652	650.6	45.9	640	*2.0%*	32991	360.1	32950	32894.9	44.2	32450	*1.7%*
132d657	649	126.1	649	642.1	108.9	551	*19.1%*	32974	155.9	**33022**	32901.6	100.3	29198	*13.7%*
145u724	716	99.6	**717**	708.6	176.6	564	*28.2%*	36288	116.7	**36316**	35964.8	171.4	29195	*25.1%*
157rat783	767	279.2	760	750.4	225.7	618	*26.5%*	38953	145.5	38487	37999.8	233.1	31279	*26.3%*
201pr1002	994	304.9	994	981.8	480.4	877	*14.1%*	50453	992.7	50172	49760.9	534.7	45314	*11.6%*
212u1060	1057	873.5	1057	1056.3	679.9	950	*11.5%*	53450	798.5	53437	53391.8	641.8	48151	*11.2%*
217vm1084	1078	489.5	1070	1059.7	832.0	942	*15.0%*	54642	655.5	54363	53744.2	733.7	48955	*11.8%*

same as, or larger than, the time required by MASOP. However, VNS-SOP improved the best known solutions for seven g_1 instances and 10 instances with profit type g_2. The high VNS-SOP computational time for the largest instances is most probably caused by the graph search used for finding the optimal selection of the cluster nodes for the particular cluster sequence that is being examined for possible improvement. In the case of a large number of clusters in a sequence, as for the largest SOP instances with $\omega = 1$, the maintenance of the graph with the shortest path from the starting cluster and the ending cluster to each vertex of the current solution requires a significant amount of time in each VNS-SOP iteration. The four ILP optimal solutions that were found show that for $\omega = 1$, the paths do not visit all the clusters in the dataset instance, and VNS-SOP finds solutions with the same optimal value.

A comparison of the results shown in Tables 1 and 2 shows that VNS-SOP robustly finds high quality solutions, and in most cases significantly faster than MASOP. For several instances, VNS-SOP does not find the best known solution, but it improves the solution of a larger number of dataset instances. Furthermore, solving the ILP formulation (2)-(9) by means of the CPLEX solver requires a fraction of the computational time needed to solve the MIP formulation in Archetti et al. (2018).

4.3. Application of the SOP to other OP variants

VNS-SOP and the ILP formulation (2)-(9) are further tested when used to solve the OPN and the DOP. Both problems can be addressed as the SOP by sampling either the neighborhood positions in the OPN or the heading angles in the DOP. In both cases, the resulting problem is an approximation of the original one, i.e., its solution space is a subset of the solution space associated with the corresponding original problem. A similar VNS-based algorithm has previously been used by the authors for solving the DOP in

This is the author's accepted version of an article that has been published in European Journal of Operational Research. The final version of paper is available at https://doi.org/10.1016/j.ejor.2019.01.047

Table 3: Computational results of the OPN solved as the SOP.

$\mathbf{T_{max}}$	$o_n = 4$				$o_n = 8$				$o_n = 12$			
	ILP		VNS-SOP		ILP		VNS-SOP		ILP		VNS-SOP	
	P	**T/gap**	**P**	**T**	**P**	**T/gap**	**P**	**T**	**P**	**T/gap**	**P**	**T**
1500	180	3.3	180	0.2	180	96.8	180	0.4	180	1710.5	180	0.7
2000	230	8.6	230	0.3	230	63.5	230	0.6	230	242.9	230	1.0
2300	230	10.9	230	0.4	240	1798.1	240	0.8	240	14971.0	240	1.2
2500	260	39.5	260	0.4	260	6462.2	260	0.8	260	*10.7%*	260	1.2
2700	280	99.6	280	0.5	290	12165.4	290	0.8	270	*16.3%*	290	1.3
3000	320	432.4	320	0.4	320	*7.0%*	340	1.0	320	*15.6%*	340	1.5
3200	360	261.3	360	0.5	370	*2.7%*	370	0.9	360	*25.0%*	370	1.4
3500	410	708.3	410	0.6	410	*9.8%*	430	1.0	390	*15.4%*	430	1.5
3800	450	35.0	450	0.4	450	4314.8	450	0.9	430	*4.7%*	450	1.6
4000	450	6.9	450	0.4	450	6.2	450	1.0	450	8.3	450	1.6
4500	450	0.2	450	0.5	450	321.7	450	1.1	450	284.8	450	1.8

Table 4: Computational results of the DOP solved as the SOP.

$\mathbf{T_{max}}$	$o_h = 4$				$o_h = 8$				$o_h = 12$			
	ILP		VNS-SOP		ILP		VNS-SOP		ILP		VNS-SOP	
	P	**T**	**P**	**T**	**P**	**T**	**P**	**T**	**P**	**T**	**P**	**T**
1500	115	2.97	115	0.21	120	8.18	120	0.39	120	20.62	120	0.62
2000	175	1.62	175	0.26	190	6.14	190	0.50	190	14.34	190	0.88
2300	190	1.96	190	0.31	200	9.14	200	0.61	200	17.76	200	1.00
2500	205	2.76	205	0.45	220	8.45	220	0.70	220	13.23	220	1.12
2700	215	3.14	215	0.39	230	5.67	230	0.70	230	17.25	230	1.18
3000	240	9.05	240	0.48	255	12.15	255	0.87	255	33.35	255	1.48
3200	265	4.11	265	0.50	280	12.06	280	0.90	290	51.76	290	1.31
3500	295	5.55	295	0.41	315	18.57	315	0.74	315	30.02	310	1.26
3800	330	9.17	330	0.47	345	23.15	345	1.10	345	46.18	345	1.91
4000	360	3.75	360	0.54	375	14.71	375	1.08	375	50.30	375	1.74
4500	415	3.54	415	0.59	430	14.29	430	0.93	440	21.31	440	1.85

Pěnička et al. (2017a); however, it was not addressed as the SOP studied here, nor solved using the ILP.

Orienteering Problem with Neighborhoods

The performance of the proposed methods is tested for the OPN dataset instances derived from the Tsiligirides (1984) Set 2 (see Section 4.1-OPN dataset). The results are shown in Table 3 for various budget limit $\mathrm{T_{max}}$ and number of neighborhood samples $o_n \in \{4, 8, 12\}$. The different values of o_n lead to instances with 78, 154, and 228 nodes, respectively. The maximal collected profit P and the average computational times T are reported for both VNS-SOP and the solutions found by the CPLEX solver when addressing the ILP formulation. For the instances with a medium budget and a large number of samples, the optimal solutions of the ILP formulation are not found within the given computational time, and the optimization gap is reported instead.

VNS-SOP finds the same optimal solutions in all cases where the optimal solution of the ILP formulation is found. In other cases, VNS-SOP achieves either the same results as, or better results than, the solutions found when addressing the ILP. The maximal computational time of VNS-SOP is 1.8 s, while the optimal solution of the ILP formulation is found within maximally 14 971.0 s. The selected numbers of samples o_n demonstrate how the solution quality improves when the OPN is better approximated using more samples.

Dubins Orienteering Problem

The DOP solved as a sampling-based SOP is evaluated on instances also derived from the Tsiligirides (1984) Set 2 (see Section 4.1-DOP dataset). For the DOP, the heading angles of the airplane-like Dubins vehicle are sampled into $o_h \in \{4, 8, 12\}$ values, creating datasets with 84, 168, and 252 nodes, respectively. Such SOP instances have starting and ending clusters with o_h nodes and the heading angles in these clusters have to be found during optimization. Table 4 reports on the achieved results of the solution methods for various budget limit $\mathrm{T_{max}}$ and number of heading samples o_h.

According to the presented results, the optimal solution is found by the CPLEX solver when addressing the ILP formulation for all the tested instances with the maximally required computational time 51.76 s. VNS-SOP finds solutions that are optimal in all instances with the exception of one case with $\mathrm{T_{max}} = 3500, o_h = 12$. The maximal computational time of VNS-SOP is 1.91 s. Similarly to the OPN, the DOP is better approximated using more samples o_h, as can be seen for $o_h = 4$ and $o_h = 12$. We can also observe that the optimal solution when addressing the ILP formulation is found much faster in the case of the asymmetric DOP than for the OPN. The branch-and-cut algorithm used by the CPLEX solver thus performs better for the DOP with a large difference in the lengths of the Dubins maneuvers between samples of the heading angle connecting the same clusters.

This is the author's accepted version of an article that has been published in European Journal of Operational Research. The final version of paper is available at https://doi.org/10.1016/j.ejor.2019.01.047

The results in both Table 3 and Table 4 show that the SOP can be successfully used for solving the sampled OPN and DOP. VNS-SOP can find high-quality solutions within 1.91 seconds for problems with up to 252 nodes. Furthermore, the solution of the ILP formulation, found with very low computational time in the case of the DOP, indicates that VNS-SOP can achieve the optimal solution of the sampled DOP and OPN for almost all evaluated instances.

5. CONCLUSIONS

In this paper, we introduce a Variable Neighborhood Search (VNS) metaheuristic and a novel Integer Linear Programming (ILP) formulation for the Set Orienteering Problem (SOP). The SOP is a generalization of the OP where customers are grouped in clusters, and the objective is to find a tour with a predefined starting cluster and ending cluster, a restricted budget, and such that the tour maximizes the profit collected from clusters with at least one visited customer. The VNS algorithm for the SOP (VNS-SOP) robustly provides high-quality solutions and improves the solution of several benchmark instances. The computational times of finding the SOP solution using both the novel ILP formulation and VNS-SOP are significantly lower than those of the existing approaches, especially in low to medium-size test instances. Furthermore, we show other variants of the Orienteering Problem that can be addressed as the SOP using a sampling-based approach. The Orienteering Problem with Neighborhoods, with profit collection within the neighborhood radius of each customer, and the Dubins Orienteering Problem for an airplane-like vehicle constrained by the minimum turning radius, can both be addressed as the studied SOP. The implementation of VNS-SOP and of the ILP formulation in the CPLEX solver are published as open-source software together with the dataset instances that have been used.

ACKNOWLEDGMENT

The work has been supported by the Czech Science Foundation under project No. 16-24206S and project No. 17-16900Y, and by OP VVV MEYS funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics". The support of CTU in Prague grant No. SGS17/187/13 is gratefully acknowledged. Finally, we thank the anonymous reviewers whose comments and suggestions helped to significantly improve this manuscript.

References

Angelelli, E., Archetti, C., & Vindigni, M. (2014). The clustered orienteering problem. *European Journal of Operational Research*, *238*, 404–414.

Archetti, C., Carrabs, F., & Cerulli, R. (2018). The set orienteering problem. *European Journal of Operational Research*, *267*, 264–272.

Dubins, L. E. (1957). On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics*, *79*, 497–516.

Faigl, J., Pěnička, R., & Best, G. (2016). Self-organizing map-based solution for the orienteering problem with neighborhoods. In *IEEE International Conference on Systems, Man, and Cybernetics* (pp. 1315–1321).

Feillet, D., Dejax, P., & Gendreau, M. (2005). Traveling salesman problems with profits. *Transportation Science*, *39*, 188–205.

Fischetti, M., Gonzlez, J. J. S., & Toth, P. (1997). A branch-and-cut algorithm for the symmetric generalized traveling salesman problem. *Operations Research*, *45*, 378–394.

Gunawan, A., Lau, H. C., & Vansteenwegen, P. (2016). Orienteering problem: A survey of recent variants, solution approaches and applications. *European Journal of Operational Research*, *255*, 315–332.

Laporte, G., Asef-Vaziri, A., & Sriskandarajah, C. (1996). Some applications of the generalized travelling salesman problem. *The Journal of the Operational Research Society*, *47*, 1461–1467.

Laporte, G., & Nobert, Y. (1983). Generalized travelling salesman problem through n sets of nodes: An integer programming approach. *INFOR: Information Systems and Operational Research*, *21*, 61–75.

Li, J., Andrew, L., Foh, C., Zukerman, M., & Chen, H.-H. (2009). Connectivity, coverage and placement in wireless sensor networks. *Sensors*, *9*, 7664–7693.

Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, *24*, 1097–1100.

Pěnička, R., Faigl, J., Váňa, P., & Saska, M. (2017a). Dubins orienteering problem. *IEEE Robotics and Automation Letters*, *2*, 1210–1217.

Pěnička, R., Faigl, J., Váňa, P., & Saska, M. (2017b). Dubins orienteering problem with neighborhoods. In *International Conference on Unmanned Aircraft Systems* (pp. 1555–1562).

Sevkli, Z., & Sevilgen, F. E. (2006). Variable neighborhood search for the orienteering problem. In *21th International Symposium on Computer and Information Sciences* (pp. 134–143).

Tsiligirides, T. (1984). Heuristic methods applied to orienteering. *The Journal of the Operational Research Society*, *35*, 797–809.

Vansteenwegen, P., Souffriau, W., & Oudheusden, D. V. (2011). The orienteering problem: A survey. *European Journal of Operational Research*, *209*, 1–10.

Yu, J., Schwager, M., & Rus, D. (2016). Correlated orienteering problem and its application to persistent monitoring tasks. *IEEE Transactions on Robotics*, *32*, 1106–1118.

Chapter 6

Unsupervised learning-based flexible framework for surveillance planning with aerial vehicles

The fourth core publication of this thesis is the manuscript named *Unsupervised learning-based flexible framework for surveillance planning with aerial vehicles* [4c] published in the Journal of Field Robotics.

[4c] J. Faigl, P. Váňa, **R. Pěnička**, and M. Saska, "Unsupervised learning-based flexible framework for surveillance planning with aerial vehicles," *Journal of Field Robotics*, vol. 36, no. 1, pp. 270–301, 2019

The article is motivated by our participation in the Mohamed Bin Zayed International Robotics Challenge 2017 competition, specifically by the Challenge 3. In this challenge, a team of three unmanned aerial vehicles is requested to search, pick up, and deliver colored objects placed in a given arena. One of the crucial tasks in the challenge was to localize the objects in the arena. This was addressed by a quick scan of the arena from a high altitude, which provided a large number of object detections with possible false positives. Therefore, an additional flight of all UAVs at a low altitude over the object detentions was planned to verify the detections and identify the rewards associated with the objects. The multi-robot planning over the object detections is the main topic of this core publication.

The problem of finding the shortest path over the object detections is formulated as a multi-vehicle variant of the DTSPN. In the targeted *minmax* variant of the m-DTSPN, the objective is to minimize the longest path for m robots while all target locations are visited in their neighborhoods by at least one of the robots modeled as Dubins vehicle. The main approach proposed in the article for the m-DTSPN is based on the unsupervised learning framework using the growing SOM. However, the flexible framework is further shown for a more complex Bézier curve model of the UAVs that can exploit the maximal velocity and acceleration of the vehicle rather than the Dubins vehicle. Finally, the Bézier curve model allows generalizing the approach to 3D.

Nevertheless, the author of this thesis contributed mainly to the development of the Variable Neighborhood Search method newly introduced for the m-DTSPN alongside the SOM-based approach. The VNS for the m-DTSPN uses a fast initialization procedure that greedily assigns targets to particular UAVs based on a competitive rule that minimizes deviation from average path length. The *shaking* and *local search* procedures of the VNS method have, similarly to the variants for the (D)OP(N) and SOP, a set of operators that are sequentially tried. In the case of the VNS for the *minmax* m-DTSPN, the operators of *local search* mainly focus on shortening the longest tour by moving its assigned targets to a different tour.

The computational results show that the SOM-based approach can find high-quality solutions in the shortest time. However, the proposed VNS approach has the shortest average solution lengths. In particular instances, the VNS-based solver outperforms the SOM solver, and even the proposed initialization method of the VNS achieves high-quality solutions.

The contribution of the author of this thesis on the publication is 20 % as the third author. The author contributed mainly by Section 5 about the VNS-based method for the m-DTSPN and by helping with the real experiments in Section 7.3.1.

Received: 15 October 2017 | Revised: 30 July 2018 | Accepted: 31 July 2018

DOI: 10.1002/rob.21823

REGULAR ARTICLE

WILEY

Unsupervised learning-based flexible framework for surveillance planning with aerial vehicles

Jan Faigl[1] | Petr Váňa[1] | Robert Pěnička[2] | Martin Saska[2]

[1]Department of Computer Science, Faculty of Electrical Engineering, Czech Technical University, Prague, Czech Republic

[2]Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University, Prague, Czech Republic

Correspondence
Jan Faigl, Department of Computer Science, Faculty of Electrical Engineering, Czech Technical University, Technická 2, 166 27 Prague, Czech Republic.
Email: faiglj@fel.cvut.cz

Funding information
Czech Science Foundation, Grant/Award Number: 16-24206S; Ministry of Education Youth and Sports, Czech Rupublic, OP VVV, Research Center for Informatics, Grant/Award Number: CZ.02.1.01/0.0/0.0/16_019/0000765

Abstract

The herein studied problem is motivated by practical needs of our participation in the Mohamed Bin Zayed International Robotics Challenge (MBZIRC) 2017 in which a team of unmanned aerial vehicles (UAVs) is requested to collect objects in the given area as quickly as possible and score according to the rewards associated with the objects. The mission time is limited, and the most time-consuming operation is the collection of the objects themselves. Therefore, we address the problem to quickly identify the most valuable objects as surveillance planning with curvature-constrained trajectories. The problem is formulated as a multivehicle variant of the Dubins traveling salesman problem with neighborhoods (DTSPN). Based on the evaluation of existing approaches to the DTSPN, we propose to use unsupervised learning to find satisfiable solutions with low computational requirements. Moreover, the flexibility of unsupervised learning allows considering trajectory parametrization that better fits the motion constraints of the utilized hexacopters that are not limited by the minimal turning radius as the Dubins vehicle. We propose to use Bézier curves to exploit the maximal vehicle velocity and acceleration limits. Besides, we further generalize the proposed approach to 3D surveillance planning. We report on evaluation results of the developed algorithms and experimental verification of the planned trajectories using the real UAVs utilized in our participation in MBZIRC 2017.

KEYWORDS

aerial robotics, computing architectures, planning

1 | INTRODUCTION

The surveillance planning problem studied in this paper is motivated by practical needs of our participation in the *Mohamed Bin Zayed International Robotics Challenge* (MBZIRC) 2017 (MBZIRC, 2017; Saska, 2017). In particular, in our effort towards the Challenge 3, where a team of *unmanned aerial vehicles* (UAVs) is requested to search and collect objects of interest located in a specified arena. Placement of the objects is not known a priori, and therefore, a quick scan of the whole area is performed at a high altitude to provide a rough estimation of the possible object locations with particular preference of false positives rather than false negatives. Then, a group of up to three UAVs is requested to verify the objects and identify the reward associated with them to prefer collecting the most rewarding objects for achieving a high total score (see Figure 1 with snapshots from our preparation experiments). The particular problem addressed in this paper is the trajectory planning to identify the objects of interest that is considered as the surveillance planning with known target locations provided from the first overview scans of the area.

The time for the whole mission is limited, and the most time-consuming part is the pickup and delivery of the objects; hence, the UAVs have to quickly visit the expected locations of the objects and confirm the object location and its reward or reject false positive

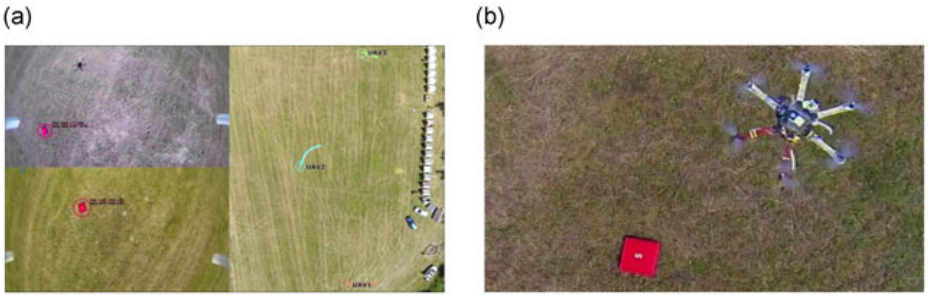


FIGURE 1 A snapshot of three UAVs following the planned trajectories in a validation of the objects of interest (left) and detail of the used object of interest (right) in the preparation phase towards the Challenge 3. UAV: unmanned aerial vehicle [Color figure can be viewed at wileyonlinelibrary.com]

estimates. Hence, it is desirable to spend as little time as possible in this verification part of the mission. Moreover, regarding the size of the arena, which is in tens of meters, and velocity of the UAVs that fly up to 5 ms^{-1}, it is preferable to do not spend too much time by planning the trajectories for objects identification as the UAV can travel a significant distance in any additional second spent in planning. Therefore, it has been requested to develop a surveillance planning algorithm with low computational requirements while still be able to provide solutions of satisfiable quality. Thus, our initial intention was to provide a cost-efficient solution in less than 1 s using a single core of a conventional computer with a central processing unit (CPU) of the iCore7 class running at the frequency around 3.4 GHz, that is, computational resources available at our UAVs (Spurný et al., 2018).

Surveillance planning as finding a cost-efficient trajectory to visit a set of locations can be addressed as a solution of the *traveling salesman problem* (TSP) which is a well-studied problem of combinatorial optimization, for which several computationally efficient heuristic algorithms have been developed (Applegate, Bixby, Chvátal, & Cook, 2007; Helsgaun, 2000). Regarding trajectory planning for a team of vehicles, such that the total time required to validate all possible object locations is minimized, it is necessary to consider the m-TSP approaches that directly minimize the longest tour length, i.e., the *minmax* variant of the m-TSP (Bektas, 2006). Notice, the problem where the sum of the lengths (*minsum*) of all tours is minimized can be addressed by a transformation of the m-TSP to the single vehicle TSP using (Bellmore & Hong, 1974); however, such solutions are of poor quality as they can contain degenerative solutions with zero tour lengths for particular vehicles. Therefore, it is necessary to address the *minmax* m-TSP directly.

Moreover, when planning trajectories for UAVs, it is suitable to provide smooth trajectories even for our hexacopter UAVs utilized in MBZIRC 2017 because the low-level trajectory following controller can more precisely navigate the vehicle along the planned path (Báča, Loianno, & Saska, 2016). An example of the trajectory following performance is shown in Figure 2. Therefore, a curvature-constrained path is desirable to enable fast motion with the maximal forward velocity and precise trajectory following rather than paths with sharp turns that can be found as a solution of the regular Euclidean TSP (ETSP).

A suitable kinematic model widely used for the UAVs is the Dubins vehicle for which the curvature-constrained TSP becomes the Dubins TSP (DTSP; Savla, Frazzoli, & Bullo, 2005) and we further call the multivehicle problem for m vehicles as the m-DTSP. In addition, it is sufficient to visit proximity of the expected object location to capture the object by a camera sensor with a particular field of view, and thus, it is sufficient to reach the object location at the specific sensing range δ to reliably detect the object of interest. Hence, the problem can be formulated as the DTSP with Neighborhoods (DTSPN; Isaacs, Klein, & Hespanha, 2011; Obermeyer, 2009; Oberlin, Rathinam, & Darbha, 2010) and its multivehicle variant is denoted the m-DTSPN (Macharet, Neto, da Camara Neto, & Campos, 2013).

For the Dubins vehicle model with the minimal turning radius ρ, the forward velocity is assumed to be constant, and thus the required time to complete the surveillance mission is proportional to the longest tour. Besides, we can consider smaller ρ which requires lower velocity in turning parts of the path, but the vehicle can accelerate and then decelerate on straight line segments to achieve the required velocity in turns. The vehicle can eventually finish the mission sooner than for a high but a constant forward

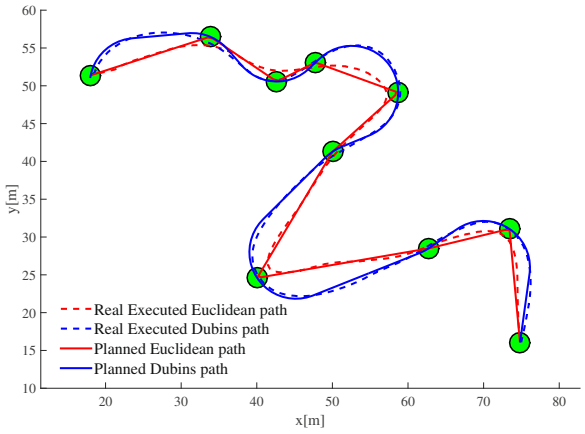


FIGURE 2 An example of the planned paths and their real execution by the used model predictive control-based controller for trajectory following (Báča et al., 2016) [Color figure can be viewed at wileyonlinelibrary.com]

velocity and longer ρ. In general, smooth trajectories can be parametrized, for example, by B-splines (Neubauer & Müller, 2015) or Bézier curves (Yang & Sukkarieh, 2010), and the trajectory curvature can be then utilized with the maximal vehicle velocity and acceleration to determine velocity profile along the trajectory from which the *travel time estimation* (TTE) can be computed. Thus, the herein addressed problem is to determine m trajectories to visit the given set of n object locations such that the longest time to travel the particular trajectory is minimized, and it is allowed to visit the location in δ distance, that is, the problem is formulated as a variant of the m-DTSP for $\delta = 0$ and as the m-DTSPN for $\delta > 0$.

1.1 | Focus of the proposed approaches and contributions

The motivation and practical needs of the surveillance planning deployed in the robotic competition steered our effort towards a suitable solution of the m-DTSPN instances arising from MBZIRC 2017. Therefore, we focused on the development of surveillance planning framework that is capable of providing a feasible solution for a typical scenario of MBZIRC 2017 with up to three vehicles and around 20 object locations relatively sparsely placed in the arena around 80 m × 60 m large. In addition, the required computational time of the planning should be significantly shorter than the time to travel across the arena, and at best, it should be around 1 s, and it should not exceed 60 s. Thus, heuristic algorithms providing solutions of satisfiable quality are preferred than a computationally demanding optimal solution of the DTSP, which is known to be NP-hard (Le Ny, Feron, & Frazzoli, 2012).

Due to these requirements, the studied and proposed approaches have been evaluated in the scenario called *mbzirc22* with 22 targets with additional up to three starting locations, one for each of three UAVs, to obtain a realistic estimation of the real performance in MBZIRC 2017 (Figure 3). Although efficient solutions for such a relatively small problem may not scale well with the number of vehicles or the number of targets, the practical deployment, and real experimental verification provide realistic validation of the real and time-critical deployment as it is the participation in a robotics competition.

FIGURE 3 Motivational scenario called *mbzirc22* on top of the test field site (about 80 m × 60 m large) used for real experiments [Color figure can be viewed at wileyonlinelibrary.com]

Regarding the particular approaches to the m-DTSP(N), we consider a purely combinatorial optimization approaches already proposed in the literature to address the m-DTSPN and *minmax* variant of the m-TSP. We also consider our previous effort towards surveillance planning with UAVs based on unsupervised learning of the *self-organizing map* (SOM) first deployed in a solution of the DTSP in (Faigl & Váňa, 2016) and later generalized for the m-DTSPN in (Faigl & Váňa, 2017). Following the sampling-based approaches of the continuous optimization problem of Dubins planning (Oberlin et al., 2010; Obermeyer, Oberlin & Darbha, 2012), the *variable neighborhood search* (VNS) metaheuristic (Soylu, 2015) is also considered for a direct solution of the *minmax* m-DTSP and its generalization to the m-DTSPN. Besides, an evolutionary-based memetic algorithm (Zhang, Chen, Xin, & Peng, 2014) has been selected for a comparison with the proposed solutions. The promising results and very low computational requirements of the SOM-based solution motivate us to further generalize the unsupervised learning for 3D surveillance planning using Bézier curves (Jolly, Sreerama Kumar, & Vijayakumar, 2009; Yang & Sukkarieh, 2010) and computation of the velocity profile along the planned trajectory using the vehicle velocity and acceleration limits.

Even though the presented work is built on the previous approaches published in the literature, that is, the VNS for the m-TSP (Soylu, 2015) and SOM-based unsupervised learning for the m-DTSPN (Faigl & Váňa, 2017), they have been further developed to address the m-DTSPN by the VNS-based approach and the SOM-based approach has been generalized to 3D surveillance planning. Therefore, we consider the main contributions of the paper with respect to the existing approaches as follows:

- Deployment of the VNS-based m-TSP solver in the m-DTSPN.
- Fast and efficient initialization for the VNS-based optimization in the m-DTSPN.
- Comprehensive evaluation of the proposed VNS-based solver and the existing memetic and SOM-based approaches in the *mbzirc22* scenarios of the m-DTSPN with varying number of vehicles.
- Experimental verification of the found trajectories using real UAVs utilized in MBZIRC 2017.
- Generalization of the SOM-based solver to 3D surveillance planning.
- Verification of the feasibility of the found 3D trajectories using real aerial vehicles.
- Since the developed unsupervised learning-based solver allows a straightforward extension from the Dubins vehicle model to a Bézier curve or any similar model (e.g., Dubins-Helix model; Wang, Wang, Tan, Zhou, & Wei, 2015b), while the main principles are the same, we consider the proposed planner as a suitable flexible framework for surveillance planning with aerial vehicles.
- Unsupervised learning framework for surveillance planning addressing the m-DTSPN but also the multivehicle planning problem where it is requested to quickly find surveillance trajectories

considering the maximal vehicle velocities and acceleration limits that better fit the real motion capabilities of multirotor UAVs than Dubins vehicle describing curvature-constrained trajectories suitable for fixed-wing vehicles.

The paper is organized as follows. An overview of the related work is presented in the next section. A formal definition of the addressed problems with a brief overview of the Dubins vehicle model is presented in Section 3. Necessary background on the related *Dubins touring problem* (DTP; Faigl, Váňa, Saska, Báča, & Spurný, 2017) and 3D smooth trajectory parametrization based on Bézier curve is described in Section 4. The proposed VNS-based *m*-DTSPN solver is introduced in Section 5 and the generalized SOM-based planner to the 3D surveillance planning is presented in Section 6. Reports on empirical evaluation and experimental deployment are presented in Section 7. Conclusion is dedicated to Section 8.

2 | RELATED WORK

Surveillance planning for an aerial vehicle is usually closely related to the curvature-constrained path planning for which the fundamental work is Dubins (1957) where the problem of the optimal planning for a vehicle with the minimal turning radius ρ is studied. In 1957, Dubins showed that the optimal path connecting two states $q_i, q_j \in SE(2)$ (representing the vehicle configurations as two points in the special Euclidean group $SE(2)$) is one of six possible maneuvers that consist of a straight line segment and a part of a circle with the radius ρ. Although a closed-form expression of the optimal path for the Dubins vehicle between two states exists, it is not sufficient to directly solve surveillance planning where a vehicle is requested to collect information from the given set of target locations. It is due to the initially unknown optimal sequence of visits to the targets, and also the particular headings at the target locations are not known. Therefore, it is necessary to determine both the sequence and the headings, which can be formulated as the DTSP.

The DTSP can be considered as an extension of the regular TSP for the Dubins vehicle, and thus the path connecting the particular locations are the Dubins maneuvers respecting the minimal turning radius ρ. Similarly to the regular TSP, also the DTSP stands to determine the optimal sequence of visits to the targets, which is a discrete combinatorial problem. However, the DTSP also includes a continuous optimization part in finding the optimal heading of the vehicle at each target location. Each particular heading value can be selected from the interval $[0, 2\pi)$ and every change of a single heading may significantly change the Dubins tour connecting the locations. Therefore, the DTSP can be considered as a more challenging problem than a discrete optimization of the regular TSP, although both problems are NP-hard (Le Ny et al., 2012) as the DTSP becomes the regular ETSP for $\rho = 0$.

Moreover, in the DTSPN, it is also required to determine the most suitable waypoint locations from which information about the targets is collected such that the waypoint is at a distance equal or shorter than the given sensing range δ (Isaacs & Hespanha, 2013). Hence, the DTSPN contains two continuous optimization parts in addition to the determination of the sequence of visits to the targets. The first continuous part is the determination of the optimal heading at the waypoints and the second is the determination of the waypoint locations themselves. Therefore, finding optimal solutions of the DTSP or the DTSPN is computationally challenging and approximation algorithms (Ny, Feron, & Frazzoli, 2012; Oberlin et al., 2010; Yu, 2015), heuristics (Isaiah & Shima, 2015; Savla et al., 2005; Váňa & Faigl, 2015), and evolutionary (Yu & Hung, 2012) approaches have been proposed.

The existing approaches to the DTSP and DTSPN can be categorized into four main classes. The first class represents decoupled approaches where the sequence of the visits to the targets is determined independently on the determination of the headings. The second class is sampling-based methods where a finite discrete set of possible heading values and/or waypoint locations are sampled, and the problem is then transformed into a discrete optimization problem, for example, the ATSP, that can be solved by existing optimal solvers such as Concorde (Applegate, Bixby, Chvátal, & Cook, 2003) or heuristic algorithms such as the Lin-Kernighan-Helsgaun (LKH) algorithm (Helsgaun, 2000). The third class of approaches is evolutionary methods that can provide high-quality solutions but are usually computationally very demanding. Finally, the fourth class is the recently proposed unsupervised learning which combines a solution of the sequencing part of the problem with the online sampling of the suitable heading values (Faigl & Váňa, 2016) and for the DTSPN also the waypoint locations (Faigl & Váňa, 2017). Selected approaches of the particular classes are briefly described in the rest of this section to support our selection of the considered methods in our effort towards a suitable solution for a practical deployment motivated by MBZIRC 2017.

One of the simplest approaches, that is also computationally very efficient, is the decoupled approach called the *alternating algorithm* (AA) proposed by (Savla et al., 2005). The sequence of visits to the targets is determined by a solution of the ETSP without considering the curvature-constrained path. After that, headings at the waypoints are established in such a way that even edges are connected by straight line segments which prescribe all the headings, and thus odd edges are connected by the optimal Dubins maneuvers that can be computed analytically (Dubins, 1957). The AA has been improved by a randomized adaptive search in (Macharet, Neto, da Camara Neto, & Campos, 2011) and by considering a distance between two consecutive waypoints in the sequence (Macharet & Campos, 2014). Because only two consecutive waypoints in the sequence are considered in these approaches, determination of the headings is computationally very efficient, and for n targets, the computational complexity can be bounded by $O(n)$.

Following the idea of the AA, a receding horizon technique has been utilized in the look-ahead approach proposed in Ma & Castanon (2006), where the heading at the next waypoint in the sequence is determined according to the three waypoint locations and heading at the previous waypoint. The reported results are better than for the AA which is also reported in Isaiah and Shima

(2015), where a combination of the k-look-ahead technique is accompanied by a local improvement based on the 2-Opt heuristic (Croes, 1958). However, the authors do not report on the required computational times.

Another promising decoupled approach called the *local iterative optimization* (LIO) algorithm has been proposed in Váňa and Faigl (2015) to address the computationally challenging DTSPN. In particular, the proposed approach is focused on problem instances where a distance of the waypoints in the sequence is longer than 4ρ, that is, the so-called D_4 instances of the DTSP(N). The initial sequence of the visits to the targets is determined as a solution of the ETSP for target locations in their respective neighborhoods. Then, the problem is addressed as a continuous optimization of two variables for each target. The first variable is the waypoint heading, and the second variable is for the waypoint location which is considered as a single variable denoting its position on the boundary of the target's neighborhood. Both variables are then iteratively optimized until the solution is not improving. According to the reported results, the LIO algorithm provides almost about 10% better solutions than the AA while the computational requirements are still around tens or hundreds of milliseconds using a single core of a conventional CPU. LIO has been proposed for the D_4 instances, but it can also be utilized for solving any instance of the DTSP and DTSPN; however, the quality of found solutions depends on the sequence determined as the ETSP, which can be inadequate for dense and mutually close target locations.

The problem of determining the optimal headings at the waypoints for a given sequence of visits to the targets is called the DTP in Faigl et al. (2017), and it has been addressed by several approaches. An optimal solution of the D_4 instances of the DTP has been proposed in Goaoc, Kim, and Lazard (2013). The solution is based on solving a family of n-dimensional convex optimization subproblems, where n is the number of waypoints in the sequence. The number of subproblems can be bounded by 2^{2n-2}, which regarding the computational complexity of the whole algorithm is relatively high in comparison with simple heuristics such as the AA (Savla et al., 2005) or LIO (Váňa & Faigl, 2015). Notice, a solution of the DTSP with a given sequence of visits to the targets can be easily found as a solution of the DTP for a discrete set of possible heading values at each waypoint (see Figure 4a and a description of the forward search procedure in Section 4.1).

A very important result on the tight lower bound of the DTP has been proposed in Manyam and Rathinam (2015) which has been further evaluated in (Manyam, Rathinam, and Casbeer (2016), but unfortunately without reporting the computational requirements. The computation of the tight lower bound is based on the solution of the so-called the *Dubins interval problem* (DIP) introduced in Manyam, Rathinam, Casbeer and Garcia (2015). DIP is a variant of the Dubins planning between two waypoints, that is, locations with the prescribed headings. In DIP, the heading at the waypoint is not a single value but an interval. Thus, for the interval of the full range 2π, the solution of DIP is a straight line segment connecting the locations. The tight lower bound (Manyam & Rathinam, 2015) has been utilized to guide sampling of the possible heading intervals and

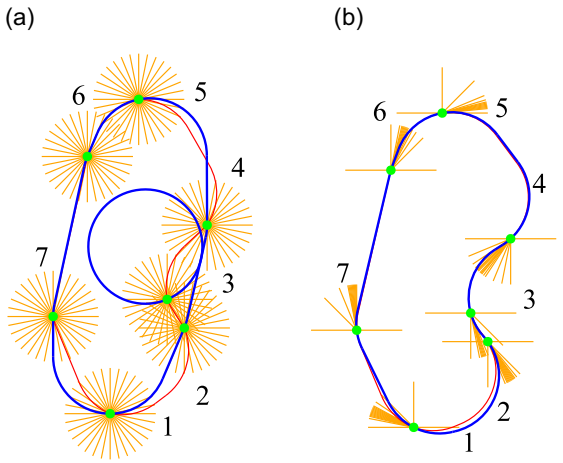


FIGURE 4 A solution of the Dubins traveling salesman problem for a given sequence of the targets (the green disks) with the total number of samples N, final path length $\mathcal{L}$, and lower bound $\mathcal{L}_U$. The found solution is the blue curve, and the red curve is its lower bound determined as a solution of the *Dubins interval problem* with the cost $\mathcal{L}_U$ (Manyam & Rathinam, 2015). The uniform sampling utilizes 32 heading values per each target. The required computational time is denoted t. (a) Uniform sampling— $N = 224$, $\mathcal{L} = 19.8$, $\mathcal{L}_U = 13.8$, $t = 128$ ms, (b) Guided sampling (Faigl et al., 2017)–$N = 128$, $\mathcal{L} = 14.4$, $\mathcal{L}_U = 14.2$, $t = 76$ ms [Color figure can be viewed at wileyonlinelibrary.com]

the heading values themselves in Faigl et al., (2017), where the authors show improved results over a uniform sampling of the heading (see Figure 4 for an example of the DTSP solution based on the DTP and DIP).

Transformation (or also sampling-based) methods represent the second class of the approaches to the DTSP(N). Similarly to the aforementioned discretization of the headings in the DTP, these methods consider a finite set of discrete heading values at each waypoint location or a set of possible locations in the case of the DTSPN. Then, the optimal Dubins maneuvers between all possible pairs of the waypoint locations are computed to build a complete graph representing the original problem, which can be solved by combinatorial graph-based solvers.

One of the first sampling-based and resolution complete approaches to the DTSPN has been proposed in Obermeyer, Oberlin, and Darbha (2010). In this approach, the DTSPN is transformed into the generalized TSP (GTSP) where the targets with their neighborhoods are represented by mutually exclusive finite sets of nodes. The GTSP is then transformed into the asymmetric TSP (ATSP) because the optimal Dubins maneuvers between two states depend on the path direction. Such a transformed problem is solved by the LKH algorithm (Helsgaun, 2000). Even though the LKH algorithm is one of the most powerful heuristics for the TSP, due to the samples and transformation, the final problem has many nodes. The reported computational times for problems with 20 targets and 1,500 random samples are several hundreds of seconds (Obermeyer et al., 2010), which is reported to be faster than the genetic algorithm for the

DTSPN proposed by the same authors in Obermeyer (2009), but still far from our needs and expectations.

A comparison of the DTSPN approaches is provided in Macharet, Neto, da Camara Neto, and Campos (2012) where significant improvement of the solution quality is reported for evolutionary techniques. A memetic algorithm for the DTSPN with the disk-shaped neighborhoods and relaxed terminal heading is proposed in Zhang et al., (2014). The superior solution quality is reported for the memetic algorithm in the DTSPN instances with 10 targets. The reported computational times are 8.3 s for 10 targets and 45.5 s for problems with 17 targets. A genetic algorithm for the DTSPN with polygonal goals has been proposed in Obermeyer (2009) but the real computational requirements are not reported. However, the same authors report that their sampling-based approach proposed in Obermeyer et al. (2010) requiring hundreds of seconds is faster than the genetic algorithm (Obermeyer, 2009).

The recently proposed unsupervised learning method for the DTSP (Faigl & Váňa, 2016) is based on an evolution of the growing SOM for the TSP (Faigl, 2018). The input layer of the two-layered neural network servers for presenting the input signals which are the target locations. The neurons' weights represent locations in the input space, and the output layer is an array of nodes representing the waypoints. Since the output layer is one-dimensional and the nodes are organized in an array, it forms a ring of neurons that directly represents a TSP tour. In Faigl & Váňa (2016), a possible heading value at the target is determined in the selection of the best matching neuron to the target location presented to the network. Besides, additional heading values are associated with the winner neuron which is adapted towards the presented target, that is, its weights are moved towards the target location in the input space. The adaptation of the network is performed in learning epochs in which all targets are presented to the network. The weight of the adaptation is decreased after each epoch according to a cooling schedule, and the network is stabilized in hundreds of epochs. However, a solution of the DTSP is determined as a solution of the DTP represented by the winner neurons of the current epoch, where the ring of nodes prescribes the sequence of visits to the targets and the particular headings are the associated headings to the neurons. Thus, a solution is available after each learning epoch, and the final solution is found as the best-found solution among all the learning epochs. The reported results are better than the solutions provided by the memetic algorithm (Zhang et al., 2014) with the computational time restricted to 100 s while the SOM needs less than 30 s for problems with up to 100 targets.

The SOM-based algorithm (Faigl & Váňa, 2016) has been significantly improved in (Faigl & Váňa, 2017), where the reported required computational time for scenarios (motivated by MBZIRC 2017) with 22 targets is found in less than 600 ms, while the solutions are better than those provided by the memetic algorithm (Zhang et al., 2014) with the computational time restricted to 10 s. Moreover, the SOM-based approach has been generalized for the DTSPN, where the particular waypoint locations are determined during the winner selection together with the expected heading at the waypoint. In addition, the *m*-DTSPN is addressed by creating an individual neural network for each vehicle, and during the winner selection, neurons from the network which represents a shorter tour are preferred to address the *minmax* variant of the *m*-TSP (Somhom, Modares, & Enkawa, 1999).

Regarding approaches for the *m*-DTSPN, they are similar to the *m*-TSP in many ways (Bektas, 2006; Oberlin et al., 2010), especially the transformation/sampling-based solvers, but only a few approaches directly address the challenges of the *minmax* variant of the *m*-DTSPN. One of them is the memetic algorithm (Zhang et al., 2014), which has been compared with the additional direct approach based on SOM in Faigl & Váňa (2017). Another evolutionary based approach to the *minmax* variant of the *m*-DTSPN has been proposed in Macharet et al. (2013), but the authors do not report on the computational requirements, which also hold for the improved version presented in Macharet, Monteiro, Mateus, & Campos (2016).

Having a transformed problem with a graph representation, graph-based *m*-TSP approaches may be considered. The *minmax* variant of the *m*-TSP has been addressed by França, Gendreau, Laporte, and Müller (1995) where exact algorithms are proposed. In Kulich, Faigl, Kléma, and Kubalík (2004), the authors compare genetic algorithm, ant colony optimization, and SOM-based solver in *m*-TSP scenarios arising from rescue missions, where the superior results are provided by SOM. In addition to the soft-computing techniques, a general metaheuristic called the VNS proposed by (Hansen & Mladenović, 2001) has been applied to the *minmax m*-TSP in (Soylu, 2015).

Regarding the presented overview of the existing methods for the DTSPN and more specifically to the *m*-DTSPN. We consider the memetic algorithm (Zhang et al., 2014) and SOM-based approach (Faigl & Váňa, 2017) as the most promising because the memetic algorithm is capable of providing a high-quality solution, and thus it may represent a suitable reference approach. On the other hand, the SOM-based approach has the computational requirements lower than the desired 1 s while it also provides better solutions than the simple heuristics AA and LIO (Faigl & Váňa, 2016). Besides, solutions of the *m*-DTSPN are reported for both the memetic and SOM-based algorithms and both approaches are any-time as they provide the first solution very quickly, which is also desirable property for a practical deployment under real-time constraints.

In addition, we also included sampling-based approach in our evaluation to cover purely combinatorial optimization approaches which work on some finite discrete set of possible heading values and waypoint locations. In this case, we consider the VNS method (Soylu, 2015) as a particularly interesting method. First, it directly addresses the *minmax m*-TSP, and it improves the initial solution if more computational time is available. Besides, the VNS metaheuristic has been recently successfully deployed in a solution of the closely related problem of the surveillance planning called the *Dubins orienteering problem* (DOP; Pěnička, Faigl, Váňa, & Saska, 2017a) which has been further generalized to the DOP with neighborhoods in Pěnička, Faigl, Váňa, and Saska (2017b). Therefore, we consider VNS as a promising method for the sampling-based approach to the

herein addressed m-DTSPN. However, an initial solution of the m-DTSPN is needed for the VNS-based optimization which is addressed by a newly proposed procedure described in Section 5.

In addition to the Dubins vehicle model, which is a suitable model for rotary vehicles because it provides smooth trajectories with a constant speed, we are interested also in other types of trajectory parametrization because the motion of the rotary UAV is limited mainly by the maximum speed and acceleration, and the minimal turning radius is not defined. Various types of curves such as splines (Lepetič, Klančar, Škrjanc, Matko, & Potočnik, 2003), polynomial functions (Papadopoulos, Papadimitriou, & Poulakakis, 2005), and Bézier curves (Jolly et al., 2009) can be utilized for continuous and smooth path generation (Wang, Wang, & Tan, 2015a) for which the final trajectory with the velocity profile is computed according to the maximum possible velocity and acceleration of the vehicle. Moreover, we are also interested in the generalization of the surveillance planning with curvature-constrained paths from the 2D environment representation to 3D. An extension of the Dubins vehicle for the 3D is possible using Dubins-Helix method (Wang et al., 2015b) or using the so-called *Dubins Airplane model* proposed in Chitsaz and LaValle (2007, December) to address the bounded curvature and also limited pitch angle of real UAVs, especially fixed-wing vehicles. The Dubins Airplane model has been used for solving the 3D-DTSPN (Váňa, Sláma, & Faigl, 2018) with fixed-wing vehicle. However, hexacopters are used in our motivational problem, and therefore, we consider Bézier curves (Yang & Sukkarieh, 2010) that can describe trajectories of arbitrary curvature and are specified only by four control points. Thus, trajectory parametrization based on Bézier curves is selected as a suitable generalization of the proposed surveillance planning framework to directly find smooth trajectories for a team of UAVs in 2D but also in 3D scenarios.

The SOM-based approach (Faigl & Váňa, 2017) has been selected as a suitable optimization framework for the generalized surveillance planning with Bézier curves because of two main reasons. The first reason is related to the expected increased computational requirements related to the optimization of Bézier curves that is more demanding than the analytical solution of Dubins maneuvers, and regarding the reported results, SOM is computationally efficient. Besides, the unsupervised learning principles used in SOM are flexible to relatively straightforwardly utilize different parametrization of the curves. Therefore, the proposed unsupervised learning based 3D surveillance planning framework for the m-DTSPN is presented in Section 6.

A summary and evolution of the existing approaches together with the herein proposed methods for solving variants of the DTSP is presented in Table 1 with an indication of their particular properties and capabilities. Besides, we further distinguish if the approach performs continuous trajectory optimization, which may further improve the solution. The transformation methods perform sampling, and thus, they transform the problem to the combinatorial optimization. On the other hand, the decoupled approaches first determine the sequence of visits and then generate the requested trajectories where the recent approaches employ continuous optimization of the headings and possibly also the locations of the waypoints. The unsupervised learning is similar to the decoupled approaches in the trajectory optimization. However, the continuous trajectory optimization is performed during the solution of the sequencing part that is the main difference to decoupled and transformation approaches and makes it similar to evolutionary methods, but the convergence of the learning is much faster than finding satisfiable solutions by, for example, memetic algorithms.

3 | PROBLEM STATEMENT

The studied surveillance planning problem is motivated by the MBZIRC 2017 competition where it is needed to identify possible objects of interest as quickly as possible by three aerial vehicles. The problem is considered as surveillance planning where a team of m vehicles is requested to take a camera snapshot of the objects using nonzero sensing range δ to save the travel cost. Moreover, due to the used *model predictive control* (MPC) trajectory following (Báča et al., 2016), the surveillance trajectories have to fit the vehicle motion constraints to allowing fast and precise motion of the vehicle along the trajectory. Thus, the problem is to determine a sequence of visits to the object locations for each vehicle together with the corresponding trajectories connecting the waypoints from which objects are captured such that all the objects are identified as quickly as possible, and the vehicles return to their initial locations. The expected computational requirements for the surveillance planning and the specific setup of the MBZIRC 2017 deployment allow to relax the collision avoidance in the planning part, and it is addressed by the reactive collision avoidance implemented in the used MPC-based trajectory following controller (Báča, Hert, Loianno, Saska, & Kumar, 2018; Spurný et al., 2018). Therefore, an explicit finding of collision-free trajectories is not considered in the following problem formulations.

First, the problem is formulated as the m-DTSPN in which m curvature-constrained paths (one path for each vehicle) for the Dubins vehicle with the minimal turning radius ρ are found such that each of the given n target locations is visited by at least one of the planned path in the distance not exceeding the sensing range δ and the length of the longest path is minimized. In addition to ρ, the utilized Dubins vehicle model (Dubins, 1957) assumes the constant forward velocity v and the state q of the Dubins vehicle is described as a triplet $q = (x, y, \theta)$, where $p = (x, y)$ is the vehicle position in the plane $p \in \mathbb{R}^2$ and θ is the vehicle heading at p and $\theta \in \mathbb{S}^1$, that is, $q \in SE(2)$. The motion of the vehicle is described as

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

where u is the control input.

The team of UAVs consists of m identical vehicles with the same ρ allowing the constant, maximal, and safe forward velocity while the error of the trajectory following is acceptable to capture the object of

TABLE 1 Summary of the existing methods for surveillance planning with aerial vehicles

Approach	Method type	Sampling	Trajectory optimization	Any-time	Neighborhoods	Multiple vehicles	3D trajectory	Any-curvature	Computational requirements[a]
Savla et al. (2005)	Decoupled								Low
Ma and Castanon (2006)	Decoupled								Low
Obermeyer (2009)	Evolutionary			√	√				Moderate
Obermeyer et al. (2010)	Transformation	√			√				Moderate
Oberlin et al. (2010)	Transformation	√			√	√			High
Macharet et al. (2011)	Decoupled		√						Low
Macharet et al. (2012)	Evolutionary			√	√				High
Le Ny et al. (2012)	Decoupled								Low
Yu and Hung (2012)	Evolutionary			√					High
Macharet et al. (2013)	Evolutionary			√	√	√			High
Zhang et al. (2014)	Evolutionary		√	√	√	√			High
Macharet and Campos (2014)	Decoupled		√						Low
Váňa and Faigl (2015)	Decoupled		√		√				Low
Isaiah and Shima (2015)	Decoupled								Low
Manyam et al. (2015)	Transformation	√			√				Moderate
Macharet et al. (2016)	Transformation	√			√	√			Moderate
Faigl and Váňa (2016)	Unsupervised learning			√					Low
Faigl and Váňa (2017)	Unsupervised learning			√	√				Low
Váňa et al. (2018)	Decoupled		√		√		√		Low
Proposed methods									
VNS – Section 5, 2018	Transformation	√		√	√	√			Low
SOM Dubins – Section 6.1, 2018	Unsupervised learning		√	√	√	√			Low
SOM Bézier – Section 6.2, 2018	Unsupervised learning		√		√	√	√	√	Moderate

Note. [a]Computational requirements are considered *low* if a satisfiable solution (for $n \approx 20$, i.e., *mbzirc22* scenario; see Section 7) is found in less than 1 s and *moderate* in less than 60 s using conventional computational resources; otherwise, the requirements are considered *high*.

interest at the target location from the determined waypoint location within the δ sensing range from the target location. In fact, the real field of view of the utilized camera is wider than δ used for planning such that the used MPC-based controller (Báča et al., 2016) follows the trajectory with the error less than the difference of the real field of view and δ, and thus, it is assured that the object of interest can be identified from the snapshot taken at the particular waypoint location.

Each vehicle (denoted r) starts at its individual initial location $p_d^r \in \mathbb{R}^2$ (further denoted as depot) and the requested path for the rth vehicle terminates at the same location p_d^r, that is, we are searching for m closed trajectories. The trajectories consist of a sequence of Dubins maneuvers connecting the determined waypoints. Thus, two consecutive waypoints in the sequence q_i and q_{i+1} both from $SE(2)$ are connected by one of the six Dubins maneuvers respecting the kinematic constraints of the Dubins vehicle (1).

In the DTSPN with a single vehicle, the goal is to find the shortest trajectory to take a snapshot of all n objects of interest $\mathcal{O} = \{o_1, ..., o_n\}$. For simplicity and readability, we consider o_i be the target location of the object i, that is, $o_i \in \mathbb{R}^2$. Since it is allowed to collect information about o_i within δ distance, we need to determine for each o_i a waypoint location p_i such that $|(p_i, o_i)| \leq \delta$. Besides, for each waypoint location p_i, we need to determine the heading θ_i and for the all waypoints, we search for a sequence $\Sigma = (\sigma_1, ..., \sigma_n)$ of the waypoints $q_i = (p_i, \theta_i)$ such that the sum of the lengths of the Dubins maneuvers connecting the waypoints in the sequence Σ is minimal.

Problem 1 (DTSPN)

$$\begin{aligned} &\text{minimize}_{P,\Theta,\Sigma} && \mathcal{L}(Q, O) = \textstyle\sum_{i=1}^{n} \mathcal{L}(q_{\sigma_{i-1}}, q_{\sigma_i}) + \mathcal{L}(q_{\sigma_n}, q_{\sigma_0}) \\ &\text{subject to} && Q = (q_{\sigma_1}, ..., q_{\sigma_n}),\ q_{\sigma_i} = (p_{\sigma_i}, \theta_{\sigma_i}),\ q_{\sigma_i} \in SE(2), \\ & && P = (p_{\sigma_1}, ..., p_{\sigma_n}),\ p_{\sigma_i} \in \mathbb{R}^2 \text{ and } |(p_{\sigma_i}, o_i)| \leq \delta \text{ for } o_i \in O \\ & && \Theta = (\theta_{\sigma_1}, ..., \theta_{\sigma_n}),\ 0 \leq \theta_{\sigma_i} < 2\pi, \\ & && \Sigma = (\sigma_1, ..., \sigma_n),\ 1 \leq \sigma_i \leq n \text{ and } \sigma_i \neq \sigma_j \text{ for } i \neq j, \\ & && q_{\sigma_0} \in SE(2), \textstyle\sum_{0=0} \text{ and } \mathcal{P}(q_0) = p_d \text{ is the vehicle depot,} \end{aligned} \tag{2}$$

where $\mathcal{L}(q_i, q_j)$ is the length of the shortest Dubins maneuver between q_i and q_j computed analytically according to Dubins (1957) and $\mathcal{P}(q) = p$ is a projection of the waypoint $q = (p, \theta)$ to $\mathbb{R}^2$, that is, $p \in \mathbb{R}^2$. Notice, we may further distinguish a single depot location p_d or a depot with a neighborhood defined by the sensing range δ as for other target locations. Since the practical, motivational deployment is for specified initial locations of the vehicles, we focus on depots without the neighborhoods, and $\delta > 0$ for depots is further discussed in the description of the particular methods and empirical evaluation.

For the m-DTSPN, it is requested to find m trajectories $\{Q^1, ..., Q^m\}$ satisfying the limited curvature of the Dubins vehicles (1), one for each of m vehicles, such that the length of the longest trajectory is minimal, that is, the *minmax* variant of the m-DTSPN. An individual trajectory for the rth vehicle can be considered as a solution Q^r of the DTSPN formulated as Problem 1 for a subset of objects of interest $O^r \subseteq O$ that are covered along the trajectory Q^r with the length $\mathcal{L}(Q^r, O^r)$. Besides, the initial location of the vehicle is prescribed by p_d^r. Since a solution of the DTSPN is a closed and continuous trajectory, it is sufficient that p_d^r is a part of Dubins tour; however, for this special waypoint location, the sensing range is individually set to zero. Thus, the m-DTSPN can be formulated as a problem to determine a subset of n^r locations O^r for each vehicle r, $1 \leq r \leq m$ such that all objects O are covered, and the length of the longest trajectory is minimal.

Problem 2 (m-DTSPN)

$$\begin{aligned} &\text{minimize}_{(Q^r, O^r) \text{ for } r \in \{1, ..., m\}} && \max_{r \in \{1, ..., m\}} \mathcal{L}(Q^r, O^r \textstyle\bigcup \{p_d^r\}) \\ &\text{subject to} && Q^r \text{ is a solution of Problem 3.1 for the} \\ & && \quad \text{subset } O^r \text{ and } p_d^r \\ & && O = \textstyle\bigcup_{r=1}^{m} O^r \text{ and for each } o \in O \text{ there is} \\ & && q \in \textstyle\bigcup_{r=1}^{m} Q^r \text{ such that } |(\mathcal{P}(q), o)| \leq \delta. \end{aligned} \tag{3}$$

In addition to the Dubins vehicle model (1), the addressed surveillance planning is also considered for a general 3D trajectory satisfying constraints of the utilized hexacopters, that is, the maximal velocity and acceleration. Such a problem formulation is formally identical to Problem 2 except Q^r which needs to be substituted by the parametrization of the trajectory $\mathscr{Q}^r$ and the length of the trajectory $\mathcal{L}(Q^r, O^r)$ needs to be replaced by the TTE of the trajectory $\mathcal{T}(\mathscr{Q}^r, O^r)$. For simplicity and w.l.o.g., we assume that each object of interest is covered from some point x on the determined trajectories $\mathscr{Q}^1$, ..., $\mathscr{Q}^r$ and x can be $x \in \mathbb{R}^2$ or $x \in \mathbb{R}^3$ in the case of the 3D trajectory.

Problem 3 (Surveillance Planning with a 3D Smooth Trajectory)

$$\begin{aligned} &\text{minimize}_{(\mathscr{Q}^r, O^r) \text{ for } r \in \{1, ..., m\}} && \max_{r \in \{1, ..., m\}} \mathcal{T}(\mathscr{Q}^r, O^r) \\ &\text{subject to} && O = \textstyle\bigcup_{r=1}^{m} O^r \text{ and for each } o \in O \text{ there is} \\ & && q \in \textstyle\bigcup_{r=1}^{m} \mathscr{Q}^r \text{ such that } |(\mathcal{P}(q), o)| \leq \delta. \end{aligned} \tag{4}$$

4 | BACKGROUND

4.1 | Dubins touring problem

An important part of the sampling-based approaches for the DTSP is a solution of the DTP (Faigl et al., 2017). The DTP stands to determine the optimal heading values for a given sequence of the waypoint locations, and it can be formally defined as follows. Let the given sequence of n waypoint locations be $P = (p_1, ..., p_n)$ and it is requested the vehicle returns to the initial location because of the context of solving the DTSP. The problem is to find the particular heading value at each target

location, that is, the headings $T = (\vartheta_1, \ldots, \vartheta_n)$ such that the optimal Dubins maneuvers (Dubins, 1957) connecting the targets in the sequence form a Dubins tour with the minimal length. Thus, the cost function is piecewise continuous, and the DTP is a continuous optimization problem.

Problem 4 (DTP)

$$\begin{aligned} &\text{minimize}_T\ \mathcal{L}(T, P) = \textstyle\sum_{i=1}^{n-1}\ \mathcal{L}(q_i, q_{i+1}) + \mathcal{L}(q_n, q_1), \\ &\text{subject to } q_i = (p_i, \vartheta_i),\ p_i \in P,\ \vartheta_i \in T,\ 0 \le \vartheta_i < 2\pi,\ i = 1, \ldots, n, \end{aligned} \tag{5}$$

where $\mathcal{L}(q_i, q_j)$ is the length of the shortest Dubins maneuver between q_i and q_j which can be computed by a closed-form expression (Dubins, 1957).

4.1.1 | Sampling-based solution of the DTP

Having a discrete finite set of possible heading values per each target location in the sequence P, for example, h heading values for each target, we can construct a graph where nodes represent particular vehicle states and edges represents an optimal Dubins maneuver connecting the states. For a sequence of n targets, the graph has n layers and each layer has up to h nodes (Figure 5).

Then, the optimal solution of the DTP for the given discretization of the headings can be found by a forward search of the graph. Since we need a closed tour, the graph has to be searched for h initial/termination headings, and thus, the overall time complexity of the search procedure can be bounded by $O(nh^3)$.

4.2 | Three-dimensional smooth trajectory based on Bézier curve

The utilized parametrization of the 3D smooth trajectory is based on the cubic Bézier curve, that is, defined by four control points.

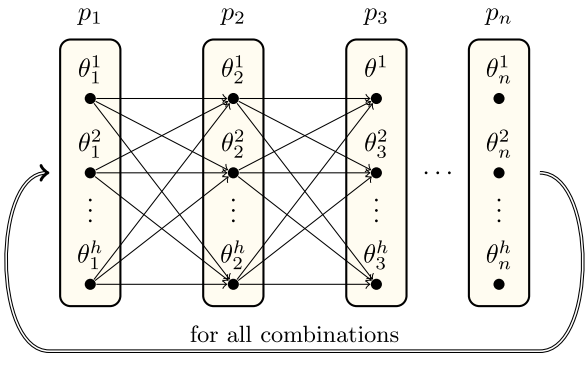


FIGURE 5 A search graph where each layer corresponds to one target location $p_i \in P$ with particular heading values $\Theta_i = \{\vartheta_i^1, \ldots \vartheta_i^h\}$. Two neighboring layers are fully connected by the oriented edges representing the optimal Dubins maneuver between the states [Color figure can be viewed at wileyonlinelibrary.com]

The first two control points define the end locations and direction of the curve directly, and two additional points define the departure and terminal tangents. Therefore, Bézier curves can be easily connected into a smooth path from multiple segments. A general Bézier curve of the dth degree can be parametrized by

$$\mathbf{X}(\tau) = \sum_{i=0}^{d} \mathbf{B}_i J_{d,i}(\tau), \quad 0 \le \tau \le 1, \tag{6}$$

where $\mathbf{B}_i$ stands for the control points and $J_{d,i}(\tau)$ is the Bézier polygon of the dth degree which prescribes weights for the control points $\mathbf{B}_i$ (Bézier, 1973). Since the Bézier polygon is given by

$$J_{d,i} = \binom{d}{i} \tau^i (1-\tau)^i, \tag{7}$$

the parametrization of the utilized cubic Bézier curve in the expanded form can be expressed as

$$\mathbf{X}(\tau) = \mathbf{B}_0(1-\tau)^3 + 3\mathbf{B}_1\tau(1-\tau)^2 + 3\mathbf{B}_2\tau^2(1-\tau) + \mathbf{B}_3\tau^3. \tag{8}$$

Notice, the Bézier curve can be used for a path parametrization in 2D and 3D, the only difference is in the dimension of the control points, that is, $\mathbf{B}_i \in \mathbb{R}^2$ and $\mathbf{B}_i \in \mathbb{R}^3$, respectively.

4.3 | Travel time estimation

Having a parametrization of the trajectory as a sequence of Bézier curves described by (8), the travel time of the vehicle along the trajectory can be computed from the velocity profile for the trajectory. The maximal velocity and acceleration of the utilized vehicles are limited individually for the horizontal movements by the maximal speed v_{horiz} and the maximal acceleration a_{horiz}. Similarly, the maximal speed v_{vert} and the maximal acceleration a_{vert} limit the vertical movements. Regarding these limitations, the vehicle velocity along the given path is adjusted to minimize the travel time of the trajectory, which is further referred as the TTE. The maximal achievable velocity is determined concerning the path curvature and the acceleration limits as follows.

The profile for the vertical velocity is directly computed from the altitude differences along the curve, and thus the first and second derivatives along the z axis are utilized, and the vertical velocity is limited by v_{vert} and a_{vert}. In the horizontal plane, two different acceleration components are affecting the vehicle simultaneously. The first one is the tangent acceleration a_{tan} which is responsible for the speed changes. The second component is the radial acceleration a_{rad} which is caused by the path curvature, but it does not directly influence the vehicle velocity. The tangent and radial accelerations are always perpendicular, and their combined value cannot exceed a_{horiz} which can be expressed as

$$a_{\text{tan}}^2 + a_{\text{rad}}^2 \le a_{\text{horiz}}^2. \tag{9}$$

The radial acceleration a_{rad} in the horizontal plane is given by

$$a_{\text{rad}} = v^2\, \kappa_h, \tag{10}$$

where κ_h stands for the horizontal curvature of the trajectory, for example, computed as

$$\kappa_h = \frac{|x'y'' - y'x''|}{(x'^2 + y'^2)^{3/2}}. \tag{11}$$

Notice that for Bézier curves, the curvature has a closed-form expression. From the curvature, the maximal possible velocity v_{pos} of the vehicle along the trajectory can be computed from

$$v_{\text{pos}} = \min\left(v_{\text{horiz}}, \sqrt{\frac{a_{\text{horiz}}}{\kappa_h}}\right). \tag{12}$$

The maximal possible tangent acceleration a_{tan} is then defined by

$$a_{\text{tan}}^2 = a_{\text{horiz}}^2 - a_{\text{rad}}^2 = a_{\text{horiz}}^2 - v_{\text{pos}}^4 \kappa_h^2. \tag{13}$$

The right side of (13) is always positive because of (12). Based on these preliminaries, the velocity profile, and thus the TTE can be numerically determined in the following six steps.

- Sample the parametrized path into a finite set of uniformly sampled points and compute the horizontal curvature of the trajectory (11) at each sample using the first and second derivatives expressed from (8).
- Set the initial and final vehicle velocity to zero.
- Determine derivatives along the z axis and limit the vertical velocity according to the v_{vert} and a_{vert}.
- Compute v_{pos} for every sampled point of the trajectory (12).
- Iterate over the samples forward and limit the vehicle velocity and acceleration by the maximum possible tangent acceleration (13), that is, adjust the travel time between the respective samples.
- Iterate over the samples backward and limit the vehicle velocity and acceleration by the maximum possible tangent acceleration (13), similarly as in the previous step.

5 | VNS FOR THE *m*-DTSPN

The proposed VNS-based solution of the *m*-DTSPN is based on the existing deployment of the VNS metaheuristic to the *m*-TSP (Soylu, 2015). The expected locations of the objects of interest O are considered as target locations in the *m*-DTSPN and the extension towards the minimal turning radius ρ of the Dubins vehicle model and nonzero sensing range δ is based on sampling possible heading values and waypoint locations. In particular, s locations are uniformly sampled for the neighborhood of each target $o \in O$ on the circle with the radius δ centered at o. Then, h possible heading values are uniformly sampled for each such a sampled location. Besides, an individual starting location for each vehicle is considered, which better corresponds with the practical deployment in the surveillance planning contrary to a common depot utilized in (Soylu, 2015). Therefore, a modified initialization of the VNS-based solver is proposed to support the individual starting locations.

The VNS metaheuristic consists of two main procedures: The *shake* and *local search*. The *shake* procedure is used to get the currently best incumbent solution x from possible local optima by changing it randomly to a solution x' within the neighborhoods $\{N_1, ..., N_{k_{\max}}\}$. On the other hand, the *local search* procedure searches fully specific neighborhoods of a solution x' using $l_{\max}$ predefined operators to find a possibly better incumbent solution, which in the addressed *minmax* variant of the *m*-DTSPN, is the one with a smaller the longest tour. The utilized procedures are detailed below, and a summary of the VNS-based solver for the *m*-DTSPN is in Algorithm 1.

Algorithm 1: VNS-based solver for the m-DTSPN

Input : $\mathcal{O} - \Lambda$ given set of objects of interest
Input : $(p_d^1, \ldots, p_d^m)$ – The requested initial locations (depots) for the m vehicles
Parameter : ρ – The minimal turning radius
Parameter : δ – The sensing range
Parameter : h – The number of heading samples
Parameter : s – The number of samples of the waypoint locations
Parameter : l_{max} – The number of local search neighborhood operators
Parameter : k_{max} – The number of shaking neighborhoods
Output : $x = \{Q^1, \ldots, Q^m\}$ – The found Dubins tours for the m vehicles

$\mathcal{O}' \leftarrow$ sampleWaypoints$(\mathcal{O}, (p_d^1, \ldots, p_d^m), s, h, \delta, \rho, m)$
$x \leftarrow$ initialization$(\mathcal{O}')$
while $m > 1$ and *stopping condition is not met* **do**
  $k \leftarrow 1$
  while $k \leq k_{max}$ **do**
    $x' \leftarrow$ shake$(x, k, \mathcal{O}')$
    $x'' \leftarrow$ localSearch$(x', l_{max}, \mathcal{O}')$
    if $\max_{Q^r \in x} \mathcal{L}(Q^r) < \max_{Q^r \in x''} \mathcal{L}(Q^r)$ **then**
      $x \leftarrow x''$
      $k \leftarrow 1$
    else
      $k \leftarrow k + 1$
return x

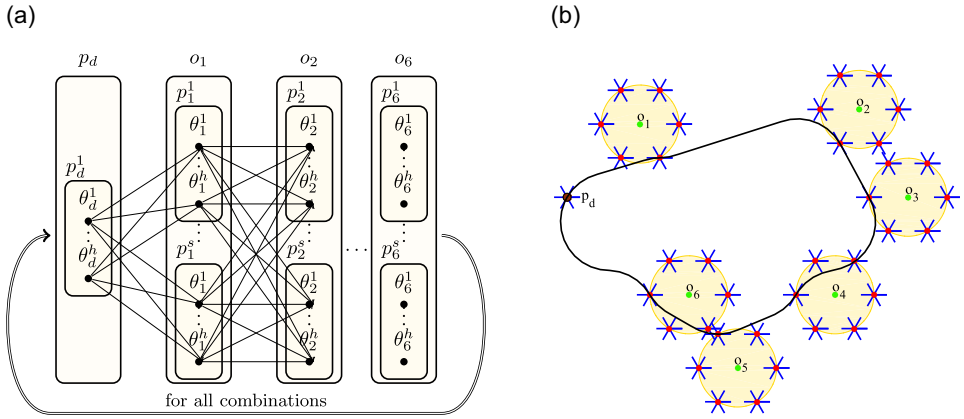


FIGURE 6 A search graph utilized in the proposed variable neighborhood search-based approach to the *m*-DTSPN with an example of the found solution. (a) A search graph with *h* heading samples per each neighborhood sample $p_1^i,...,p_s^i$ for each target location o_i. In this case, a fixed initial vehicle location (depot) p_d is considered without the neighborhood. (b) An example of a solution of the DTSPN with disk-shaped neighborhood (yellow) around each target location o_i with uniform sampling of $s = 6$ waypoint locations (red) inside the neighborhood and $h = 6$ heading samples at each possible waypoint location visualized as blue segments. DTSPN: Dubins traveling salesman problem with neighborhoods [Color figure can be viewed at wileyonlinelibrary.com]

The VNS-based solution to the *m*-DTSPN uses static sampling. Therefore, all possible Dubins maneuvers are precomputed, and all the lengths are stored in a distance matrix to reduce the computational burden during the VNS optimization. Thus, a solution of a single vehicle for the prescribed visits to the targets can be determined in a similar way as finding a Dubins tour in the DTP, just instead of *h* possible states per each target, *sh* states are considered (see the extended search graph in Figure 6).

Besides, a dynamic programming technique is utilized for storing the particular distances from the tour start in the forward direction and also from the tour end in the backward direction. For each target location o_i and the corresponding sample of the waypoint location p_i^j and sample of the heading value θ_i^k in the current solution, the shortest distances from the starting samples (i.e., all samples corresponding to the starting target location) and from the ending samples together with the respective sequence of the particular samples are stored. Then, the evaluation of the resulting path length for a simple target location removal or addition require significantly less computational time because all paths are precomputed and stored. Only the calculation of the shortest connection from the previous and to the following target location samples in the target location sequence is required without the need to find the shortest path in the whole search graph shown in Figure 6a. However, after each change to the sequence, the stored shortest paths to particular samples have to be updated. Notice, the first waypoint is the same as the final waypoint because the tours are closed in the *m*-DTSPN. Therefore, also the particular heading values and the waypoint locations (for depots with neighborhood) must be the same for both of these waypoints. When computing the shortest tour over a given sequence of sampled waypoints (as shown in Figure 6b), the shortest tour has to be evaluated for each sampled heading value at the depot to keep the tour closed and minimal. In the case of the depot with the neighborhood, it has to be also evaluated for every possible waypoint location and the heading, which is naturally more demanding.

A tour in the VNS optimization represents a sequence of the targets for which the most suitable heading and waypoint location is determined from the sampled values. Therefore, a tour for the *r*th vehicle is denoted Q^r and it is a sequence of waypoints

$$Q^r = (q_0^r, q_1^r,...,q_{n_r}^r, q_0^r), \tag{14}$$

where q_0^r is the waypoint corresponding to the requested initial and terminal location of the *r*th vehicle (i.e., the depot p_d^r) and n_r is the number of objects of interest visited by the *r*th vehicle except q_0^r. The waypoints q_u^r for $u > 0$ are alternated during the VNS optimization (while q_0^r are fixed), but each q_u^r in all tours always corresponds to a unique object of interest $o \in O$ and all objects are visited by the tours, that is, $n = \sum_{r=1}^{m} n_r$. For better readability, we consider the subscript *u* of q_u^r as an index of the particular object and its waypoint in the respective sequence of the waypoints Q^r.

The most time-consuming part of the initialization is the computation of all Dubins maneuvers between all possible waypoints in the sampleWaypoints() function, which are saved for further usage in the VNS optimization. Regarding the particular numbers for the considered *mbzirc*22 scenario with 22 targets locations and up to $m = 3$ vehicles, we consider $s = 6$ and $h = 12$ which gives up to 1,620 waypoints. For such a small number of waypoints, the initialization is fast, and it is done in tens of milliseconds using conventional computational resources; however, the precomputation becomes quickly computationally more demanding with increasing *n* and the number of samples (see empirical evaluation in Section 7).

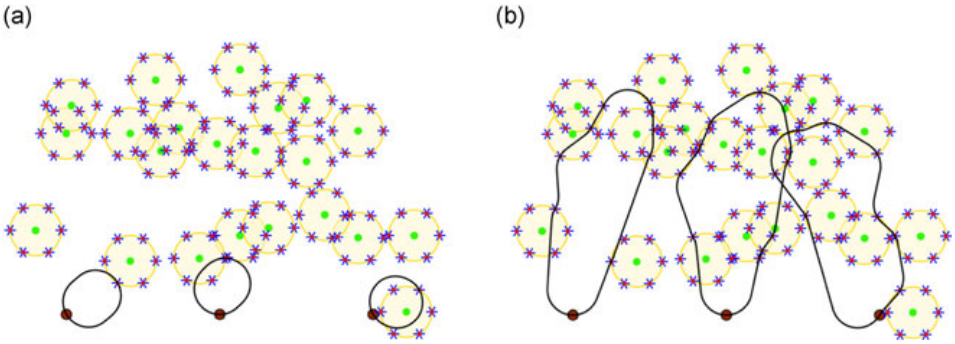


FIGURE 7 Tours for all $m = 3$ vehicles created by the proposed Initialization procedure of the variable neighborhood search based m-DTSPN (a) Initial small tours each with one target location. (b) Final solution created by the Initialization procedure. DTSPN: Dubins traveling salesman problem with neighborhoods [Color figure can be viewed at wileyonlinelibrary.com]

5.1 | Initialization procedure

The initialization heuristic utilized in the VNS-based algorithm for the m-TSP in Soylu (2015) is based on the competitive rule initially proposed to address the *minmax* variant of the m-TSP by SOM in Somhom et al. (1999) to favor shorter tours and rather do not extend the longest one. The initialization starts with sorting all the target locations $o_i \in O$ according to its minimal distance $d^i_{\min}$ to any of the m starting locations using the Euclidean distance. Then, a small tour for each vehicle $1 \le r \le m$ is created by adding one waypoint location such that the Dubins tour connecting the initial location of the rth vehicle with the added location has the minimal tour length (Figure 7a). Thus, each Q^r has the form $Q^r = (q^r_0, q^r_1, q^r_0)$ and it represents a Dubins tour consisting of two Dubins maneuvers from q^r_0 to q^r_1 and from q^r_1 to q^r_0 with the lengths $\mathcal{L}(q^r_0, q^r_1)$ and $\mathcal{L}(q^r_1, q^r_0)$, respectively. The length of the Dubins tour represented by Q^r is further denoted $\mathcal{L}(Q^r)$ for brevity.

After the creation of the first tours, particular waypoints for all not assigned objects are sequentially inserted to the tours in the order defined by the increasing distance $d^i_{\min}$ of the target location to the initial location. The respective waypoint $q(o_i)$ (together with its heading and location) is selected during the determination of the most suitable tour r^* and the particular position j^* in the tour using the competitive rule

$$r^*, j^* = \underset{1 \le r \le m, 1 \le j \le n_r}{\arg\min} \left(\mathcal{L}(q^r_j, q(o_i)) + \mathcal{L}(q(o_i), q^r_{j+1})\right) \mathcal{W}(m, r), \quad (15)$$

where $q(o_i)$ represents the most suitable location from up to s samples around o_i with the best heading of the h heading samples. The term $\mathcal{W}(m, r)$ represents the competitive weight introduced in Somhom et al. (1999) to address the *minmax* variant of the m-TSP. It is computed as

$$\mathcal{W}(m, r) = 1 + \frac{\mathcal{L}(Q^r) + \mathcal{L}^m_{\text{avg}}}{\mathcal{L}^m_{\text{avg}}}, \quad (16)$$

where $\mathcal{L}^m_{\text{avg}}$ is the average length of the Dubins tours $Q^1,\ldots,Q^m$

$$\mathcal{L}^m_{\text{avg}} = \frac{1}{m} \sum_{r=1}^{m} \mathcal{L}(Q^r). \quad (17)$$

An example of the solution created by the proposed Initialization procedure is shown in Figure 7b.

5.2 | Shake procedure

The *shake* procedure is utilized to get the currently best incumbent solution x from possible local optima by using up to $k_{\max}$ consecutive simple one point moves. Each such a single move starts with a random selection of two distinct tours $i, j \in \{1,\ldots,r\}$, $i \ne j$ and one target location in each tour $u \in Q^i$, $v \in Q^j$, where u and v are the position indexes of the particular selected target locations in the ith and jth tours, respectively. Then, the corresponding object associated with q^i_u is moved from Q^i to Q^j where it is placed after the vth position, such that the tours after the operation become $Q^i = (q^i_0,\ldots,q^i_{u-1}, q^i_{u+1},\ldots,q^i_0)$ and $Q^j = (q^j_0,\ldots,q^j_v, q^i_u, q^j_{v+1},\ldots,q^j_0)$. By using the one-point move operation for $k = 1,\ldots,k_{\max}$ times, the *shake* procedure creates a random solution x' within N_k neighborhood of the original solution (see Figure 8). The particular number of the performed operations for the results presented in this paper is $k_{\max} = 5$.

5.3 | Local search procedure

The *local search* procedure uses a randomly created solution produced by the *shake* procedure and systematically tries to find a better solution. The used variant of the procedure is called the sequential local search, which indicates the fact that all the neighborhood operators are tested in a sequence according to the value l of the six operators defined below (i.e., $l_{\max} = 6$). Once the solution quality is improved, the *local search* is started again to optimize the improved solution. Notice that the ordering of the operators in the *local search* procedure can significantly influence the final solution quality. To improve the efficiency of the VNS

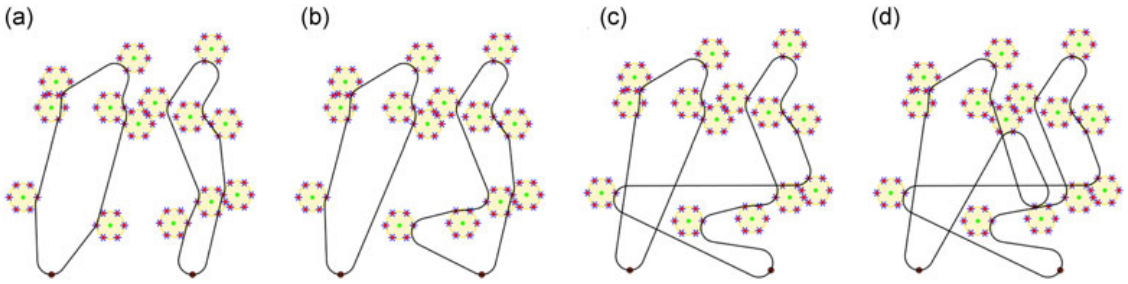


FIGURE 8 An example of the shake procedure sequence with $k = 1...3$ random moves for $m = 2$ vehicles. (a) Incumbent solution x. (b) First random move $k = 1$. (c) Second random move $k = 2$. (d) Third random move $k = 3$ [Color figure can be viewed at wileyonlinelibrary.com]

search, only operators that can decrease the length of the longest tour are considered. The *local search* operators (Soylu, 2015) adopted for the m-DTSP(N) are as follows:

- One-point move ($l = 1$) operator uses the smallest neighborhood possible to move only a single target location to a different tour.
- Or-opt2 move ($l = 2$) operator moves two adjacent target locations to a different tour.
- Two-point move ($l = 3$) operator exchanges two points (target locations).
- Or-opt3 move ($l = 4$) operator moves three adjacent target locations to a different tour.
- Three-point move ($l = 5$) operator exchanges two adjacent target locations from the longest tour with one target location in a different tour.
- 2-Opt move ($l = 6$) operator (Croes, 1958) selects two target locations in a tour and swaps the subtour between the targets which tries to improve all individual tours separately. The operator is repeatedly performed until it improves the tour length. Notice, the original idea of this heuristic is to remove unnecessary self-crosses in a solution of the ETSP.

6 | UNSUPERVISED LEARNING FOR m-DTSPN AND 3D BÉZIER CURVE-BASED MULTIVEHICLE SURVEILLANCE PLANNING

A solution of the m-DTSPN based on unsupervised learning has been introduced in Faigl and Váňa (2017), and therefore, an

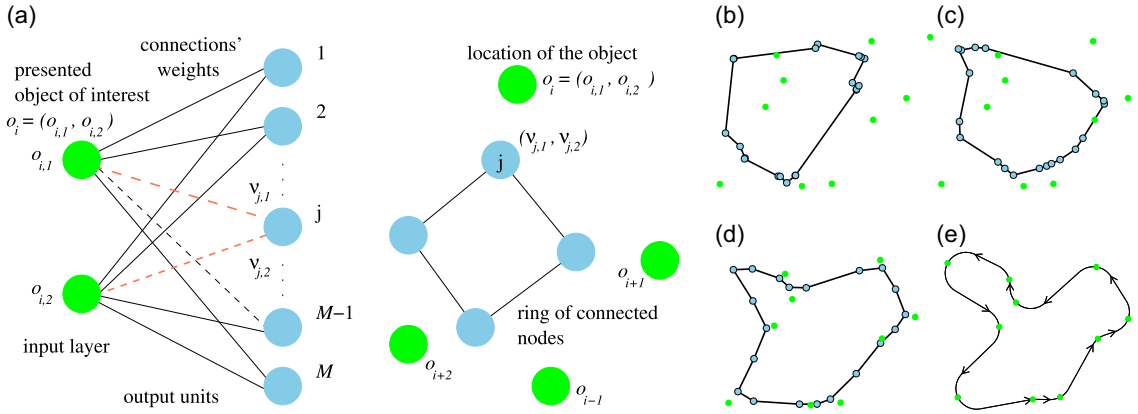


FIGURE 9 A structure of the SOM for the TSP and visualization of the ring evolution during the learning. The green disks are the target locations to be visited by the tour, and blue disks represent the neuron weights in the input space $\mathbb{R}^2$. The connections between the input and output layers represent that the best matching neuron is computed using its distance to the input signal (location). For the DTSP (Faigl & Váňa, 2017), each neuron is in addition to the neuron weights (locations) as a point in $\mathbb{R}^2$ also associated with the particular target (or waypoint) location and with h heading values, and thus a solution of the DTSP can be determined from the ring of neurons after each learning epoch by solving the related DTP, for example, using the forward search method described in Section 4.1. (a) A structure of SOM for the TSP, (b) Epoch 12, (c) Epoch 28, (d) Epoch 42, and (e) A DTSP solution. DTSP: Dubins traveling salesman problem; SOM: self-organizing map; TSP: traveling salesman problem [Color figure can be viewed at wileyonlinelibrary.com]

overview of the method is presented in this section to provide the necessary details for the proposed generalization to trajectory planning using Bézier curves. The unsupervised learning framework is based on the growing SOM for the TSPN proposed in Faigl (2018) which differs from a regular SOM, that is, usually a 2D lattice (Kohonen, Schroeder, & Huang, 2001) in the organization of the output layer and incremental adding new neurons to the network. In general, SOM for the TSP is a two-layered neural network in which the input layer serves for presenting the target locations O and the output layer is organized into an array of neurons (Angéniol, Vaubois, & Texier, 1988; Fort, 1988), which defines a sequence of visits to the targets. The neuron weights are in the same space as the input signals (target locations) and the connected neuron weights form a ring in the input space $\mathbb{R}^2$, and thus represent a closed path in $\mathbb{R}^2$ (see Figure 9a, i.e., the weights are considered as the neuron locations). SOM can be represented as a sequence of neurons $\mathcal{N} = (\nu_1, .., \nu_M)$, where M is usually more than two times the number of target locations (Somhom, Modares, & Enkawa, 1997).

The unsupervised learning of the network is realized by an iterative procedure in which all the target locations are presented to the network and for each such a presented location $o \in O$, the best matching neuron is selected in the winner selection procedure, that is, the neuron with the closest weights to o. Then, the winner neuron is adapted towards the presented input together with the neighboring neurons to the winner neuron with decreasing power of the adaptation defined by the neighboring function. In a single learning epoch, all targets are presented to the network, and a solution of the TSP can be retrieved after each epoch by traversing the ring, that is, the tour is constructed from the targets associated with their winner neurons into a sequence of targets defined by the position of the winner neurons in the ring. During the adaptation, the winner neurons are getting closer to the targets, and the network is stabilized in tens or hundreds of epochs because of cooling schedule of the power of the adaptation. An evolution of SOM in solving an instance of the TSP is shown in Figure 9. For the DTSP, the neurons are associated not only with a location in $\mathbb{R}^2$ but also with up to h heading values (Faigl & Váňa, 2016). Therefore, a solution of the DTSP can be determined after each learning epoch by a solution of the DTP with the sequence of visits to the targets defined by the order of the winner neurons in the SOM output layer (ring), for example, using the feed-forward search method presented in Section 4.1.

In addition to the headings associated with the neurons, the main part of the unsupervised learning for the DTSPN is the winner selection in which expected heading of the vehicle at the waypoint location is determined together with the waypoint location itself. The idea of the winner selection is visualized

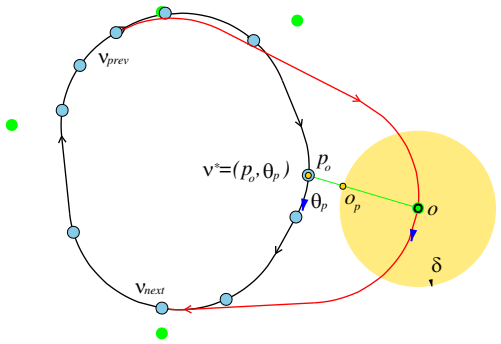


FIGURE 10 A selection of the winner neuron for the presented location o in unsupervised learning for the Dubins traveling salesman problem with neighborhoods. The current ring of neurons represents the Dubins path showed as the black curve connecting the blue neurons. The closest point p_o of the Dubins path to o is used as the neuron weights for the winner neuron. The point o_p corresponds to the alternate target location towards which the network is adapted because o can be covered within δ sensing range from the target location. The shortest possible path connecting ν_{prev} and ν_{next} through the point o using the vehicle heading θ_p is in red [Color figure can be viewed at wileyonlinelibrary.com]

in Figure 10. The range of the neighboring neurons that are adapted together with the winner neuron is restricted by the neurons ν_{prev} and ν_{next} such that the expected length of the Dubins path (see the red curve in Figure 10) to visit the target location o is minimized:

$$\mathcal{L}_g = \mathcal{L}(\nu_{\text{prev}}, (o, \theta)) + \mathcal{L}((o, \theta), \nu_{\text{next}}), \tag{18}$$

where θ is one of the h heading values associated with the winner neuron. Nevertheless, the search for ν_{prev} and ν_{next} is limited to the range $0.2M$ around the winner, where M is the current number of neurons in the ring, as in other SOM-based TSP solvers, for example, Somhom et al. (1997). The neurons adapted with the winner neuron ν^* are in the range of ν_{prev} and ν_{next} for which a value of the neighbouring function (19) is above a threshold, i.e., empirically set to 10^{-5}. The neighbouring function is defined for the active neuron in a similar way as in a regular SOM for the TSP (Cochrane & Beasley, 2003; Somhom et al., 1997):

$$f(\sigma, d) = \begin{cases} e^{\frac{-d^2}{\sigma^2}} & \text{for neurons around } \nu^* \text{ in the range defined by } \nu_{\text{prev}} \\ & \quad \text{and } \nu_{\text{next}} \\ 0 & \text{otherwise,} \end{cases}, \tag{19}$$

where σ is the learning gain and d is a distance of the neuron from ν^* in the number of neurons in the ring.

Algorithm 2: Unsupervised learning algorithm for solving the DTSPN

Input : $\mathcal{O}$ – A given set of objects of interest

Input : p_d – The requested initial location (depot) of the vehicle

Output : Q – Determined Dubins path covering $\mathcal{O}$

▷ **Initialization:**

1. For n target locations $\mathcal{O}$, create a ring $\mathcal{N}$ with one neuron with the weights set to the starting location p_d.
2. Set the learning gain $G = 10$, the learning rate $\mu = 0.6$, the gain decreasing rate $\alpha = 0.1$, and the epoch counter $i = 1$.

▷ **Learning Epoch:**

3. For each target o in the randomized set $o \in \Pi(\mathcal{O} \bigcup \{p_d\})$
   (a) *Winner selection*: Determine the point p_o together with the expected heading ϑ_p and waypoint location o_p as in Figure 10. Create a new neuron with the weights p_o and add it as a new winner ν^* to the ring.
   (b) *Adapt* ν^* and its neighbouring neurons (defined by ν_{prev} and ν_{next}(18)) to o_p using the neighbouring function (19). The weights of each adapted neuron ν are set to a new location $\nu' = \nu + \mu f(\sigma, d)(o_p - \nu)$.

▷ **Update:**

4. *Ring regeneration*: Remove all nonwinner neurons. Use the sequence of the winner neurons defined by the ring together with the headings and waypoint locations associated with the neurons to solve the DTSPN as a solution Q_i of the related DTP using the sampling-based algorithm described in Section 4.1 with the length $\mathcal{L}(Q_i, \mathcal{O})$.
5. *Update learning parameters:* $\sigma = \sigma(1 - i\alpha)$, $i = i + 1$.
6. *Termination condition:* If $i \geq i_{\max}$ or *winner neurons are negligibly close to the waypoint locations* (e.g., less than 10^{-3}) Stop the adaptation. Otherwise go to Step 3.

▷ **Final Tour Construction:**

7. *Improve the solution* Q using 2-Opt heuristic (Croes, 1958).
8. *Return* the final trajectory as a solution of the DTP found by the guided sampling with up to 1,024 samples or the approximation factor 1.01 by the algorithm (Faigl et al., 2017).

The schema of the unsupervised learning is depicted in Algorithm 6. Due to the nonmonotonicity of the length of the Dubins maneuvers, the ring may contain unnecessary loops and crossings, and therefore, the simple 2-Opt heuristic (Croes, 1958) is used to improve the solution similarly as in other SOM-based TSP solvers (Ahmad & Kim, 2015). The 2-Opt heuristic is computationally inexpensive procedure $O(n^3)$ which can improve the solution about few percentage points. In addition, the final trajectory is determined by the high-quality DTP solver (Faigl et al., 2017) which utilizes a tight lower bound (Manyam & Rathinam, 2015) to stop the refinement of the heading samples when the maximal number of samples 1,024 is reached or when the ratio of the trajectory length to the lower bound solution is less than 1.01. Notice, the sensing range δ can be easily individualized for each particular object of interest by a simple usage of the particular range in the winner selection. Besides, in the case of the fixed starting location, the range can be set to zero and the target location o is directly used as the alternate target location o_p similarly to the solution of the DTSP (Faigl & Váňa, 2016, 2017).

Based on the empirical evaluation, the parameters of the learning $\mu = 0.6$, $\alpha = 0.1$, and the initial value of $\sigma = 10$ can be considered as fixed and they have been selected as a trade-off between the computational requirements and quality of the found solutions, although they can be further tuned for specific scenarios. Thus, the only parameters of the learning procedure are the number of additional heading values h per each neuron and the maximal number of learning epochs $i_{\max}$. Regarding the results presented in Faigl and Váňa (2017), values $h > 3$ only increase the computational burden and do not significantly improve the solution quality, therefore $h = 3$ is used for all the results presented in this paper. The network is usually stabilized in around 130 learning epochs, and thus the maximal number of learning epochs $i_{\max}$ is set to $i_{\max} = 150$. A further discussion of the network convergence can be found in Faigl and Hollinger (2018). Nevertheless, a solution is available after each learning epoch using the waypoint locations associated with the winner neurons.

The computational complexity of a single learning epoch depends on the number of targets n presented to the network and the number

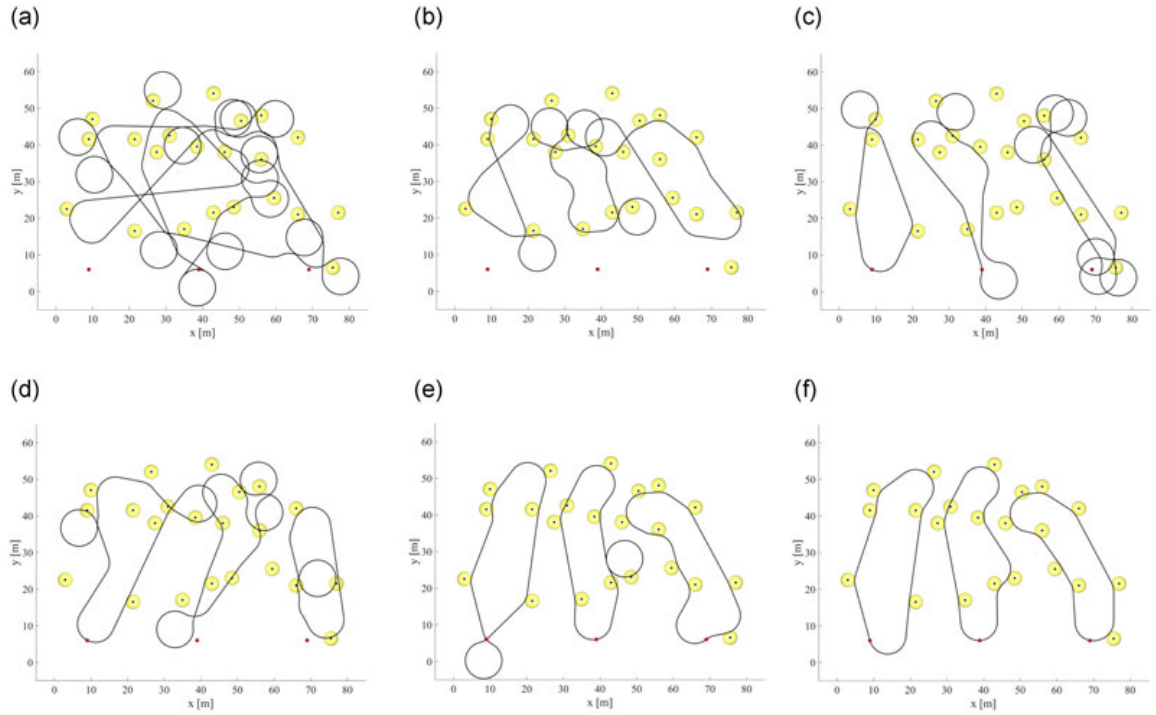


FIGURE 11 Evolution of SOM solving the *mbzirc22* *m*-DTSPN scenario with $n = 22$ target locations that are shown as small black disks. The sensing range $\delta = 2$ m is visualized by yellow disks around the target locations. The shown Dubins paths connect the neuron locations using the determined headings. The minimal turning radius of the Dubins vehicle is $\rho = 5$ m, and the initial locations of the vehicles are highlighted by the red disks. A solution is available after each learning epoch using the waypoints associated with neurons (not shown), and the network converges (the neuron locations match their waypoint locations) in 118 learning epochs. The final solution is then improved by a solution of the related DTP. (a) Epoch 1, (b) Epoch 25, (c) Epoch 50, (d) Epoch 75, (e) Epoch 118 – SOM solution, (f) final tours. DTP: Dubins touring problem; DTSPN: Dubins traveling salesman problem with neighborhoods; SOM: self-organizing map [Color figure can be viewed at wileyonlinelibrary.com]

of neurons M in the ring, which does not exceed $2n$ because of the ring regeneration (Faigl & Váňa, 2017), and thus it can be bounded by $O(n^2)$. The number of learning epochs is constant ($i_{\max} = 150$), and thus the computational complexity depends on the 2-Opt improvement that can be bounded by $O(n^3)$ and a solution of the DTP (Faigl et al., 2017), which depends on the iterative forward search procedure described in Section 4.1 with $O(nh^3)$. For a fixed $h = 3$, the total computational complexity can be bounded by $O(n^3)$ because of 2-Opt. Nevertheless, the real required computational time for the *mbzirc22* scenario is in hundreds of milliseconds as it is reported in Section 7, which perfectly fits our expectation about the computational requirements.

6.1 | Learning for a team of vehicles

The described learning procedure for a single vehicle can be straightforwardly applied to a team of vehicles by creating an individual ring of neurons for each vehicle as $\mathcal{N}^r = (\nu_1^r, .., \nu_{M^r}^r)$, where M^r is the number of the neurons in the rth ring. The application follows existing extension of the SOM-based solution for the TSP to the *minmax* variant of the *m*-TSP (Faigl, 2016; Somhom et al., 1999) where a winner neuron is preferably selected from the ring which represents the shortest tour, which is motivated to minimize the longest tour (Somhom et al., 1999). In the winner neuron determination, the distance $|(p_o, o)|$ of the point p_o on the Dubins tour represented by the current ring $\mathcal{N}^r$ and the target location o is weighted according to the difference of the length $\mathcal{L}(\mathcal{N}^r)$ of the Dubins tour represented by $\mathcal{N}^r$ and the average length of the tours represented by the rings. The winner neuron ν^* is selected from the ring r for which the respective point $|(p_o, o)|$ used as the weights of ν^* has the minimal weighted distance:

$$r = \underset{r \in \{1,\dots,m\}}{\arg\min}\, \mathcal{W}(m, r)\,|(p_o^r, o)|, \tag{20}$$

where $\mathcal{W}(m, r)$ is the competitive weight (16) with the trajectory length $\mathcal{L}(Q^r)$ computed as the length of the trajectory represented by the ring $\mathcal{N}^r$, that is, $\mathcal{L}(\mathcal{N}^r)$ is used instead of $\mathcal{L}(Q^r)$ in (16) and (17).

A straightforward usage of the learning procedure depicted in Algorithm 6 in multirobot planning would provide a set of m independent patrolling routes. Therefore, in the case the initial locations of the vehicles are prescribed by the depots $p_d^1, \dots, p_d^m$, each ring $\mathcal{N}^r$ is individually adapted towards p_d^r without the competition among the rings prior a regular adaptation of the rings to the targets O without the depots, which ensures each ring will be connected with

the respective initial location p_d^r. Moreover, for such a case it is suitable to consider the initial location without the neighborhood, and thus, for the depots, $\delta = 0$ is considered in the selection of winner neurons and adaptions. An example of the SOM evolution is visualized in Figure 11.

The computational complexity of the unsupervised learning in solving the m-DTSPN does not significantly increase because the learning depends on the number of neurons that are distributed into the particular rings. Therefore, the complexity grows only with the additional m locations that are the individual depots of the vehicles. Hence, the computational complexity can be bounded by $O((n+m)^3)$ which for $m \ll n$ can be bounded by $O(n^3)$, and thus it is independent on the number of vehicles (see Best, Faigl, and Fitch (2018) for a detail discussion.

6.2 | Surveillance planning with Bézier curves

The SOM-based solution of the m-DTSPN can be easily generalized to a different parametrization of the trajectory. Even though the Dubins vehicle is used in the above-described procedure, the unsupervised learning does not rely on the Dubins vehicle model. In fact, the Dubins maneuvers can be substituted by any curve parametrization, and in this study, we consider Bézier curves briefly introduced in Section 4.2. Since Bézier curve can be used for the parametrization of the 3D path, we do not use the kinematic model of the Dubins vehicle (1). Instead of that, we consider the hexacopters can generally follow any 3D path, and therefore, we consider the position of the UAVs along the 3D path described as a point $p = (x, y, z) \in \mathbb{R}^3$. The velocity $\mathbf{v}$ is defined by the turning angle θ and the climb/dive angle of the trajectory ψ at the position p

$$\mathbf{v} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = v \begin{bmatrix} \cos\theta\cos\psi \\ \sin\theta\cos\psi \\ \sin\psi \end{bmatrix}. \tag{21}$$

Notice, here, we do not model the orientation of the vehicle; however, the parameters of the Bézier curves are adjusted during the unsupervised learning to provide fast execution of the determined path by a real vehicle. Finally, a trajectory of the final solution is constructed from the determined Bézier curves with respect to the vehicle motion constraints using the computation of the velocity profile and the TTE described in Section 4.2.

There are two main parts of the learning procedure where additional computations related to Bézier curves have to be included: (a) The winner selection and adaptation, and (b) the determination of the trajectory represented by the ring instead of a solution of the DTP with heading values sampled during the learning. In the winner selection, the point p_o is determined in a similar way as for the DTSPN, just a sequence of the Bézier curves, each defined by four control points, (8) is utilized. However, it is requested that the final trajectory is smooth and continuous, and therefore, the following conditions have to be satisfied after the adaptation of neurons to satisfy this requirement.

Let $\mathcal{B}^i$ and $\mathcal{B}^j$ be two consecutive Bézier curves (i.e., $j = i + 1$) with the control points $(\mathbf{B}_0^i, \mathbf{B}_1^i, \mathbf{B}_2^i, \mathbf{B}_3^i)$ and $(\mathbf{B}_0^j, \mathbf{B}_1^j, \mathbf{B}_2^j, \mathbf{B}_3^j)$, respectively. The last control point $\mathbf{B}_3^i$ of $\mathcal{B}^i$ and the first control point $\mathbf{B}_0^j$ need to be identical to keep the trajectory continuous

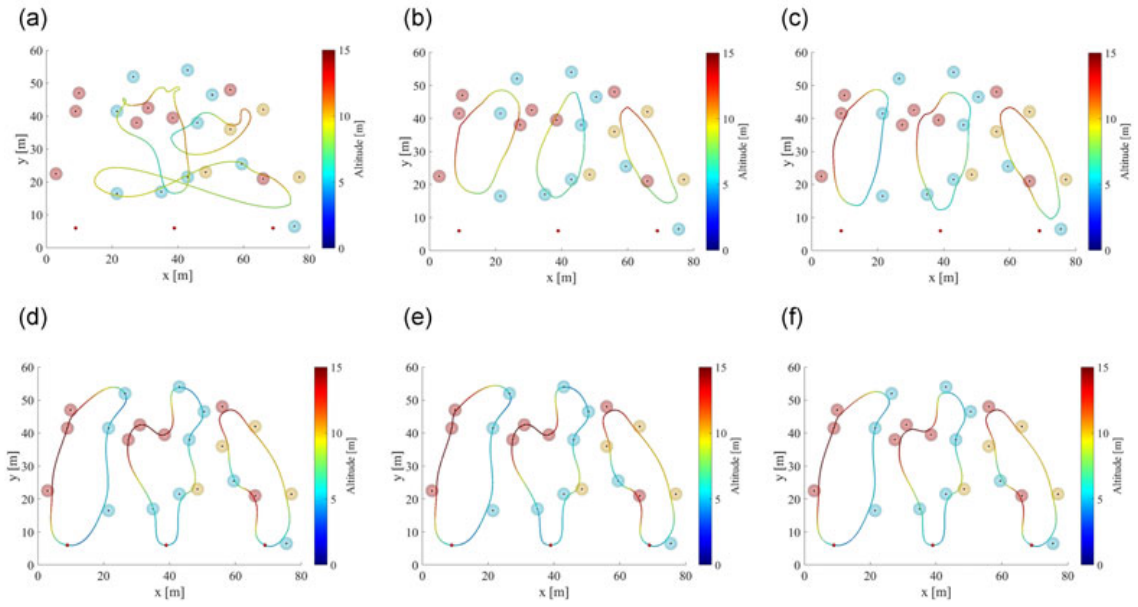


FIGURE 12 An evolution of the proposed SOM-based 3D surveillance planning using Bézier curves in solving 3D instance of the *mbzirc22* scenario with target locations at different altitudes. The target locations are visualized as small disks surrounded by a spherical neighborhood for sensing range $\delta = 2$ m. The altitude of the targets and paths is indicated by the color (from a low altitude in the blue color to the highest altitude in the red). (a) Epoch 1, (b) Epoch 25, (c) Epoch 50, (d) Epoch 75, (e) Epoch 77 – SOM solution, and (f) final tours. SOM: self-organizing map [Color figure can be viewed at wileyonlinelibrary.com]

$$\mathbf{B}_3^i = \mathbf{B}_0^j. \tag{22}$$

Besides, the tangents of the Bézier curve have to point to the same direction to support traveling of the vehicle along the final trajectory. The tangents can be defined as

$$t_a^i = \mathbf{B}_1^i - \mathbf{B}_0^i, \quad t_b^i = \mathbf{B}_3^i - \mathbf{B}_2^i \tag{23}$$

with the length of the particular tangent vector

$$l_a^i = |t_a^i|, \quad l_b^i = |t_b^i|. \tag{24}$$

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

$$l_a^j\, t_b^i = l_b^i\, t_a^j. \tag{25}$$

The waypoint locations and headings are not sufficient to define a Bézier curve represented by two neighboring neurons, which is further called maneuver. Therefore, each neuron is associated with the tangent vectors for the unique characterization of the maneuver. Each Bézier maneuver is defined by two tangent vectors separately, and thus a continuity of the velocity is ensured by (25). Hence, each neuron ν_i is associated with the heading angle ϑ_i, pitch angle ψ_i, and the lengths of the tangent vectors l_a^i and l_b^i. The tangent vectors for the two maneuvers, for which ν_i is incident with, can be expressed as

$$t_b^{i-1} = l_b^i\, \frac{\mathbf{v}}{|\mathbf{v}|}, \quad t_a^i = l_a^i\, \frac{\mathbf{v}}{|\mathbf{v}|}. \tag{26}$$

Notice that the tangent vector t_b^{i-1} corresponds with the Bézier curve $\mathcal{B}_{i-1}$ which terminates at ν_i. Conversely, t_a^i defines the initial part of $\mathcal{B}_i$, and thus the indexes of these two tangent vectors that are related to the same neuron differ.

In addition to the constraints on the consecutive Bézier curves, a local optimization of the trajectory is performed after the ring regeneration because the neurons that are not winners are removed from the ring at the end of each learning epoch. For the multivehicle planning, each ring is treated independently as an optimization problem of minimizing the TTE along the trajectory as follows.

The whole trajectory is described by the sequence of the neurons $\mathcal{N}$, where each neuron $\nu_i \in \mathcal{N}$ represents the particular parameters of the Bézier curve. A single change of one neuron influences the two incident Bézier curves, and it can also influence the velocity profile of the whole trajectory. However, based on our empirical observations, the changes are mostly local, and therefore, the optimization of the whole trajectory is performed locally and the values of ϑ_i, ψ_i associated with ν_i are numerically optimized with respect to the velocity profile of the trajectory defined by the three consecutive neurons in the ring ν_{i-1}, ν_i, and ν_{i+1}. Notice, the ring is closed, and therefore, the subscripts of the neurons are closed to the modulo of the number of neurons in the ring. An example of the evolution of the proposed SOM-based solution for the 3D surveillance planning with Bézier curves is visualized in Figure 12.

Beside these local optimizations, we used the idea of LIO (Váňa & Faigl, 2015), and the individual local optimizations of all neurons in the ring are performed in multiple iterations of the whole ring. In particular, three iterations of the whole ring are performed, and each neuron is locally optimized in each iteration. The local numerical optimization uses a step 0.5% of the variable range, and thus, the step for ϑ and ψ is 0.01π. On the other hand, the 2-Opt heuristics (Line 7 in the unsupervised learning Algorithm 6) is not utilized, because any change would require optimization of the control points. The modified learning procedure is summarized in Algorithm 6.2. Each Bézier curve is defined by the control points associated with the neurons including the locations of the neurons, and thus a feasible solution is not available after each learning epoch unless the waypoints associated with the neurons are used, and a new trajectory is determined. However, it is one of the most computationally demanding parts, especially for completely changed locations, and therefore, we do not consider the learning procedure with Bézier curves as the any-time algorithm. After the network convergence, the velocity profile for the Bézier curve is calculated numerically using 200 uniformly distributed samples for the range $\tau \in [0, 1]$ according to (8). The real computational requirements are reported in Section 7.

7 | RESULTS

An empirical evaluation of the proposed VNS-based and SOM-based solvers for the *m*-DTSPN consists of four main parts. First, the algorithms' performance is studied in the *mbzirc22*[1] scenario because of our motivation for the addressed problem. After that, the proposed generalization of the SOM-based solver for surveillance planning using Bézier curves is studied in 2D problems first, and we compare trajectories consisting of Dubins maneuvers with Bézier curves in the second part of the evaluation. Then, the proposed unsupervised learning based 3D surveillance planning with Bézier curves is studied in 3D scenarios. Finally, a brief evaluation of the algorithms' performance in larger problems is presented in the fourth part of the herein reported results to provide an overview of the expected performance of the evaluated algorithms in different scenarios.

In addition to the proposed algorithms, the memetic algorithm (Zhang et al., 2014) is included in the evaluation as it demonstrates

[1]The *mbzirc22* scenario contains 22 objects of interest positioned at the target locations (in meters): (27.5, 47.0), (10.0, 36.5), (51.5, 41.5), (32.0, 37.5), (67.0, 16.0), (44.0, 49.0), (44.0, 16.5), (49.5, 18.0), (60.5, 20.5), (39.5, 34.5), (78.0, 16.5), (67.0, 37.0), (76.5, 1.5), (28.5, 33.0), (22.5, 11.5), (57.0, 31.0), (47.0, 33.0), (4.0, 17.5), (36.0, 12.0), (57.0, 43.0), (22.5, 36.5), (11.0, 42.0). The scenario is visualized in Figure 3 where the initial locations (depots) of the vehicles are (10,1) for the first vehicle, (40,1) for the second vehicle, and (70,1) for the third vehicle.

Algorithm 3: Unsupervised learning algorithm for surveillance planning with Bézier curves

Input : O – A given set of objects of interest

Input : p_d – The requested initial location (depot) of the vehicle

Output : $\mathcal{X}, \mathcal{T}$ – Determined surveillance trajectory that covers O and the computed TTE (using velocity profile)

▷ **Initialization:**

1. For n target locations O, create a ring $\mathcal{N}$ with n neurons with the weights set such that the connected weights form a closed path around the center of the target locations.
2. Set the learning gain $G = 10$, the learning rate $\mu = 0.6$, the gain decreasing rate $\alpha = 0.1$, and the epoch counter $i = 1$.

▷ **Learning Epoch:**

3. For each target o in the randomized set $o \in \Pi(O \bigcup \{p_d\})$
   (a) *Winner selection*: Determine the point p_o and waypoint location o_p similarly as in Figure 10 but p_o is the closest point of the sequence of Bézier curves to the location of o. The particular Bézier curve on which p_o is located is constructed from the respective two consecutive neurons and the associated control points, that is, the tangents (directions) of the curves (Section 4.2). Create a new neuron ν^* with the weights according to p_o (for the location) and the tangents according to the Bézier curves to split it into two parts, and add ν^* to the ring.
   (b) *Adapt* ν^* and its neighbouring neurons to o_p using the neighbouring function (19) but with the neighborhood defined as $0.2M$, where M is the current number of neurons in the ring. The weights of each adapted neuron ν are set to a new location $\nu' = \nu + \mu f(\sigma, d)(o_p - \nu)$, that is, only the control points corresponding to the neuron location are modified and the tangents remain the same.

▷ **Update:**

4. *Ring regeneration*: Remove all nonwinner neurons. Use the sequence of the winner neurons defined by the ring together with the control points and waypoint locations associated with the neurons to optimize the sequence of Bézier curves using LIO (Váňa & Faigl, 2015).
5. *Update learning parameters*: $\sigma = \sigma(1 - i\alpha)$, $i = i + 1$.
6. *Termination condition*: If $i \geq i_{\max}$ or *winner neurons are negligibly close to the waypoint locations* (e.g., less than 10^{-3}) Stop the adaptation. Otherwise go to Step 3.

▷ **Final Tour Construction:**

7. *Return* the final trajectory as a sequence of Bézier curves for which the velocity profile is determined using the procedure described in Section 4.2.

high-quality solution in Faigl and Váňa (2017), although it is computational demanding. Regarding the motivation, the computational time for the VNS and memetic solvers has been limited to 1, 5, 10, and 60 s, because of our initial intention to have a solution in less than 1 s. The SOM provides a solution of the addressed problem in less than 1 s, and therefore, the computational time is not explicitly limited, but the maximal number of the learning epochs is set to $i_{\max} = 150$. The particular values of VNS solver parameters and also learning parameters of the SOM are used as they are reported in Section 5 and Section 6, respectively. The parameters of the Memetic algorithm are selected as in (Faigl & Váňa, 2017 according to the recommendation of (Zhang et al., 2014, i.e., the population size is set to $20n$, where n is the number of target locations of the solved problem.

All the evaluated algorithms are randomized; therefore 20 trials are computed for every problem instance by each of the evaluated algorithms, and the reported performance indicators are computed as the average values accompanied by the standard deviations and the best-found solution from the solved trials. The indicators of the solution quality are computed as the length of the longest Dubins tour among the tours for the vehicles in the team. Besides, the TTE is used in the case of Bézier curve and velocity profiles computed for the Dubins tours. In addition to average values of the length of the longest tour L_{avg}, and its standard deviation L_{std}, the quality of the best solution among the trials is reported as L_{best}.

The computational requirements are measured as the real required computational time. All the algorithms have been implemented in C++, and they use the same implementation for computing Dubins maneuvers and solution of the related DTP. All implementations are compiled by the same compiler Clang 4.0 and executed within the identical computational environment using a single core of the iCore7 processor running at 4 GHz. Therefore, all the reported computational times represent realistic requirements and can be directly compared.

The particular evaluated algorithms and their variants with the restricted computational time are denoted: Memetic 1 s, Memetic 5 s, Memetic 10 s, Memetic 60 s, VNS 1 s, VNS 5 s, VNS 10 s, VNS 60 s, SOM (Dubins), and SOM (Bézier). The problems being solved are parametrized by the number of vehicles

TABLE 2 Average and best found solutions of the m−DTSP *mbzirc22* scenarios

	$m = 1$			$m = 2$			$m = 3$		
Method	L_{best}	L_{avg}	L_{std}	L_{best}	L_{avg}	L_{std}	L_{best}	L_{avg}	L_{std}
Memetic 1 s	402.8	451.8	20.1	258.4	292.1	16.6	194.1	227.3	13.1
Memetic 5 s	318.5	343.3	17.4	194.4	222.1	19.0	139.6	163.2	15.9
Memetic 10 s	310.7	330.3	10.9	180.4	206.1	16.2	134.3	159.2	12.3
Memetic 60 s	306.4	323.6	28.7	170.7	193.2	13.6	131.0	145.1	6.7
VNS 1 s	318.6	318.6	0.0	173.7	173.7	0.0	130.5	133.8	1.5
VNS 5 s	318.6	318.6	0.0	173.7	173.7	0.0	130.5	133.2	0.9
VNS 10 s	318.6	318.6	0.0	173.7	173.7	0.0	130.0	132.3	1.6
VNS 60 s	318.6	318.6	0.0	173.7	173.7	0.0	130.0	130.6	0.8
SOM	311.2	326.8	10.6	170.5	195.5	14.5	136.3	156.1	13.0

Note. DTSP: Dubins traveling salesman problem; SOM: self-organizing map.

$m \in \{1, 2, 3\}$, the sensing range δ limited to $0 \le \delta \le 5$ m, and the minimal turning radius ρ, which has the default value $\rho = 5$ m. For the comparison of the Dubins maneuvers with the Bézier curves the value of ρ is selected from the set $\rho \in \{5, 6, 7, 8, 9, 10, 11, 12, 12.5\}$ in meters and the velocity profile is computed for $v_{horiz} = 5\ ms^{-1}$ and the maximal vehicle acceleration $a_{horiz} = 2\ ms^{-2}$, which are also used for velocity profiles along the trajectories consisting of Bézier curves.

7.1 | Performance evaluation in m-DTSP and m-DTSPN

The m-DTSP formulation represents the basic surveillance planning and the lengths found by the evaluated algorithms for m vehicles are reported in Table 2, where the best results found under less than 1 s are highlighted in bold, while the shortest solution regardless of the computational requirements are underlined. For a single vehicle, the SOM-based approach provides the best solution in less than 250 ms, which is a bit more demanding than the initialization part of the VNS (further denoted as the VNS Init), see computational requirements depicted in Table 3. For the relatively small problem *mbzirc22* and $m = 1$, the VNS initialization is very fast, and a solution is provided in less than 100 ms. Besides, the standard deviation for SOM is about 10 m, and therefore, the most suitable algorithms seem to be the SOM-based planning framework and VNS-based optimization. However, for a team of UAVs, the best solutions found in less than 1 s are provided by the proposed VNS solver, and they are found with very low standard deviations because they are mostly based on the initial solutions.

TABLE 3 CPU time–SOM and initialization of the VNS in m-DTSPN Scenarios

	SOM–T_{CPU} (ms)			VNS Init–T_{CPU} (ms)		
Problem	$m = 1$	$m = 2$	$m = 3$	$m = 1$	$m = 2$	$m = 3$
*m*bzirc22, $\delta = 0.0$	224.2	195.5	173.9	94.5	50.5	39.9
*m*bzirc22, $\delta = 0.5$	165.9	157.5	159.7	1,108.8	1,232.8	1,206.8
*m*bzirc22, $\delta = 1.0$	152.7	151.1	151.6	1,312.2	1,160.1	1,100.0
*m*bzirc22, $\delta = 2.0$	134.4	135.2	142.0	1,978.8	1,085.9	983.7

Note. CPU: central processing unit; DTSP: Dubins traveling salesman problem; SOM: self-organizing map.

The memetic algorithm is capable of providing high-quality solutions, but as it has been reported in other studies mentioned in the related work, it is computationally demanding. Even though only the single *mbzirc22* scenario is evaluated, the results indicate the VNS probably scales better with the number of vehicles than the memetic algorithm. Contrarily, the solution improvement for increasing computational time is more evident for the memetic algorithm than for the VNS which is highly related to the proposed initialization of the VNS. Therefore, the main observation from the results is that the proposed initialization procedure (Section 5.1) performed prior the VNS optimization perfectly fits the properties of the *mbzirc22* scenario and our practical deployment.

The performance of the algorithms in the m-DTSPN instances with sensing range $\delta = \{0.0, 2.0\}$ meters is depicted in Figure 13 and the best solutions found by the selected algorithms are visualized in Figure 14. Most of the algorithms provide slightly shorter solutions for increasing δ; however, two and three vehicles are more beneficial, and the longest tour is shortened more significantly than for a longer δ. Even though SOM provides better results than the memetic 1 s (as it is reported in Faigl and Váňa, 2017), it can be noticed that the SOM-based approach provides a bit worse results for $\delta > 0$ than for $\delta = 0$, which is especially noticeable for $m = 2$. It is probably caused by marking the neurons within the δ distance from the target as the winner without the adaptation as the neuron already covers the target. Besides, it can be related to the nonmonotonicity of the length

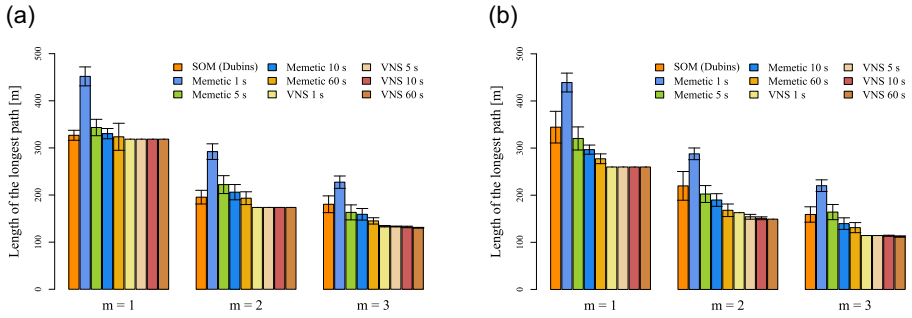


FIGURE 13 Average lengths of the longest path found by the evaluated algorithms in the *mbzirc22 m*-DTSPN scenario with the sensing range δ and m vehicles. The shown lengths are average values computed from 20 trials, and the standard deviations are shown as error bars, and very low values are not visible. (a) m-DTSP, $\delta = 0.0$ m; (b) m-DTSPN, $\delta = 2.0$ m. DTSPN: Dubins traveling salesman problem with neighborhoods; SOM: self-organizing map; VNS: variable neighborhood search [Color figure can be viewed at wileyonlinelibrary.com]

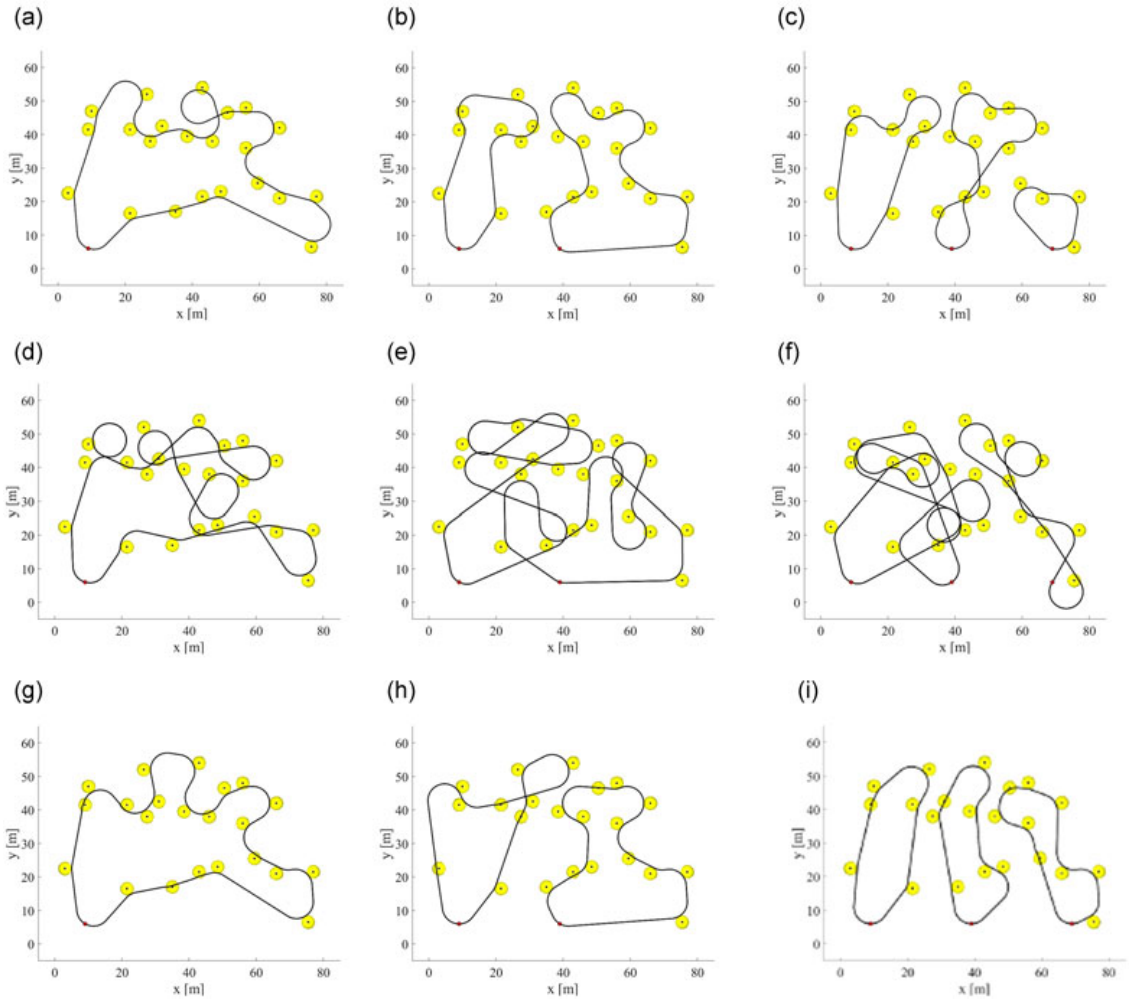


FIGURE 14 Selected best found solutions of the *mbzirc22 m*-DTSPN scenario $\delta = 2$ m and with one (left), two (middle), and three (right) vehicles found by the evaluated algorithms with the computational time limited to 1 s. (a) SOM (Dubins), $L_{best} = 287.5$ m; (b) SOM (Dubins), $L_{best} = 176.1$ m; (c) SOM (Dubins), $L_{best} = 136.8$ m; (d) memetic 1 s, $L_{best} = 398.2$ m; (e) memetic 1 s, $L_{best} = 266.1$ m; (f) memetic 1 s, $L_{best} = 202.4$ m; (g) VNS 1 s, $L_{best} = 259.9$ m; (h) VNS 1 s, $L_{best} = 163.0$ m; (i) VNS 1 s, $L_{best} = 114.4$ m. DTSPN: Dubins traveling salesman problem with neighborhoods; SOM: self-organizing map; VNS: variable neighborhood search [Color figure can be viewed at wileyonlinelibrary.com]

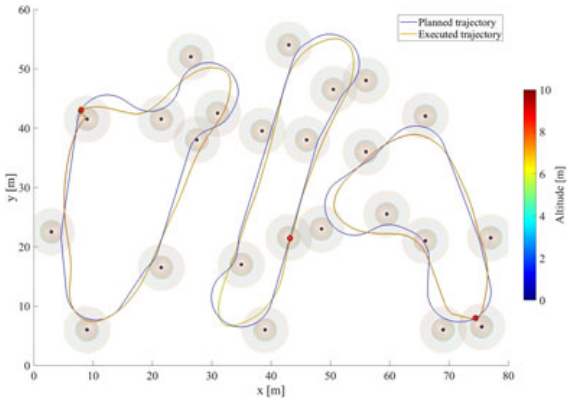


FIGURE 15 Planned and real executed trajectories by unmanned aerial vehicles, results adopted from Faigl and Váňa (2017) [Color figure can be viewed at wileyonlinelibrary.com]

of Dubins maneuvers as such behavior is not observed in Faigl and Hollinger (2018) nor for Bézier curves (Figure 15). However, this phenomenon needs to be further investigated.

The required computational times of the SOM and the proposed initialization procedure for the VNS are presented in Table 3. The SOM-based solver scales well with increasing m and δ. It is partially because the network converges in less number of learning epochs, but mostly because of saving the adaptation once a winner neuron is in the neighborhood of the particular target location. Decreasing computational requirements with m can be surprising, as algorithms are usually more demanding for multirobot problems. However, the SOM benefits from the spatial allocation of the neurons and the total number of neurons is almost the same as for the single robot instances. Therefore, a selection of ν_{prev} and ν_{next} in the minimization of (18) is quicker because fewer neurons are in the ring and the most time-consuming operation is the computation of Dubins maneuvers. Notice, in SOM, Dubins maneuvers are computed on demand, and none of the precomputed distances are utilized because sampling of the heading and waypoint locations is performed online during the learning.

The SOM is far the fastest solver from the evaluated algorithms, especially for the m-DTSPN instances where the VNS initialization suffers from the sampled waypoint locations ($s = 6$) which make the construction of the initial tours demanding. For nonzero sensing range δ, the initialization takes more than 1 s, and therefore, VNS 1 s does not satisfy the limit on the computational time. Also, less time is available for the VNS optimization for the higher limit of the computational time because of the demanding initialization. Besides, the optimization itself is also demanding because of the evaluation of the possible waypoint locations, and thus only a few iterations are performed, and the solution is not improving within the given time limit up to 60 s.

The memetic algorithm provides worse results than SOM for 1 s limit, but it is capable of improving the solution if more computational time is available. However, it starts with a relatively poor solution, and even in 60 s, the solution is improved to be only close to the solution provided by the VNS solver.

Based on the reported results, it can be summarized that the proposed SOM-based approach for the m-DTSPN can be preferred whenever the computational requirements matters. It is a far way the fastest approach providing solutions in hundreds of milliseconds, and thus it perfectly fits real-time requirements. On the other hand, if the computational requirements are not limited, the proposed VNS-based approach is capable of providing best solutions. However, in the case of exploiting nonzero sensing range δ, depending on the selected number of samples of possible waypoint locations and heading values, the VNS-based algorithm can be quickly computationally demanding.

7.1.1 | Real deployment

Verification that the planned paths are feasible for the real vehicles has been performed in real experiments with three vehicles, and thus the setup of the experiment corresponds to the evaluated *mbzirc22* scenario with $m = 3$. Since the mutual trajectory collisions are not explicitly addressed in the m-DTSPN formulation, a solution without mutually crossing trajectories is selected from the found trajectories. It is not a big issue for the *mbzirc22* instances because the initial locations of the vehicles support splitting the field that is approximately 60 m × 80 m large (Figure 3). Besides, in our early results on the SOM-based planner (Faigl & Váňa, 2017), we further consider initial positions of the vehicles not only with different coordinates along the x axis but also along the y axis (see the small red disks denoting depots in Figure 17). In addition, the SOM-based solver tends to find mutually noncrossing paths because of SOM property to preserve the topology of the input space (Faigl, 2016). The mutually noncrossing tours are not guaranteed, and this can be further addressed by adjusting velocity profiles which is out of the scope of the herein presented approach. Nevertheless, empirical results provide sufficient solutions that have been deployed in the field testing.

A snapshot of the planned and real trajectories is visualized in Figure 17. The UAVs have been operating at the altitude of 7 m with the trajectory following provided by the MPC (Báča et al., 2016). The real-time kinematic global positioning system (GPS) with precision less than 2 cm has been utilized for controlling the UAVs and recording the real trajectories. The particular value of sensing range according to the camera field of view is 4 m, but δ has been set to $\delta = 2$ m for the trajectory planning because of noise and imperfections in the trajectory following, which can be observed in the recorded real trajectories. Even though the trajectory following is not perfect, a sufficient vicinity of the object of interest has been achieved. Besides, the real deployment of the proposed Dubins-based planning approach has been thoroughly validated during participation of the CTU team in MBZIRC 2017 (MBZIRC, 2017). Thus, a further step in the proposed approach for surveillance planning with UAVs is the utilization of Bézier curves as the trajectory parametrization.

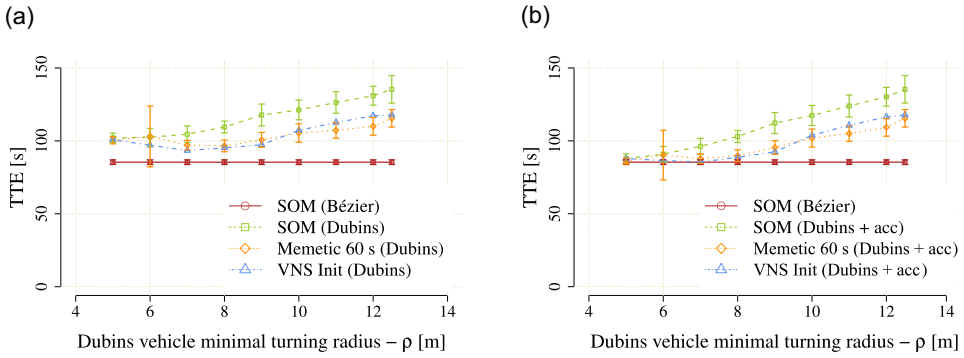


FIGURE 16 Average values of the TTE along the planned trajectories for the Dubins vehicle with different minimal turning radius ρ. The SOM-based framework allows usage of the Dubins vehicle model denoted as the SOM (Dubins) and Bézier curves denoted as the SOM (Bézier). (a) A constant forward velocity;(b) acceleration/deceleration up to v_{horiz}. SOM: self-organizing map; TTE: travel time estimation; VNS: variable neighborhood search [Color figure can be viewed at wileyonlinelibrary.com]

7.2 | Surveillance planning with Dubins maneuvers and Bézier curves

The proposed extension of the unsupervised learning to the utilization of Bézier curves is compared with the Dubins vehicle model utilized in the SOM, memetic, and VNS algorithms in a single vehicle scenario. The planning with Bézier curves directly optimizes the TTE using the velocity profiles computed according to the algorithm presented in Section 4.2. On the other hand, the Dubins vehicle model assumes a constant forward velocity v, and therefore, the TTE can be computed from the path length and v. However, the velocity depends on the allowed radial acceleration when the vehicle follows the circular path with the minimal turning radius ρ. We consider the maximal allowed horizontal acceleration $a_{\text{horiz}} = 2\ \text{ms}^{-2}$ for which the forward velocity is computed as $v = \sqrt{\rho a_{\text{horiz}}}$. Hence, the vehicle can travel at the maximal horizontal velocity $v_{\text{horiz}} = 5\ \text{ms}^{-1}$ along Dubins maneuvers with $\rho = 12.5\,\text{m}$, which may provide longer paths. Shorter paths can be determined for shorter ρ, but the vehicle needs to travel with a lower velocity. Therefore, we consider $5.0 \le \rho \le 12.5$ and determine the most suitable ρ for the *mbzirc22* instance of the DTSP regarding the TTE.

In addition to such a simple computation of the TTE along the Dubins path with a constant forward velocity, we determine a faster trajectory considering the motion of the hexacopter is limited by the maximal acceleration and not by the minimal turning radius. Therefore, the hexacopter can accelerate on the straight segments of the Dubins path, and a lower velocity v_{turn} can be computed for the turning segments according to the maximal allowed horizontal acceleration a_{horiz} as

$$v_{\text{turn}} = \min(v_{\text{horiz}}, \frac{a_{\text{horiz}}}{\kappa_h}), \tag{27}$$

where κ_h is the horizontal curvature (11) of the trajectory. For the turning radius ρ, (27) can be expressed as

$$v_{\text{turn}} = \min(v_{\text{horiz}}, a_{\text{horiz}}\rho). \tag{28}$$

The average values of the TTE for different ρ are depicted in Figure 16, where the simple computation of the TTE based on constant forward velocity is denoted (Dubins), and the results for the acceleration on straight segments are denoted (Dubins + acc). Notice that for the VNS, only the initialization part is performed because of $m = 1$.

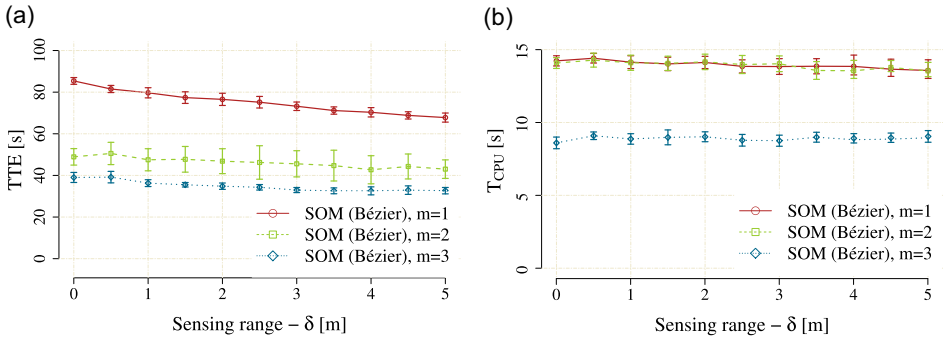


FIGURE 17 Average values of the TTE (left) and required computational times (right) for the proposed SOM-based surveillance planning with Bézier curves. SOM: self-organizing map; TTE: travel time estimation [Color figure can be viewed at wileyonlinelibrary.com]

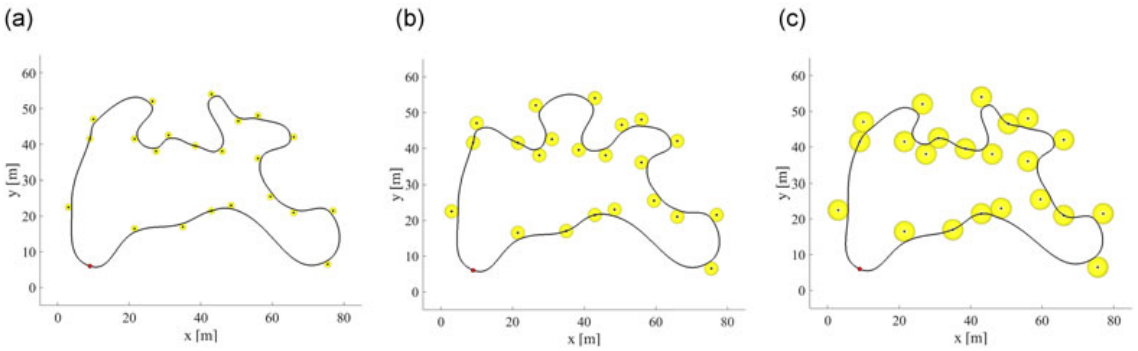


FIGURE 18 Selected found solutions for the proposed SOM-based surveillance planning with Bézier curves for a single vehicle ($m = 1$) and various sensing ranges. (a) $\delta = 1$ m, $TTE = 77$ s. (b) $\delta = 2$ m, $TTE = 70.8$ s. (c) $\delta = 3$ m, $TTE = 69.6$ s. TTE: travel time estimation [Color figure can be viewed at wileyonlinelibrary.com]

The fastest trajectories are provided by the SOM-based planning with Bézier curves. The results further indicate that for the considered *mbzirc22* scenario, the fastest Dubins paths are provided by the proposed initialization of the VNS-based algorithm with $\rho = 7$ m, for which the TTE is a bit shorter (85.5 s) than the found trajectory consisting of Bézier curves with TTE = 86.6 s. However, suitability of the particular value of ρ depends on the configuration of the target locations as it is indicated by shorter ρ and trajectories determined by the other solvers with the acceleration of the vehicle on the straight line segments. Therefore, Bézier curves seem to be a more suitable option than the Dubins vehicle model. Here, it is worth noting that the proposed SOM-based planning with Bézier curves converges better than with the Dubins maneuvers with nonmonotonicity of the maneuver length, which can also be the reason for better solutions found by the unsupervised learning than the simple heuristic used in the VNS Init.

The surveillance planning with Bézier curves is further evaluated in instances of the *mbzirc22* scenario with increasing sensing range δ and m vehicles. The average value of the TTE together with the real required computational times are depicted in Figure 15. Also, in this case, adding more vehicles to the team decreases TTE more significantly than increasing δ, but an improvement for a longer sensing range is noticeable (Figure 18). The computational requirements of the unsupervised learning are almost about two orders of magnitude higher than using the Dubins vehicle model. It is mainly because Dubins maneuvers are determined analytically while a numerical optimization is utilized for the optimization of Bézier curves. Nevertheless, solutions are provided in less than 15 s using the conventional computational resources.

Real experimental verification is not performed for the 2D Bézier curves and velocity profiles of the Dubins tours because of the utilized MPC-based controller (Báča et al., 2016) which guarantees the trajectories are finished at the desired time. Therefore, Bézier curves are further evaluated in the 3D surveillance scenarios in the next section.

7.3 | Performance evaluation in 3D surveillance scenarios

The usage of Bézier curves provides a great advantage in a direct deployment of the proposed unsupervised learning based planning in 3D surveillance scenarios. The testing scenario *mbzirc22* has been extended to 3D by adding altitudes to the particular target locations. It can be expected that a high variance in the target altitudes would need a longer trajectory than the 2D scenario. Therefore, we consider two scenarios with different ranges of the altitude changes to study limits of the maximum horizontal velocity v_{horiz} and the maximal vertical velocity as v_{vert}. For low altitude changes in the range of 5 and 10 m, the vehicle mostly needs to travel horizontally, and thus it is expected the vehicle velocity will be saturated at v_{horiz} more frequently than for high altitude changes of the target locations in the range of 5 and 20 m, where the vehicle needs to change the altitude, and thus, it is limited by v_{vert}. The considered horizontal limits are the same as for the 2D planning, that is, $v_{\text{horiz}} = 5$ ms^{-1} and $a_{\text{horiz}} = 2$ ms^{-2}. The vertical limits correspond to the capabilities of the used real UAVs and are set to $v_{\text{vert}} = 1$ ms^{-1} and $a_{\text{vert}} = 1$ ms^{-2}.

An example of the 3D surveillance scenarios with low and high altitude changes in the target locations created from the *mbzirc22* scenario is depicted in Figure 19 together with the horizontal position and altitude along the trajectories and the corresponding velocity profiles. As expected, increasing altitude changes increases the time needed to travel along the trajectory, and therefore, we validated the trajectories in a real experimental deployment.

7.3.1 | Experimental validation of the 3D surveillance planning

Feasibility of 3D trajectories planned by the proposed SOM-based framework with Bézier curves has been validated in a real experiment with three vehicles and modified *mbzirc22* scenario with the target locations at different altitudes. The same hardware and GPS-based localization as in the validation of the Dubins tours have been utilized. A snapshot from the field experiment is depicted in Figure 20.

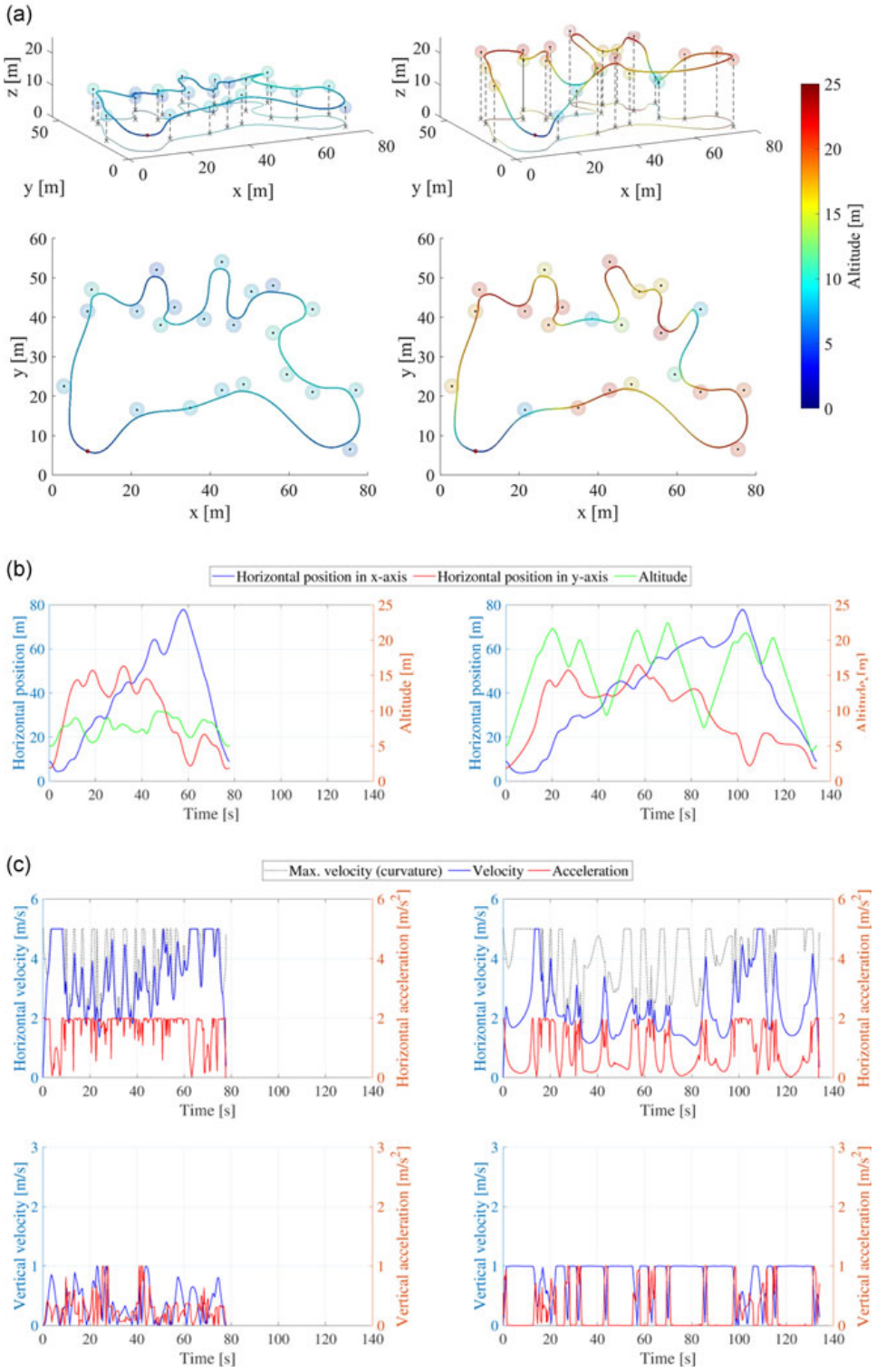


FIGURE 19 An example of the 3D surveillance problems with low (left) and high (right) altitude changes in the target locations of the *mbzirc22* scenario and solutions found by the proposed SOM-based algorithm with Bézier curves (top), particular positions of the vehicles along the found paths (middle), and velocity and acceleration profiles (bottom). (a) Three-dimensional surveillance scenarios with low (left) and high (right) altitude changes and found solutions. (b) A position of the vehicle along the trajectories. (c) Velocity and acceleration profiles for vehicles traveling along the determined trajectories. SOM: self-organizing map [Color figure can be viewed at wileyonlinelibrary.com]

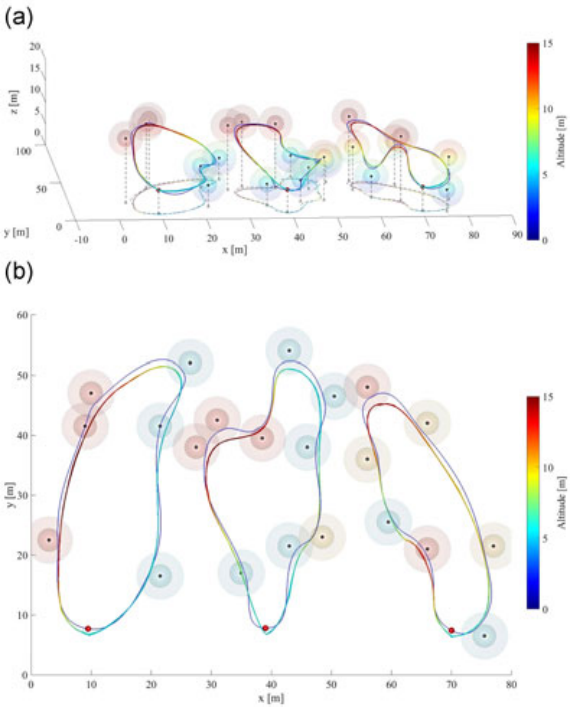


FIGURE 20 Planned and real trajectories executed simultaneously on three hexacopters. The target locations of the *mbzirc22* scenario are depicted as small black spheres each surrounded by its spherical neighborhood $\delta = 2$ m. The three additional initial locations of the vehicles are positioned at the altitude of 5 m, and they are visualized as small red spheres. The planed trajectories are shown by blue curves, and the trajectories from two experimental trials are visualized by curves with the color based on the altitude (from blue to red for increasing altitude) [Color figure can be viewed at wileyonlinelibrary.com]

The evaluated scenario with the sensing range $\delta = 2$ m together with the planned and real trajectories of the vehicles from two trials are depicted in Figure 21 with the corresponding planned and real velocity profiles presented in Figure 22. Also in this real deployment, the MPC-based trajectory following controller (Báča et al., 2016) has been utilized, and all the planned trajectories have been found feasible. Besides, all the target locations have been visited within the requested distance, which is $\delta + 2$ m because of the identified behavior of the MPC controller as in the previous case. The vehicles reached their velocity limits in both direction, that is, v_{horiz} and v_{vert}, which is indicated in Figure 22.

FIGURE 21 A snapshot of the three unmanned aerial vehicles deployed in the experimental verification of following the planned 3D trajectories [Color figure can be viewed at wileyonlinelibrary.com]

7.4 | Performance evaluation in complex instances

In this part of the presented results, we report on an overview of the performance of the evaluated algorithms in instances that are beyond the motivational *mbzirc22* scenario with the aim to show particular properties that may not be directly visible from the previous results. First, we focus on the proposed initialization of the VNS algorithm, which seems to be fast and powerful in the *m*-DTSP instances but it becomes quite demanding in *m*-DTSPN with $\delta > 0$ as it is shown in Table 3. Moreover, it is even more demanding in the instances where the initial vehicle position is not a single waypoint location, but it is considered with the same neighborhood δ as the other target locations. The required computational times for the *mbzirc*22 scenario with $\delta = 2$ m and m vehicles is depicted in Table 4. SOM-based solvers are not influenced by $\delta > 0$, but the VNS initialization is several times more demanding when the neighborhood is considered for the initial locations of the vehicles, that is, $\delta_d = \delta$, and thus, the solution is not provided in less than the desired 1 s.

An additional evaluation is focused on the solution of large problems since the *mbzirc22* scenario is relatively small. Therefore, the solvers have been deployed in two random instances with 50 and 100 relatively dense target locations. In this case, the minimal turning radius for the Dubins vehicle is set to $\rho = 1$ m, but all other parameters are the same as in Section 7.2.

The average values of the TTE using acceleration/deceleration for the Dubins maneuvers are shown in Figure 23, and selected solutions are depicted in Figure 24 for $n = 50$ and in Figure 25 for $n = 100$ target locations. The results indicate that the fastest trajectories are provided by the SOM solver with Bézier curves.

The memetic algorithm is not competitive with the proposed SOM nor the VNS algorithm. Notice, for large instances, the initialization of the VNS can be more demanding, and thus more than the dedicated computational time can be spent on the creation of the first feasible solution (Figure 26).

Regarding the computational requirements, the best trade-off between the solution quality and computational time is provided by the VNS-based solver or more precisely by the initialization procedure proposed in Section 5.1. For the larger problem with $n = 100$, the VNS initialization takes about 15 s while SOM with Dubins maneuvers takes only about three seconds for $m = 1$. However, for more vehicles, the VNS initialization is less demanding, and it is competitive to the SOM with the Dubins vehicle model. The heuristic initialization of the VNS works faster with more vehicles because the evaluation of all possible insertions is faster for tours with fewer targets.

7.5 | Discussion

Based on the presented results, the SOM-based algorithm scales better in the problems with nonzero sensing range, but for the *m*-DTSP, the superior results are provided by the proposed heuristic

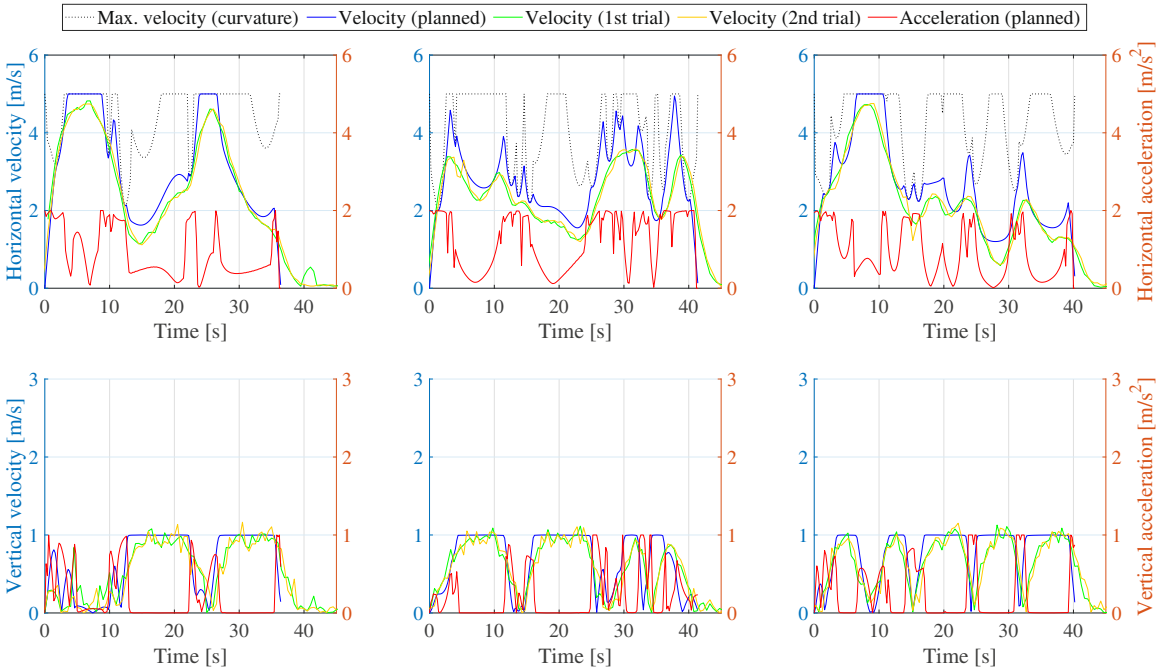


FIGURE 22 Planned and real horizontal (top) and vertical (bottom) velocity profiles from two experimental trials for each of the vehicle. Each column corresponds to one vehicle according to Figure 21 (from left to right) [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 4 Influence of vehicle initial locations with neighborhood in *mbzirc22* and $\delta = 2.0$ m

Depot	SOM (Bézier)–T_{CPU} (s)			SOM (Dubins)–T_{CPU} (s)			VNS Init–T_{CPU} (s)		
δ_d (m)	$m = 1$	$m = 2$	$m = 3$	$m = 1$	$m = 2$	$m = 3$	$m = 1$	$m = 2$	$m = 3$
$\delta_d = 0$	8.5	8.7	8.9	0.1	0.1	0.1	2.0	1.1	1.0
$\delta_d = 2$	8.3	9.3	8.9	0.2	0.2	0.1	5.3	6.2	5.5

Note. SOM: self-organizing map.

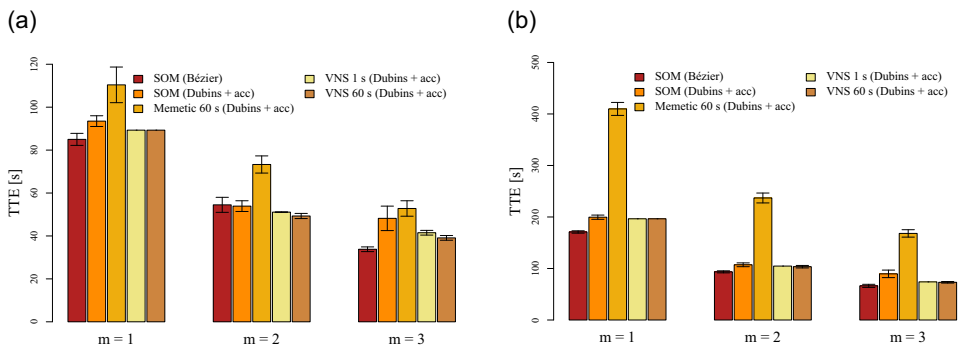


FIGURE 23 Average values of the TTE for large problems with n target locations. (a) m-DTSP instance with $n = 50$; (b) m-DTSP instance with $n = 100$. DTSP: Dubins traveling salesman problem; SOM: self-organizing map; TTE: travel time estimation; VNS: variable neighborhood search [Color figure can be viewed at wileyonlinelibrary.com]

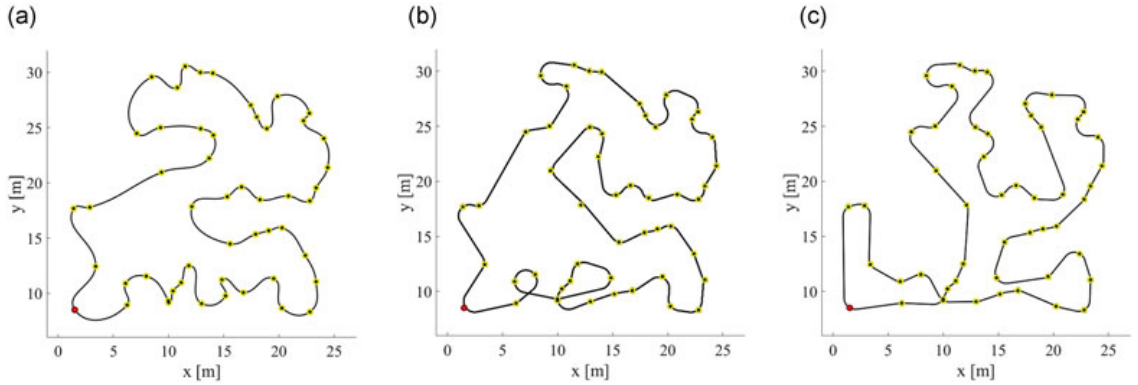


FIGURE 24 Selected found solutions for the problem with $n = 50$ target locations and one vehicle ($m = 1$). (a) SOM (Bézier), $TTE = 80.2$ s; (b) SOM (Dubins), $TTE = 87.6$ s; (c) VNS 60 s (Dubins), $TTE = 89.3$ s. SOM: self-organizing map; TTE: travel time estimation; VNS: variable neighborhood search [Color figure can be viewed at wileyonlinelibrary.com]

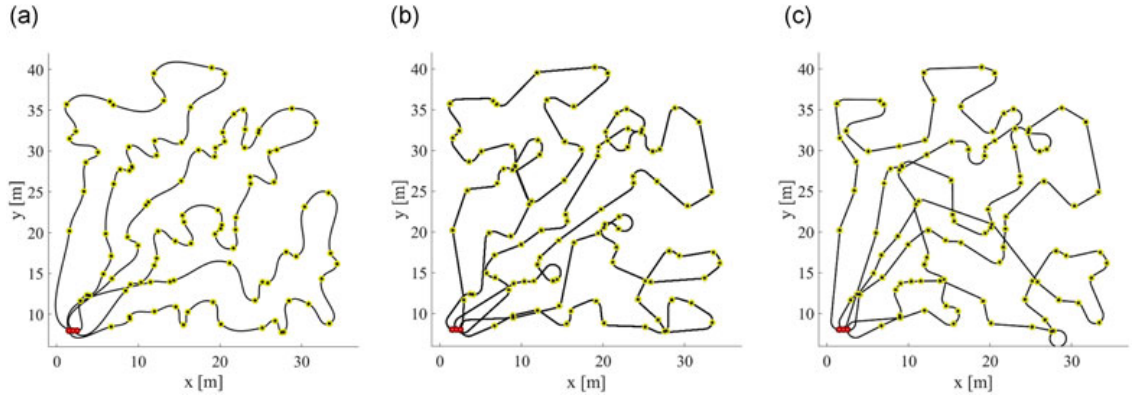


FIGURE 25 Selected best found solutions for the problem with $n = 100$ target locations and three vehicles ($m = 3$). (a) SOM (Bézier), $TTE = 55.2$ s; (b) SOM (Dubins), $TTE = 72.3$ s], (c) VNS 60 s (Dubins), $TTE = 70.8$ s. SOM: self-organizing map; VNS: variable neighborhood search [Color figure can be viewed at wileyonlinelibrary.com]

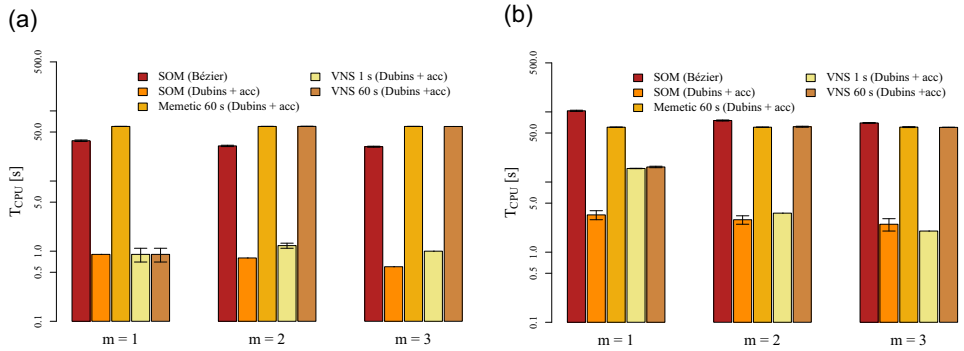


FIGURE 26 Average required computational time for large problems with n target locations. (a) m-DTSP instance with $n = 50$; (b) m-DTSP instance with $n = 100$. DTSP: Dubins traveling salesman problem; SOM: self-organizing map; VNS: variable neighborhood search [Color figure can be viewed at wileyonlinelibrary.com]

initialization used for the initial construction of a feasible solution prior a further improvement by the VNS optimization. A great benefit of the SOM solver is its flexibility to utilize Bézier curves that allow for exploiting motion capabilities of the used hexacopters which are not limited by a minimal turning radius ρ as the Dubins vehicle. Comparing parametrization of the requested surveillance trajectories

using Dubins vehicle model and the proposed sequence of Bézier curves, the main advantage of Dubins maneuvers is the closed-form solution for two waypoints with prescribed headings, which leads to lower computational requirements. On the other hand, the Bézier curves better fit the real limitations of the multirotor vehicles that are not limited by minimal turning radius, but by the maximal vehicle velocity and acceleration limits.

In small scenarios such as *mbzirc22*, it can be possible to find the best performing radius ρ for which Dubins maneuvers provide similar TTE as the Bézier curves, but only if velocity profiles are determined with the allowed acceleration/deceleration up to v_{horiz}. For larger instances, it may not be beneficial because of computational requirements for solving the *m*-DTSPN for several values of ρ would be similar or higher than a solution using the Bézier curves. Finally, the main advantage of the Bézier curves is a direct generalization of the surveillance planning to the 3D, which is not directly possible for the Dubins vehicle. The Dubins Airplane model (Chitsaz & LaValle, 2007, December) can be used for 3D trajectory planning; however, shorter and faster trajectories will be found using the proposed Bézier curves for multirotor vehicles that do not need to use the additional spiral to gain the requested altitude, as these vehicles can directly flight almost in any direction. Therefore, if the configuration of the planning problem is known in advance and enough time to compute several solutions is available, which is not the case of the robotic competition, it can be suitable to consider the Dubins vehicle model. In other cases and especially 3D planning with $\delta > 0$, the proposed SOM-based unsupervised learning framework with Bézier curves is a suitable choice.

The reported evaluation results are only for problem instances with up to three vehicles because of our motivation arising from the MBZIRC 2017 competition, where we deployed only three vehicles. All the proposed and evaluated algorithms for *m*-DTSPN including the SOM-based method for surveillance planning with Bézier curves can solve problems with a higher number of vehicles; however, we do not consider such scenarios because of the scope of this paper and challenging experimental verification, for example, with 10 vehicles, that needs significantly larger and more demanding experimental setup. Regarding scalability of the used SOM-based unsupervised learning, it is worth mentioning that it can be considered as independent on the number of vehicles if the number of target locations n is significantly higher than the number of vehicles m, that is, $n \gg m$ (see the analysis in Best et al. (2018)).

For multirobot deployment, an important part of the surveillance planning is collision avoidance. In the presented approach, we do not include the collision avoidance explicitly in the planning part because it is addressed in MPC-based controller used in the trajectory following which is considered to be out of the scope of this paper. The reactive collision avoidance based on the MPC predictions uses a slight alteration of the desired trajectory altitude if the MPC predictions contain collisions between the vehicles. The used MPC-based collision avoidance is partially described in Spurný et al. (2018) and it is presented in Báča et al. (2018). Besides, the found solutions and especially those found by the proposed unsupervised learning are such that the found trajectories are mutually noncrossing, and thus collision-free (see a discussion on that in Faigl (2016). Although noncrossing trajectories are not guaranteed; such solutions are found with a high probability in the considered scenarios also because of the selection of the vehicle depots that have to respect safety zones around each vehicle.

Regarding future work related to the proposed solvers, there are several open questions in addition to the explicit consideration of collision-free trajectories. One of them is that the proposed VNS initialization exhibits surprisingly good results and since it seems the VNS optimization scales poorly with increasing computational time, such an initial solution can be fed to the memetic algorithm for further improvement. For large instances with tens, hundreds, and more target locations, the unsupervised learning with Bézier curves seems to scale better than the VNS, and there are two ways how the computation can be further speeded up. The first is to improve the local optimization. The second is to exploit parallelization of the unsupervised learning of SOM, which has been already reported in the literature including SOM for the TSP.

The promising results of the Dubins vehicle model with various ρ accompanied with the computation of the velocity profile for hexacopters provide a source of motivation for generalization of the proposed approaches to consider multiple radii simultaneously during the optimization. In addition, a further generalization of the Dubins vehicle model used in the proposed solvers towards the 3D Airplane model or Dubins-Helix model is also a possible subject for the future work.

8 | CONCLUSION

In this paper, we address surveillance planning problem motivated by our participation in the robotic competition MBZIRC 2017. Because of the motivation, we aim to quickly find a solution to the planning problem with satisfiable quality, and thus we focus on the heuristic solution rather than optimal algorithms. The problem is first tackled as a variant of the *m*-DTSPN with the Dubins vehicle model for satisfying curvature-constrained trajectories that fit properties of the utilized trajectory follower. The *m*-DTSPN has been addressed by the proposed VNS-based and SOM-based algorithms that are significantly less demanding than the existing memetic algorithm, and both proposed algorithms provide better solutions in less computational time. However, Dubins vehicle model is suitable for vehicles with the limited turning radius, i.e., not necessarily the case of the used hexacopters whose motion is constrained by the maximal velocity and acceleration limits. Enabled by the flexibility of the used unsupervised learning, we propose to consider a more general trajectory parametrization based on Bézier curves, which enable to better exploit motion capabilities of the used vehicles. Moreover, it also allows solving 3D surveillance planning missions and finding 3D smooth trajectories for a team of our hexacopters. The solutions found by the

proposed algorithms have been numerically evaluated in several realistic problem instances. Besides, the solutions have also been experimentally verified by a real multirobotic system, where all the provided trajectories have been found feasible, and they fit the properties of the utilized trajectory following controller of the used hexacopters.

ACKNOWLEDGMENTS

The authors thank Vojtěch Spurný and Tomáš Báča for their assistance with the experimental deployment of the presented method with the real UAVs in the field and also to organizers of the MBZIRC competition. The presented work was supported by the Czech Science Foundation (GAČR) under research project No. 16-24206S. The authors acknowledge the support of the OP VVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics."

ORCID

Jan Faigl http://orcid.org/0000-0002-6193-0792
Petr Váňa http://orcid.org/0000-0003-2155-5788
Robert Pěnička http://orcid.org/0000-0001-8549-4932

REFERENCES

Ahmad, R., & Kim, D. (2015). An extended self-organizing map based on 2-opt algorithm for solving symmetrical traveling salesperson problem. *Neural Computing and Applications*, *26*(4), 987–994.

Angéniol, B., Vaubois, G. D. L. C., & Texier, J.-Y. L. (1988). Self-organizing feature maps and the travelling salesman problem. *Neural Networks*, *1*, 289–293.

Applegate, D., Bixby, R., Chvátal, V., & Cook, W. (2003). Concorde TSP Solver. Retrieved from http://www.tsp.gatech.edu/concorde.html (Accessed Aug 4, 2017).

Applegate, D., Bixby, R., Chvátal, V., & Cook, W. (2007). *The traveling salesman problem: A computational study*. Princeton, NJ, USA: Princeton University Press.

Báča, T., Hert, D., Loianno, G., Saska, M., & Kumar, V. (2018, October). Model predictive trajectory tracking and collision avoidance for reliable outdoor deployment of unmanned aerial vehicles. IEEE/RSJ Conference on Intelligent Robots and Systems (IROS), Madrid, Spain.

Báča, T., Loianno, G., & Saska, M. (2016, August-September). Embedded model predictive control of unmanned micro aerial vehicles. 21st International Conference on Methods and Models in Automation and Robotics (MMAR), 992–997. Miedzyzdroje, Poland.

Bektas, T. (2006). The multiple traveling salesman problem: An overview of formulations and solution procedures. *Omega*, *34*(3), 209–219.

Bellmore, M., & Hong, S. (1974). Transformation of multisalesman problem to the standard traveling salesman problem. *Journal of the ACM*, *21*(3), 500–504.

Best, G., Faigl, J., & Fitch, R. (2018). Online planning for multirobot active perception with self-organising maps. *Autonomous Robots*, *42*(4), 715–736.

Bézier, P. (1973). Numerical control: Mathematics and applications. *International Journal for Numerical Methods in Engineering*, *6*(3), 456–456.

Chitsaz, H., & LaValle, S. M. (2007, December). Time-optimal paths for a dubins airplane. 46th IEEE Conference on Decision and Control (CDC), 2379–2384. New Orleans, LA.

Cochrane, E. M., & Beasley, J. E. (2003). The co-adaptive neural network approach to the Euclidean travelling salesman problem. *Neural Networks*, *16*(10), 1499–1525.

Croes, G. A. (1958). A method for solving traveling-salesman problems. *Operations Research*, *6*(6), 791–812.

Dubins, L. E. (1957). On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics*, *2016*, 497–516.

Faigl, J. (2016). An application of self-organizing map for multirobot multigoal path planning with minmax objective. *Computational Intelligence and Neuroscience*, *2016*, 15. https://doi.org/10.1155/2016/2720630

Faigl, J. (2018). GSOA: Growing self-organizing array - unsupervised learning for the close-enough traveling salesman problem and other routing problems. *Neurocomputing*, *312*, 120–134.

Faigl, J., & Hollinger, G. A. (2018). Autonomous data collection using a self-organizing map. *IEEE Transactions on Neural Networks and Learning Systems*, *29*(5), 1703–1715.

Faigl, J., & Váňa, P. (2016, September). Self-organizing map for the curvature-constrained traveling salesman problem.International Conference on Artificial Neural Networks, 497–505. Barcelona, Spain.

Faigl, J., & Váňa, P. (2017, May). Unsupervised learning for surveillance planning with team of aerial vehicles. International Joint Conference on Neural Networks (IJCNN), 4340–4347. Anchorage, AK.

Faigl, J., Váňa, P., Saska, M., Báča, T., & Spurný, V. (2017, September). On solution of the dubins touring problem. European Conference on Mobile Robots (ECMR). Paris, France.

Fort, J. C. (1988). Solving a combinatorial problem via self-organizing process: An application of the Kohonen algorithm to the traveling salesman problem. *Biological Cybernetics*, *59*(1), 33–40.

França, P. M., Gendreau, M., Laporte, G., & Müller, F. M. (1995). The m-traveling salesman problem with minmax objective. *Transportation Science*, *29*(3), 267–275.

Goaoc, X., Kim, H.-S., & Lazard, S. (2013). Bounded-curvature shortest paths through a sequence of points using convex optimization. *SIAM Journal on Computing*, *42*(2), 662–684.

Hansen, P., & Mladenović, N. (2001). Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, *130*(3), 449–467.

Helsgaun, K. (2000). An effective implementation of the lin-kernighan traveling salesman heuristic. *European Journal of Operational Research*, *126*(1), 106–130.

Isaacs, J. T., & Hespanha, J. P. (2013). Dubins traveling salesman problem with neighborhoods: A graph-based approach. *Algorithms*, *6*(1), 84–99.

Isaacs, J. T., Klein, D. J., & Hespanha, J. P. (2011). Algorithms for the traveling salesman problem with neighborhoods involving a Dubins vehicle. American Control Conference, 1704–1709.

Isaiah, P., & Shima, T. (2015). Motion planning algorithms for the Dubins travelling salesperson problem. *Automatica*, *53*, 247–255.

Jolly, K., Sreerama Kumar, R., & Vijayakumar, R. (2009). A Bezier curve based path planning in a multi-agent robot soccer system without violating the acceleration limits. *Robotics and Autonomous Systems*, *57*(1), 23–33.

Kohonen, T., Schroeder, M. R., & Huang, T. S. (Eds.), (2001). *Self-organizing maps*. New York: Springer-Verlag, Inc.

Kulich, M., Faigl, J., & Kléma, J., Kubalík, J., (2004, August). Rescue operation planning by soft computing techniques. IEEE 4th International Conference on Intelligent Systems Design and Application, 103–108. Budapest, Hungary.

Le Ny, J., Feron, E., & Frazzoli, E. (2012). On the Dubins traveling salesman problem. *IEEE Transactions on Automatic Control*, *57*(1), 265–270.

Lepetič, M., Klančar, G., Škrjanc, I., Matko, D., & Potočnik, B. (2003). Time optimal path planning considering acceleration limits. *Robotics and Autonomous Systems*, *45*(3), 199–210.

Ma, X., & Castanon, D. A. (2006, December). Receding horizon planning for Dubins traveling salesman problems. 45th IEEE Conference on Decision and Control, 5453–5458. San Diego, CA.

Macharet, D. G., & Campos, M. M. (2014). An orientation assignment heuristic to the dubins traveling salesman problem. In A. Bazzan, & K. Pichara (Eds.), *Advances in artificial intelligence-IBERAMIA 2014*, Lecture notes in computer science (Vol 8864, pp. 457–468). Cham: Springer.

Macharet, D. G., Monteiro, J. W., Mateus, G. R., & Campos, M. M. (2016, October). Time-optimized routing problem for vehicles with bounded curvature. XIII Latin American Robotics Symposium and IV Brazilian Robotics Symposium, 145–150. Recife, Brazil.

Macharet, D. G., Neto, A. A., da Camara Neto, V. F., & Campos, M. M. (2011, May). Nonholonomic path planning optimization for dubins' vehicles.IEEE International Conference on Robotics and Automation (ICRA), 4208–4213. Shanghai, China.

Macharet, D. G., Neto, A. A., da Camara Neto, V. F., & Campos, M. M. (2012, July). An evolutionary approach for the Dubins' traveling salesman problem with neighborhoods. 14th Annual Conference on Genetic and Evolutionary Computation, 377–384. Philadelphia, PA.

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

Manyam, S., & Rathinam, S. (2015). On tightly bounding the Dubins traveling salesman's optimum. *Journal of Dynamic Systems, Measurement, and Control*, *140*(7), 071013.

Manyam, S., Rathinam, S., & Casbeer, D. (2016, June). Dubins paths through a sequence of points: Lower and upper bounds. International Conference on Unmanned Aircraft Systems (ICUAS), 284–291. Arlington, VA.

Manyam, S., Rathinam, S., Casbeer, D., & Garcia, E. (2015). Shortest paths of bounded curvature for the dubins interval problem. arXiv: https://arxiv.org/abs/1507.06980

MBZIRC (2017). Mohamed Bin Zayed International Robotics Challenge (MBZIRC). Retrieved from http://www.mbzirc.com (accessed July 28, 2017).

Neubauer, M., & Müller, A. (2015, September-October). Smooth orientation path planning with quaternions using B-splines. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2087–2092. Hamburg, Germany.

Ny, J., Feron, E., & Frazzoli, E. (2012). On the dubins traveling salesman problem. *IEEE Transactions on Automatic Control*, *57*(1), 265–270.

Oberlin, P., Rathinam, S., & Darbha, S. (2010). Today's traveling salesman problem. *Robotics and Automation Magazine, IEEE*, *17*(4), 70–77.

Obermeyer, K. (2009, August). Path planning for a uav performing reconnaissance of static ground targets in terrain. AIAA Guidance, Navigation, and Control Conference, 10–13. Chicago, IL.

Obermeyer, K., Oberlin, P., & Darbha, S. (2010, August). Sampling-based roadmap methods for a visual reconnaissance UAV*. AIAA Guidance, Navigation, and Control Conference. Toronto, Ontario, Canada.

Obermeyer, K., Oberlin, P., & Darbha, S. (2012). Sampling-based path planning for a visual reconnaissance unmanned air vehicle. *Journal of Guidance, Control, and Dynamics*, *35*(2), 619–631.

Papadopoulos, E., Papadimitriou, I., & Poulakakis, I. (2005). Polynomial-based obstacle avoidance techniques for nonholonomic mobile manipulator systems. *Robotics and Autonomous Systems*, *51*(4), 229–247.

Pěnička, R., Faigl, J., Váňa, P., & Saska, M. (2017a). Dubins orienteering problem. *IEEE Robotics and Automation Letters*, *2*(2), 1210–1217.

Pěnička, R., Faigl, J., Váňa, P., & Saska, M. (2017b, June). Dubins orienteering problem with neighborhoods. International Conference on Unmanned Aircraft Systems (ICUAS), 1555–1562. Miami, FL.

Saska, M. (2017). MBZIRC team of the Czech Technical University. Retrieved from http://mrs.felk.cvut.cz/projects/mbzirc (a). Accessed August 4, 2017.

Savla, K., Frazzoli, E., & Bullo, F. (2005, June). On the point-to-point and traveling salesperson problems for Dubins' vehicle. American Control Conference, 786–791. Portland, OR.

Somhom, S., Modares, A., & Enkawa, T. (1997). A self-organising model for the travelling salesman problem. *Journal of the Operational Research Society*, *48*, 919–928.

Somhom, S., Modares, A., & Enkawa, T. (1999). Competition-based neural network for the multiple travelling salesmen problem with minmax objective. *Computers and Operations Research*, *26*(4), 395–407.

Soylu, B. (2015). A general variable neighborhood search heuristic for multiple traveling salesmen problem. *Computers and Industrial Engineering*, *90*, 390–401.

Spurný, V., Báča, T., Saska, M., Pěnička, R., Thomas, J., Thakur, D., ..., Kumar, V. (2018). Cooperative autonomous search, grasping, and delivering in a treasure hunt scenario by a team of unmanned aerial vehicles. *Journal of Field Robotics*. 1–24. https://doi.org/10.1002/rob.2181

Váňa, P., & Faigl, J. (2015, September-October). On the Dubins traveling salesman problem with neighborhoods. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 4029–4034. Hamburg, Germany.

Váňa, P., Sláma, J., & Faigl, J. (2018, May). The Dubins traveling salesman problem with neighborhoods in the three-dimensional space. IEEE International Conference on Robotics and Automation (ICRA), 374–379. Brisbane, Australia.

Wang, Y., Wang, S., & Tan, M. (2015a). Path generation of autonomous approach to a moving ship for unmanned vehicles. *IEEE Transactions on Industrial Electronics*, *62*(9), 5619–5629.

Wang, Y., Wang, S., Tan, M., Zhou, C., & Wei, Q. (2015b). Real-time dynamic Dubins-Helix method for 3-D trajectory smoothing. *IEEE Transactions on Control Systems Technology*, *23*(2), 730–736.

Yang, K., & Sukkarieh, S. (2010). An analytical continuous-curvature path-smoothing algorithm. *IEEE Transactions on Robotics*, *26*(3), 561–568.

Yu, X. (2015). Optimization approaches for a Dubins vehicle in coverage planning problem and traveling salesman problems. PhD thesis, Auburn University.

Yu, X., & Hung, J. (2012). A genetic algorithm for the dubins traveling salesman problem. IEEE International Symposium on Industrial Electronics, 1256–1261.

Zhang, X., Chen, J., Xin, B., & Peng, Z. (2014). A memetic algorithm for path planning of curvature-constrained uavs performing surveillance of multiple ground targets. *Chinese Journal of Aeronautics*, *27*(3), 622–633.

How to cite this article: Faigl J, Váňa P, Pěnička R, Saska M. Unsupervised learning-based flexible framework for surveillance planning with aerial vehicles. *J Field Robotics*. 2019;36:270–301. https://doi.org/10.1002/rob.21823

Chapter 7

Physical Orienteering Problem for Unmanned Aerial Vehicle Data Collection Planning in Environments with Obstacles

The last core publication of this thesis is the article [5c] published in the IEEE Robotics and Automation Letters concerning Physical Orienteering Problem (POP).

[5c] **R. Pěnička**, J. Faigl, and M. Saska, "Physical orienteering problem for unmanned aerial vehicle data collection planning in environments with obstacles," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 3005–3012, 2019

The POP is an extension of the OP with the ordinary maximization of the collected reward within a limited budget; however, for environments with obstacles. The problem thus combines the collision-free motion planning in configuration space [27] with the OP routing over multiple target locations. The task of collision-free motion planning is to find feasible paths between individual pairs of target locations in the environment with obstacles. The POP objective is then to select a subset of the target locations and a sequence to visit them within the prescribed budget using the collision-free motion plans found so-far. However, the cost of so-far found motion plans highly influence the final solution quality through the limited budget. Therefore, the routing (subset and sequence selection) and motion planning have to be addressed simultaneously. The POP can be understood as the OP in configuration space.

The proposed method for the POP combines the VNS-based method with asymptotically optimal sampling-based Probabilistic Roadmap (PRM*) [89] method in a tightly coupled algorithm denoted as the VNS-PRM*. The VNS part for the routing uses similar *shaking* and *local search* procedures as presented for the first three core publications [1c]–[3c]. However, the *local search* procedure is additionally used to search the combinatorial routing part of the POP for high-quality and possibly over-budget solutions that can be shortened. The collision-free motion plans in such identified solutions can be then shortened by additional PRM* sampling between individual target locations. The PRM* part of the algorithm uses initially a low-dense sampled roadmap. The roadmap is then continuously expanded in every iteration of the VNS-PRM* based on the *local search* combinatorial search. The sampling strategy prefers expansion for shortening solutions with small budget overshoot and with high reward, possibly better than the so-far best found. Such expansion can shorten the motion plans between the targets in the roadmap and thus allows the VNS combination part to find higher rewarded solutions using the same budget limit.

The proposed VNS-PRM* method is compared with the optimal solution of the POP for point robot in the 2D environment found as a combination of the ILP solution of the OP on visibility graph roadmap. The VNS-PRM* is shown to find the optimal solutions for the majority of the tested instances. The comparison with the existing method for the PCTSP [95] shows that the VNS-PRM* significantly outperforms the method for the tested Dubins vehicle model. The method is finally shown when employed in a 3D environment and also in a real outdoor experiment with hexarotor UAV with a non-point robot model.

The author's contribution on this article is 70 %, including writing the manuscript and implementing the method. The co-authors contributed by giving valuable feedback.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/LRA.2019.2923949

IEEE ROBOTICS AND AUTOMATION LETTERS. PREPRINT VERSION. ACCEPTED JUNE, 2019 1

Physical Orienteering Problem for Unmanned Aerial Vehicle Data Collection Planning in Environments with Obstacles

Robert Pěnička, Jan Faigl, and Martin Saska

***Abstract*—This paper concerns a variant of the Orienteering Problem (OP) that arises from multi-goal data collection scenarios where a robot with a limited travel budget is requested to visit given target locations in an environment with obstacles. We call the introduced OP variant the Physical Orienteering Problem (POP). The POP sets out to determine a feasible, collision-free, path that maximizes collected reward from a subset of the target locations and does not exceed the given travel budget. The problem combines motion planning and combinatorial optimization to visit multiple target locations. The proposed solution to the POP is based on the Variable Neighborhood Search (VNS) method combined with the asymptotically optimal sampling-based Probabilistic Roadmap (PRM*) method. The VNS-PRM* uses initial low-dense roadmap that is continuously expanded during the VNS-based POP optimization to shorten paths of the promising solutions, and thus allows maximizing the sum of the collected rewards. The computational results support the feasibility of the proposed approach by a fast determination of high-quality solutions. Moreover, an experimental verification demonstrates the applicability of the proposed VNS-PRM* approach for data collection planning for an unmanned aerial vehicle in an urban-like environment with obstacles.**

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

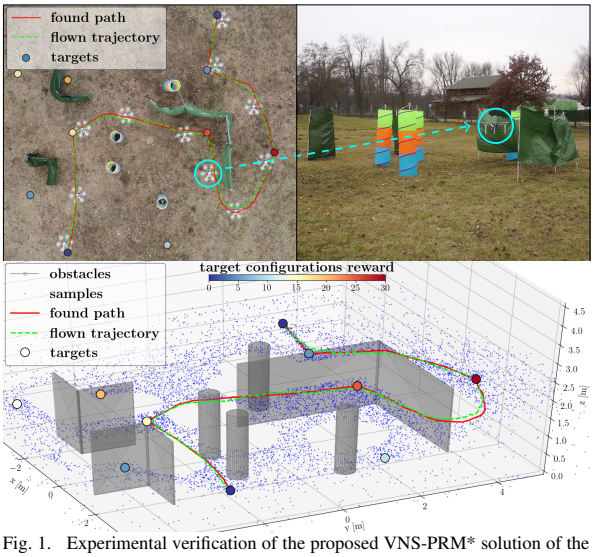


Fig. 1. Experimental verification of the proposed VNS-PRM* solution of the introduced Physical Orienteering Problem (POP) in outdoor data collection scenario with obstacles. We refer to https://youtu.be/xUXYEt4Gnvk for the video from the experimental verification.

I. Introduction

In this paper, we study a generalization of the Orienteering Problem (OP) [1] to address robotic route planning problems in environments with obstacles and with an arbitrary motion model of the used vehicle. The introduced problem is called the Physical Orienteering Problem (POP), and it can be considered as the OP explicitly deployed in the configuration space [2] where both the obstacles and vehicle motion constraints can be addressed. The OP belongs to multi-goal routing problems with profits where each target location has associated reward, and the problem sets out to maximize the sum of collected rewards without exceeding the specified travel budget. The POP stands, for the given initial and terminal locations, to select a subset of the locations and a sequence to visit them together with the determination of cost-efficient and collision-free paths between the individual locations to maximize the sum of collected rewards by saving vehicle travels to fit the budget. Hence, the route and path planning needs to be addressed in a single optimization problem to find a high-quality solution of the POP.

The motivation for the introduced problem is in data collection missions with Unmanned Aerial Vehicles (UAVs) in indoor and urban-like environments where multiple target locations need to be visited for collecting the requested data. Such a mission can be, e.g., to collect desired measurements at the particular locations of interest using UAV equipped with an onboard camera. Another example can be found in wireless sensor networks [3] where a UAV can be used to collect data from the sensors placed in the environment.

The flight time of today's UAVs is usually limited and visiting all target locations can be unfeasible, and therefore, each location can be assigned with a reward to prioritize the most important locations. The existing Euclidean OP [4] or its extension for Dubins vehicle [5] can be used to find the data collection plan. However, in a realistic robotic scenario, the operational environment can contain obstacles and motion

Manuscript received: February, 24, 2019; Revised May, 14, 2019; Accepted June, 4, 2019.

This paper was recommended for publication by Editor Nancy Amato upon evaluation of the Associate Editor and Reviewers' comments. The presented work has been supported by the Czech Science Foundation under research project No. 19-20238S and No. 17-16900Y. The authors acknowledge the support of the OP VVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics". The support of the CTU in Prague grant SGS17/187/13 is greatly appreciated.

Authors are with the Czech Technical University, Faculty of Electrical Engineering, Technicka 2, 166 27, Prague, Czech Republic {penicrob|faiglj|saskam1}@fel.cvut.cz

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

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/LRA.2019.2923949

constraints of the utilized vehicles can be more complicated than the Dubins vehicle. Therefore, the POP is introduced to allow deploying budget-limited UAVs in realistic data collection missions in environments with obstacles, see a snapshot of the experimental deployment in Fig. 1.

The proposed solution of the introduced POP is denoted VNS-PRM* because it is based on the Variable Neighborhood Search (VNS) [6] metaheuristic tightly coupled with the asymptotically optimal sampling-based Probabilistic Roadmaps method (PRM*) [7]. The VNS-based combinatorial optimization searches for the rewarding subset of the target locations and order to visit them such that the interconnecting paths are further shortened by the PRM*. The PRM* is employed to create an initial low-dense roadmap of collision-free paths between the target locations. The initial roadmap is then incrementally expanded during the iterative optimization of the OP to improve paths of promising OP solutions. In this way, the VNS-PRM* simultaneously searches both the OP solution and configuration spaces.

The rest of the paper is organized as follows. An overview of related work is summarized in the next section. Section III formally introduces the POP and the proposed VNS-PRM* is described in Section IV. Evaluation results are reported in Section V and concluding remarks are outlined in Section VI.

II. Related Work

The addressed POP combines motion planning and the OP, and thus we briefly overview the relevant sampling-based planning approaches, the most related OP methods, and also existing approaches combining routing with motion planning.

The Rapidly-Exploring Random Trees (RRT) [8] and the Probabilistic Roadmaps (PRM) [9] can be considered as the most fundamental approaches with many modifications and variants [2]. Regarding the addressed POP, the RRT* and PRM* [7] are considered as the most relevant approaches, albeit other methods with path optimality criteria can be utilized [10]. The introduced POP is a multi-goal planning problem which requires a multi-query search, and thus the PRM* is a suitable technique for the proposed solution.

The OP belongs to routing problems with profits, and it has been introduced by Tsiligirides [4] in 1984. Since then, numerous algorithmic solutions have been proposed [1] together with a wide range of formulation variants [11]. The OP can be defined as an Integer Linear Programming (ILP) problem [1] and solved by Branch-and-Bound [12] or Branch-and-Cut [13] algorithms. Existing heuristics, e.g., particle swarm optimization [14] or ant colony optimization [15], provide solutions of similar quality but within a fraction of time required for finding the optimal solution. In particular, the VNS-based [16] solution of the OP performs as one of the best considering the computational time and solution quality, and therefore, the proposed solution builds on the VNS-based optimization operators.

The POP is also related to variants of the OP where the travel cost is not a length of the straight lines connecting the locations as in the regular Euclidean OP. The Dubins Orienteering Problem (DOP) [5] is an extension for Dubins vehicle [17] that requires to optimize the heading angle of the vehicle to find the most rewarding paths. The proposed VNS-PRM* significantly extends the VNS-based method for the DOP [5] by considering the OP in the configuration space with obstacles addressed by tightly coupled PRM* with online roadmap expansion. Besides, the DOP has been used for UAVs in wildfire observation planning [18] and further extended to the DOP with Neighborhoods (DOPN) addressed by the VNS [19] and unsupervised learning [20]. To the best of the authors' knowledge, the only OP variant considering the environments with obstacles is the approach presented in [21]. The method is based on a low level A* search in a grid of Dubins maneuvers to get around obstacles, which limits its application to instances without narrow passages and predefined heading angles. On the other hand, the proposed VNS-PRM* employs sampling-based motion planning that can be used to find collision-free paths also in the 3D with various vehicle motion constraints.

The motion planning combined with routing has been mostly studied in the context of the Traveling Salesman Problem (TSP) where the Physical TSP (PTSP) [22] combines TSP with real-time motion planning in video games. In robotics, a multi-tree Transition-based RRT [23] has been proposed for creating a collision-free roadmap for arbitrary routing problem. Several existing approaches combining the routing problems with motion planning have been introduced for scenarios with Autonomous Underwater Vehicles (AUV), e.g., planning mine countermeasures missions based on the PTSP [24], the Clustered TSP [25] and high-level mission planning [26] combined with motion planning.

The most similar existing problem to the POP is a variant of the Prize Collection Traveling Salesman Problem (PC-TSP) for AUV [27] that uses sampling-based methods for finding collision-free PC-TSP plans. The approach uses initially created PRM navigation roadmap for guiding a sampling-based motion tree considering the vehicle dynamics. A separate PC-TSP solver is used to prioritize the expansion of the motion tree along PC-TSP solutions found on the static navigation roadmap. Contrarily, the proposed VNS-PRM* uses tightly coupled asymptotically optimal PRM*, where vehicle dynamics is considered by different motion primitives, with the VNS-based OP solver within a single optimization algorithm that deals with narrow passages better than the decoupled approach of [27], as shown in Section V.

III. Problem Statement

The proposed Physical Orienteering Problem (POP) combines collision-free path planning with the combinatorial routing of the Orienteering Problem in a single optimization problem. Therefore, we outline the path planning first; then the POP is introduced as an extension of the regular OP with path planning to determine the most rewarding path that does not exceed the given travel budget T_{max}.

Having the world $\mathcal{W} = \mathbb{R}^2$ or $\mathcal{W} = \mathbb{R}^3$ with the obstacles $\mathcal{O} = \{\mathcal{O}_1, \ldots, \mathcal{O}_m\} \subset \mathcal{W}$, the point-to-point path planning problem is to determine a collision free-path for a robot $\mathcal{A} \subset \mathcal{W}$ between two locations in $\mathcal{W}$ such that the path

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/LRA.2019.2923949

avoids $\mathcal{O}$. The problem can be formulated using the notion of the configuration space $\mathcal{C}$ [2], which consists of all possible robot configurations $q \in \mathcal{C}$. Let $\mathcal{A}(q) \subset \mathcal{W}$ denotes geometry of the robot at a configuration q. The robot can move in the free space $\mathcal{C}_{free} = \mathcal{C} \setminus \mathcal{C}_{obs}$, where $\mathcal{C}_{obs} = \{q \in \mathcal{C} | \mathcal{A}(q) \cap \mathcal{O} \neq \emptyset\} \subseteq \mathcal{C}$ is a set of configurations where the robot $\mathcal{A}(q)$ collides with $\mathcal{O}$. A solution of the point-to-point path planing between initial $q_I \in \mathcal{C}_{free}$ and goal $q_G \in \mathcal{C}_{free}$ configurations is a path $\tau : [0,1] \rightarrow \mathcal{C}_{free}$ with $\tau(0) = q_I$ and $\tau(1) = q_G$, respectively. The cost to travel the path τ can be expressed as a cost function $c(\tau) \rightarrow \mathbb{R}_{\geq 0}$. In this work, w.l.o.g. we consider the cost to be the length of the path, i.e., $c(\tau) = \int_0^1 |\tau(t)| dt$. In addition to have a feasible path τ that avoids obstacles $\tau \in \mathcal{C}_{free}$, we are searching for the optimal path τ^* such that $c(\tau^*) = \min\{c(\tau) | \tau \textit{ is feasible}\}$ to find a solution of the introduced POP. Moreover, a single point-to-point collision-free optimal path planning is only a part of the POP, as we need to address finding a path over multiple target locations that can be arbitrarily ordered as in the solution of the OP.

The OP belongs to a class of routing problems with profits where each of all n predefined target locations $S = \{s_1, \ldots, s_n\}$, $s_i \in \mathcal{W}$ have associated reward $r_i \geq 0$ for $i \in \{1, \ldots, n\}$. The OP stands to maximize the sum of the collected rewards R by visiting a subset $S_l \subseteq S$ such that the path to visit S_l does not exceed the given limited travel budget $\mathrm{T_{max}}$. The initial and terminal locations of the path are prescribed and for simplicity they are denoted s_1 and s_n, respectively, both with the zero reward $r_0 = r_n = 0$. The OP is an optimization problem to find the subset S_l of l target locations together with a sequence to visit the target locations in S_l within $\mathrm{T_{max}}$. The sequence can be expressed as a permutation of the target location indexes $\Sigma = (\sigma_1, \ldots, \sigma_l)$ with $1 \leq \sigma_i \leq n$, $\sigma_i \neq \sigma_j$ for $i \neq j$, $\sigma_1 = 1$ and $\sigma_l = n$, because of the prescribed initial and terminal locations. The locations in the sequence have to be connected by a collision-free path not exceeding $\mathrm{T_{max}}$, and thus we need to combine routing and path planning for a solution of the introduced POP.

In the POP, the target locations $s_i \in \mathcal{W}$ of the OP correspond to the target configurations $Q = \{q_1, \ldots, q_n\}, q_i \in \mathcal{C}_{free}$, such that $s_i \in \mathcal{A}(q_i)$ for all $1 \leq i \leq n$. A solution of the POP is a sequence Σ of the configurations $Q_l \subseteq Q$ that maximizes R using collision-free paths with the sum of the cost satisfying $\mathrm{T_{max}}$. We propose to combine the solution of the combinatorial OP with path planning to determine paths τ_i connecting locations S_l in the sequence Σ such that the individual paths are feasibly connected at the target configurations $\tau_i(0) = q_{\sigma_i}$ and $\tau_i(1) = q_{\sigma_{i+1}}$ for $1 \leq i \leq l-1$. Besides, the total path length is limited by the travel budget $\sum_{i=1}^{l-1} c(\tau_i) \leq \mathrm{T_{max}}$. The POP can be understood as the OP in $\mathcal{C}$ and can be summarized in a single optimization problem (1).

$$\begin{aligned} \underset{l, Q_l, \Sigma, \tau_i}{\text{maximize}} \;\; & R = \sum_{i=1}^{l} r_{\sigma_i} \\ \text{s.t.} \;\; & \sum_{i=1}^{l-1} c(\tau_i) \leq \mathrm{T_{max}}, \\ & \sigma_1 = 1,\, \sigma_l = n,\, \tau_i \in \mathcal{C}_{free}, \\ & \tau_i(0) = q_{\sigma_i},\, \tau_i(1) = q_{\sigma_{i+1}},\, i = 1 \ldots l-1 \end{aligned} \tag{1}$$

The POP objective is to maximize the sum of the collected rewards R by visiting the target configurations Q_l. However, the budget limit $\mathrm{T_{max}}$ requires to evaluate the cost of the path to visit Q_l, and thus it requires to find the appropriate sequence Σ of the configurations together with collision-free paths connecting the configurations in the sequence. Finding the collision-free paths is a challenging problem and determining all possible paths connecting all the locations S is computationally very demanding. Moreover, optimal paths should be determined to ensure $\mathrm{T_{max}}$ while visiting as many highly rewarding locations as possible, which is even more computationally demanding. On the other hand, it is likely that a subset Q_l contains only a small portion of Q, and thus determining all paths is not necessary. Therefore, we propose to address the introduced POP by a combination of the asymptotically optimal motion planner PRM* with the VNS-based solution of the routing part of the POP to continuously improve the PRM* roadmap using the combinatorial solutions to expand the roadmap only in parts of $\mathcal{C}$ that can contribute to the solution of the POP.

IV. Proposed VNS-PRM* method for the POP

The proposed approach to solve the POP combines asymptotically optimal sampling-based PRM* [7] with the combinatorial metaheuristic VNS [6] to solve the OP on the incrementally constructed roadmap. The POP is addressed by a single VNS-based algorithm with an online improvement of the roadmap using PRM* to support finding collision-free trajectories in the configuration space to visit multiple target configurations. The selection of the target locations and sequence to visit them to maximize the sum of the collected rewards is thus optimized together with the paths connecting the selected target locations. The proposed VNS-PRM* combines both feedbacks from (i) the PRM* for finding the OP solution (i.e., the selection and sequence of targets) on the improving roadmap; (ii) the search space of the OP to guide the PRM* sampling of $\mathcal{C}_{free}$.

A. PRM* for the Physical Orienteering Problem

The PRM* is a multi-query asymptotically optimal motion planning algorithm that firstly randomly samples configurations in $\mathcal{C}_{free}$ and creates a graph $G = (V, E)$ (further denoted as the roadmap) by connecting k neighboring samples with a collision-free path. Contrary to the ordinary PRM with a fixed k, in the employed k-nearest PRM*, the value of k increases with the number of vertices m in G as $k(m) = k_{PRM} \log(m)$ where $k_{PRM} > k^*_{PRM} = e(1 + 1/d)$ and d is the dimension of $\mathcal{C}$ [7]. Hence, the VNS-PRM* uses a low-dense initial roadmap consisting of m_{init} random configurations and the target configurations Q. Dijkstra's algorithm is then used to interconnect the target configurations using shortest paths in G between all configurations $q_i, q_j \in Q$ with the respective lengths $c_{i,j} = c(\tau), \tau(0) = q_i, \tau(1) = q_j$.

The maximization of the collected rewards needs minimal path lengths $c_{i,j}$ to visit valuable target configurations within $\mathrm{T_{max}}$. High-quality paths require a large number of samples m_{init}, where most of the samples would not be used for

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/LRA.2019.2923949

the paths of the final solution of the POP. Therefore, the roadmap is continuously expanded during the VNS-based optimization with the focused sampling on the paths between promising target configurations, i.e., configurations that are in highly rewarded POP solutions found so far. The roadmap initialization and expansion are summarized in Alg. 1.

Algorithm 1: PRM* Initialization and Expansion

In/Out: $G(V, E)$ – existing roadmap, Q – target configurations, $\mathcal{P} = \{\rho_{i,j}\}_{i,j=1,\dots,n}$ – sampling density
Input : $M = \{m_{i,j}\}$ number of samples to add between q_i, q_j

1 $V_{new} \leftarrow \emptyset$; $E_{new} \leftarrow \emptyset$
2 **if** q_1 q_n *not connectable in* G **then** `// roadmap initialization`
3 $V \leftarrow \emptyset$; $E \leftarrow \emptyset$
4 $V_{new} \leftarrow Q \cup \{\text{UniformSample } \mathcal{C}_{free}\}_{1,\dots,m_{init}}$
5 **else** `// roadmap expansion`
6 **foreach** *pair* $i, j \in (1, \dots, n), i \neq j$ **do**
7 $V_{new} \leftarrow V_{new} \cup \{\text{EllipsoidSample}(q_i, q_j, c_{i,j})\}_{1,\dots,m_{i,j}}$
8 $\rho_{i,j} \leftarrow \rho_{i,j} + m_{i,j}/\text{EllipsoidVolume}(q_i, q_j, c_{i,j})$
9 **foreach** $v \in V_{new}$ **do**
10 $U \leftarrow \text{kNearest}((V \cup V_{new}, E), v, k(|V| + |V_{new}|)) \setminus v$
11 **foreach** $u \in U$ **do**
12 **if** *CollisionFree*(v, u) **then**
13 $E_{new} \leftarrow E_{new} \cup \{(v, u)\}$
14 $V \leftarrow V \cup V_{new}$; $E \leftarrow E \cup E_{new}$

During the roadmap expansion, new random configurations V_{new} are sampled in the hyperellipsoids (Line 7, Alg. 1) corresponding to all target configurations pairs. The hyperellipsoids are defined by their foci in the respective target configurations q_i and q_j, and by the major axis length equal to the actual shortest path length $c_{i,j}$ between the corresponding target configurations. An individual hyperellipsoid between q_i and q_j is equidistantly sampled for $m_{i,j}$ times [28]. The value of $m_{i,j}$ thus defines a priority in which particular path is optimized, and it is updated at each iteration of the VNS-based solution of the POP. The sampling densities of particular ellipsoids $\rho_{i,j}$ are stored and further used to prioritize sampling of low-density sampled ellipsoids.

B. VNS-based method for the POP

The VNS is based on the two main procedures called *shake* and *local search* to iteratively improve a single incumbent solution. The *shake* procedure performs a random change of the currently best-found solution v to leave a possible local optimum. The *local search* optimizes a randomly changed solution v' using a set of neighborhoods (described as operators) to increase the quality of the incumbent solution.

In the VNS for the POP, a solution is represented as a vector $v = (q_{\sigma_1}, \dots, q_{\sigma_l}, \dots, q_{\sigma_n})$ of all target configurations Q, where the first l items $(q_{\sigma_1}, \dots, q_{\sigma_l})$ represent a path within T_{max} and the remaining part of v gathers the unvisited target configurations. The initial and terminal configurations are prescribed, and thus q_{σ_1} is always q_1 and q_{σ_l} is q_n. The operators of the *shake* and *local search* procedures change the order of target configurations in v to maximize the sum of the collected rewards $R(v) = R(v(l, Q_l, \Sigma)) = \sum_{i=1}^{l} r_{\sigma_i}$ while keeping the path length $\mathcal{L}(v) = \mathcal{L}(v(l, Q_l, \Sigma)) = \sum_{i=1}^{l-1} c_{\sigma_i, \sigma_{i+1}}$ within T_{max} by moving q_{σ_l} inside v. Thus, the operators change not only the sequence Σ but also the subset of the visited target configurations Q_l. The path of a solution v is found as the shortest path in the roadmap over the sequence of targets $(q_{\sigma_1}, \dots, q_{\sigma_l})$.

The proposed VNS-PRM* is summarized in Alg. 2. The algorithm starts with PRM*initialSampling() that uniformly samples $\mathcal{C}_{free}$ using m_{init} random configurations. Adding m_{init} samples is repeated until the initial q_1 and terminal q_n configurations are connectable by a path with $c_{1,n} \leq \text{T}_{\text{max}}$ or until the maximal computational time is reached. The lengths $c_{i,j}$ are determined as the shortest paths between all pairs of the target configurations (Line 2). A greedy procedure is used to create initial incumbent solution v (Line 3) by inserting target configurations between q_1 and q_l (for $q_l = q_n$) according to the minimal path prolongation per target reward. The VNS-PRM* then iteratively improves the incumbent solution during which the roadmap expansions are performed to minimize lengths of promising solutions. The algorithm terminates if one of the stopping condition occurs: the maximal number of iterations, or the number of iterations without improvement, or the maximal computational time.

Algorithm 2: VNS-PRM* for the POP

Input : Q – target configurations, T_{max} – budget, m_{init} – VNS-PRM* initial number of samples, m_{exp} – number of expanding samples
Output: v – Found data collecting path

1 $G \leftarrow$ PRM*initialSampling(m_{init})
2 updateRoadmapDistances $c_{i,j} \forall i, j \in (1, \dots, n), i \neq j$
3 $v \leftarrow$ createInitialPath(Q,T_{max}) `// greedy initial solution`
4 **while** *Stopping condition is not met* **do**
5 $p \leftarrow 1$; $\mathcal{B} \leftarrow 0$ `//` $\beta_{i,j} = 0$ `for all` $i, j \in (1, \dots, n), i \neq j$
6 **while** $p \leq p_{max}$ **do**
7 $v' \leftarrow$ shake(v, p)
8 $v'' \leftarrow$ localSearch(v', p)
9 **if** $\mathcal{L}(v'') \leq \text{T}_{\text{max}}$ **and**
10 $[R(v'') > R(v)$ **or** $[R(v'') = R(v)$ **and** $\mathcal{L}(v'') < \mathcal{L}(v)]]$ **then**
11 $v \leftarrow v''$; $p \leftarrow 1$
12 **else**
13 $p \leftarrow p + 1$
14 $M \leftarrow$calculateSampling($\mathcal{B}, \mathcal{P}, m_{exp}$)
15 $G \leftarrow$ PRM*roadmapExpansion(G, M)
16 $c_{i,j} \leftarrow$ updateRoadmapDistances(G,Q) for $\forall i, j \in (1, \dots, n)$

In each VNS-PRM* iteration, the operators of *shake* and *local search* procedures try to increase the sum of the collected rewards. The reward contribution $\mathcal{B} = \{\beta_{i,j}\} \forall i, j \in (1, \dots, n), i \neq j$ of each target configuration pair is stored for further focused roadmap expansion. After performing all p_{max} neighborhood operators, the number of additional samples per each target pair, $M = \{m_{i,j}\}$ for all $i, j \in (1, \dots, n), i \neq j$, is calculated (Alg. 2 Line 14) and the roadmap is expanded (Line 15) together with the update of the shortest paths between all target configurations $c_{i,j}$ (Line 16). The number of additional samples M is based on the reward contributions $\mathcal{B}$ and sampling densities $\mathcal{P} = \{\rho_{i,j}\}$ used in the proposed sampling strategy.

1) Shake: The *shake* procedure creates a new solution v' to get the incumbent solution v from possible local optima. Its two operators ($p_{max} = 2$) tries to randomly select a part of v and alter its position within the vector, but always keep q_{σ_1} and adjust the terminal configuration q_{σ_l} to maximize l but ensure $\mathcal{L}(v) \leq \text{T}_{\text{max}}$. The first **Path move** operator ($p = 1$) randomly selects a part of v and moves it to a different position. The **Path exchange** operator ($p = 2$) selects two random non-overlapping parts of v and exchanges their positions. Thus, v'

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/LRA.2019.2923949

can have changed both the subset Q_l and sequence Σ to visit the configurations in Q_l.

2) Local search: The *local search* procedure tries to optimize solution v'' (initialized by v') by a sequence of simple one target operations. The employed Randomized VNS (RVNS) variant of the VNS uses randomized *local search* operators where each operator examines $|Q|^2$ simple changes of the solution vector v''. Each change is applied to v'' only if it improves the new solution w, i.e., $R(w) > R(v'')$ or decreases the path length $\mathcal{L}(w) < \mathcal{L}(v'')$ for the same reward. The **One target move** operator ($p = 1$) examines changes where a randomly selected target is moved to a different place within the solution vector. The **One target exchange** operator ($p = 2$) examines changes where two randomly selected targets are exchanged. The procedure is summarized in Alg. 3.

Algorithm 3: Local Search Procedure

Input : Q – target configurations, $\mathrm{T_{max}}$ – budget, p – actual neighborhood number, v' – actual solution
Output: v'' – created solution, $\mathcal{B}$ – target pairs rewards

$v'' \leftarrow v'$
for $|Q|^2$ **do**
  if $p = 1$ **then** `// One target move`
    $w_u \leftarrow v''$ with one randomly moved target
  else `// (p = 2) One target exchange`
    $w_u \leftarrow v''$ with one randomly exchanged target
  $\mathcal{B} \leftarrow$updateTargetPairRewards($\mathcal{B}, w_u$)
  $w \leftarrow w_u$ maximize l such that w_{σ_l} is within $\mathrm{T_{max}}$
  if $R(w) > R(v'')$ **or** $[R(w) = R(v'')$ **and** $\mathcal{L}(w) < \mathcal{L}(v'')]$ **then**
    $v'' \leftarrow w$

The local search operators firstly create a possibly unfeasible solution w_u without adjusted position of q_{σ_l} within w_u, and thus $\mathcal{L}(w_u) \not\leq \mathrm{T_{max}}$. It is because the paths connecting the configurations in w_u can be shortened by a roadmap expansion. Therefore, w_u is used for updating $\mathcal{B}$ in $\mathrm{updateTargetPairRewards}()$ where the reward of each target pair contributing to w_u is stored for the prioritization of the promising solutions in the sampling strategy of roadmap expansion. In this way, promising solutions are stored during the search over the POP combinatorial solution space to guide the expansion of the roadmap.

3) Sampling strategy: The sampling strategy of the roadmap expansion uses equidistant sampling within the ellipsoid (Alg. 1) based on the solution space search done by the randomized *local search* procedure. The update of the reward contribution $\mathcal{B}$ in $\mathrm{updateTargetPairRewards}()$ is performed for all consecutive pairs of the target configurations q_i, q_j in the solution w_u. The reward of each pair $\beta_{i,j}$ is considered to be increased by $\Delta_\beta(w_u)$ computed from the average reward per a single target in w_u, using the solution reward $R(w_u)$, multiplied by a relative budget overshoot of the solution length $\mathcal{L}(w_u)$ determined as

$$\Delta_\beta(w_u) = \frac{R(w_u)}{l-1}\left(1 - \frac{\mathcal{L}(w_u) - \mathrm{T_{max}}}{r_o \mathrm{T_{max}} - \mathrm{T_{max}}}\right). \tag{2}$$

The ratio r_o is introduced to allow tuning of the overshoot. The pair reward $\beta_{i,j}$ is then updated by

$$\beta_{i,j} \mathrel{+}= \begin{cases} 0 \text{ for } \mathcal{L}(w_u) > r_o \mathrm{T_{max}} \\ \Delta_\beta(w_u) \text{ for } \mathcal{L}(w_u) \leq r_o \mathrm{T_{max}}, R(w_u) \leq R(v). \\ 10\Delta_\beta(w_u) \text{ for } \mathcal{L}(w_u) \leq r_o \mathrm{T_{max}}, R(w_u) > R(v) \end{cases} \tag{3}$$

In (3), we further distinguish solutions w_u satisfying $r_o \mathrm{T_{max}}$ with the higher reward $R(v)$ than the current best solution for which the increase of $\beta_{i,j}$ is $10\times$ higher to focus sampling of the roadmap. The roadmap expansion thus depends on the rewards $\mathcal{B}$ to focus sampling on the promising sequence of configurations. Besides, the sampling strategy is also designed to depend on its densities $\mathcal{P} = \{\rho_{i,j}\}$ to avoid adding samples to already densely sampled ellipsoids. Thus, the sampling priority $m'_{i,j}$ of each configuration pair (q_i, q_j) is proportional to the reward $\beta_{i,j}$ and inversely proportional to the sampling density $\rho_{i,j}$. This leads to disabling sampling of almost straight line paths with high density.

$$m'_{i,j} = \frac{\beta_{i,j}}{\rho_{i,j}}, \; m_{i,j} = \lceil m_{exp} \frac{m'_{i,j}}{\sum_{i=1}^{i=n}\sum_{j=1}^{j=n} m'_{i,j}} \rceil \tag{4}$$

The number of samples $m_{i,j}$ added to the ellipsoid corresponding to the path between q_i and q_j is determined using (4), where m_{exp} is the number of samples intended to be added to the roadmap during each roadmap expansion.

V. Results

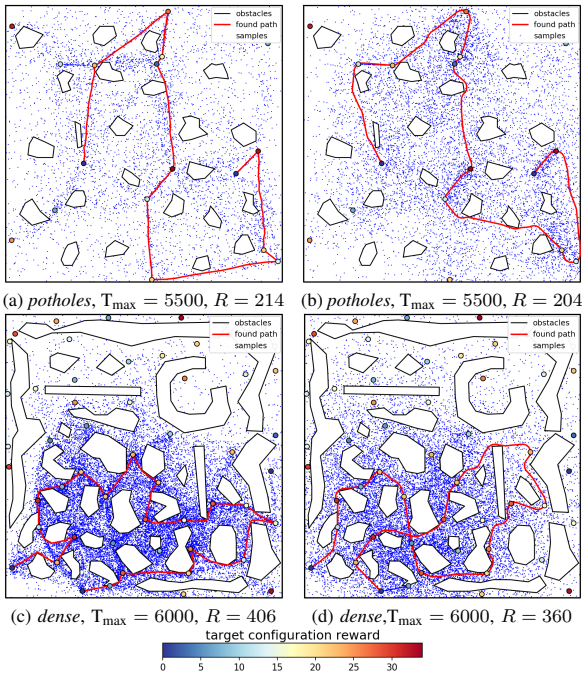
The proposed VNS-PRM* for the introduced POP is evaluated in three simulation scenarios and verified in realistic field deployment. First, the feasibility of the approach is verified for instances with a point robot $q = (x, y) \in \mathbb{R}^2$ and compared to the optimal solution found by the Integer Linear Programming (ILP) using visibility graph for the shortest paths between the target locations. Besides, the proposed online roadmap expansion is compared with the usage of a single static high-density roadmap. The VNS-PRM* is then applied to the POP with the curvature-constrained Dubins vehicle, $q = (x, y, \theta) \in SE(2)$ and compared with our implementation of [27]. Finally, the method is used for $q = (x, y, z) \in \mathbb{R}^3$ environment and further verified in a small real outdoor experiment with a hexarotor UAV.

Two different environments denoted *potholes* and *dense* with 17 and 52 target locations, respectively, and with the dimension of 2000×2000 map units are used for the evaluations. The initial roadmap is constructed with $m_{init} = 1000$ uniformly sampled configurations. The number of samples in the roadmap expansion is $m_{exp} = 50$ and the budget overshoot ratio r_o is empirically set to $r_o = 1.2$. The optimization is terminated after the maximal number of 1000 iterations, 50 iterations without improvement, or after one hour of the computational time. The proposed method[1] is implemented in C++, and all the reported results are achieved using a single core of the Intel Xenon processors cluster (2.2GHz-3.3GHz). An example of solutions found by the proposed VNS-PRM* for $q \in \mathbb{R}^2$ are depicted in Fig. 2.

The first evaluation scenario is focused on the comparison of the proposed method with the optimal solutions for $q \in \mathbb{R}^2$ that has been found using ILP OP formulation [1] in CPLEX 12.6.1 that is denoted ILP-VIS. The VNS-based solution without the PRM* is denoted VNS-VIS and both the ILP-VIS and VNS-VIS use path lengths determined from the visibility graph. The

[1]Method implementation, benchmark instances and obtained solutions are available at https://github.com/ctu-mrs/vns-prm-pop

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/LRA.2019.2923949



(a) *potholes*, $\mathrm{T_{max}} = 5500$, $R = 214$ (b) *potholes*, $\mathrm{T_{max}} = 5500$, $R = 204$

(c) *dense*, $\mathrm{T_{max}} = 6000$, $R = 406$ (d) *dense*,$\mathrm{T_{max}} = 6000$, $R = 360$

Fig. 2. Example of the POP solutions for $q \in \mathbb{R}^2$ in (a) and (c), and $q \in SE(2)$ for $\rho = 60$ in (b) and (d) for both simulation testing environments.

achieved results from fifty trials of all considered instances and budget constraints $\mathrm{T_{max}}$ are reported in Table I, where R_m and R is the maximal and average collected reward, respectively, σ is the standard deviation, $\mathcal{L}_r$ is the ratio of the average path length with respect to the particular $\mathrm{T_{max}}$, and T is the average computational time (in seconds). The best solutions are highlighted in bold.

TABLE I
RESULTS ON THE POP INSTANCES WITH VISIBILITY GRAPH FOR $q \in \mathbb{R}^2$

Pr.	$\mathrm{T_{max}}$	ILP-VIS		VNS-VIS		VNS-PRM*			
		R_m	T	R_m	T	R_m	$R \pm \sigma$	$\mathcal{L}_r$	T
potholes	1500	**48**	0.01	**48**	0.07	**48**	48.0± 0.0	0.87	4.0
	2500	**91**	0.06	**91**	0.11	**91**	89.1± 0.4	0.90	4.4
	3500	**143**	0.11	**143**	0.14	**143**	131.1± 8.1	0.98	9.3
	4500	**176**	0.08	**176**	0.17	168	161.6± 7.9	0.94	13.2
	5500	**214**	0.05	**214**	0.19	**214**	204.0± 8.3	0.97	17.4
	6500	**247**	0.09	**247**	0.21	245	235.8± 8.0	0.97	24.6
	7500	**270**	0.07	**270**	0.18	**270**	266.5± 4.7	0.97	24.4
	8500	**292**	0.06	**292**	0.20	**292**	292.0± 0.0	0.94	21.5
	9500	**299**	0.02	**299**	0.19	**299**	298.6± 1.7	0.93	18.6
dense	2000	**121**	1.22	**121**	1.06	**121**	117.8± 2.7	0.96	31.9
	4000	**284**	0.99	**284**	1.43	**284**	274.5± 6.4	0.98	71.8
	6000	**406**	3.71	**406**	1.70	**406**	397.9± 7.0	0.99	210.3
	8000	**522**	2.08	514	1.88	498	485.0± 9.2	0.98	264.0
	10000	**630**	16.06	618	2.32	613	568.5±17.9	0.99	489.9
	12000	**741**	0.99	718	2.51	705	667.7±20.7	0.99	937.6
	14000	**827**	0.75	803	2.20	791	745.0±22.8	0.99	1221.1
	16000	**892**	1.10	883	2.00	881	826.8±22.0	0.99	1884.0
	18000	**922**	4.11	**922**	1.97	**922**	891.1±16.8	0.99	2187.9

Although the ILP-VIS provides optimal solutions, the approach is usable only with the point robot and configuration space where the visibility graph can be used to determine the shortest collision-free paths. The heuristic VNS-VIS provides competitive results to the optimal solutions, but most importantly, the proposed VNS-PRM* provides the optimal solutions in most of the cases, except the *dense* environment which contains many obstacles. The ILP-VIS and VNS-VIS utilize precomputed visibility graph (not counted in their computational times), and therefore, the high computational requirements of VNS-PRM* are not surprising. The main advantage of the VNS-PRM* is in the applicability for different motion model, e.g., Dubins vehicle, and extendibility for more complex robot shapes. The reported results for the VNS-VIS and VNS-PRM* indicate that the inability to find the optimal solutions using VNS-PRM* is caused by the VNS part of the method as the optimal solution is not found using the shortest paths in the VNS-VIS. Nevertheless, based on the reported results, we consider the proposed approach feasible, and we further report on the impact of the proposed sampling strategy.

The online sampling strategy with the preference of sampling between target configurations of the promising solutions found by the VNS is compared with a solution found on a roadmap created only by the initial sampling, but with a high number of samples m_{init}. The evaluation is performed for the *potholes* environment and the results are reported in Table II, where VNS-Static_Roadmap denotes the variant with only initial sampling that has been considered with $m_{init} \in \{1 \times 10^4, 3 \times 10^4, 6 \times 10^4, 1 \times 10^5, 1.5 \times 10^5\}$ uniform samples in $\mathcal{C}_{free}$ without the online expansion. In addition to the maximal sum of the collected rewards R_m from fifty trials and the corresponding average computational time T in seconds, the average time of the last solution improvement of the VNS-PRM* is reported in the column T_i.

TABLE II
ONLINE SAMPLING STRATEGY VS. INITIAL SAMPLING ONLY

$\mathbf{T_{max}}$	VNS-PRM*			VNS-Static_Roadmap with m_{init} samples							
				10^4		3×10^4		6×10^4		10^5		1.5×10^5	
	R_m	T	T_i	R_m	T	R_m	T	R_m	T	R_m	T	R_m	T
1500	**48**	4	1	**48**	34	**48**	92	**48**	153	**48**	223	**48**	361
2500	**91**	4	1	89	42	89	83	89	155	**91**	259	**91**	419
3500	**143**	9	4	125	43	132	78	132	177	127	299	**143**	521
4500	**168**	13	5	**168**	39	**168**	115	**168**	219	**168**	369	**168**	569
5500	**214**	17	6	204	41	204	95	**214**	222	**214**	395	**214**	721
6500	**245**	25	8	**245**	47	**245**	123	**245**	204	**245**	478	237	916
7500	**270**	24	9	**270**	45	**270**	120	**270**	292	**270**	515	**270**	818
8500	**292**	21	6	**292**	52	**292**	122	**292**	260	**292**	511	**292**	836
9500	**299**	19	7	**299**	40	**299**	119	**299**	252	**299**	519	**299**	1056

The average computational time of the VNS-PRM* solution is similar to the initial sampling with $m_{init} = 10^4$, but the time of the last solution improvement T_i is significantly lower. Therefore, the relatively high number of 50 iterations without improvement, which however causes the termination in a majority of cases, can be decreased without affecting the solution quality. The computational time of VNS-Static_Roadmap is dominated by the roadmap construction and finding the shortest paths between all target pairs using Dijkstra's algorithm. The computational requirements of VNS itself using already known shortest paths can be seen for VNS-VIS approach in Table I. The VNS-PRM* finds the best solutions in all instances while the computationally demanding high number of initial samples does not provide the best solution for all considered $\mathrm{T_{max}}$. The average number of samples needed to find solutions using VNS-PRM* in Table II is 6678. Furthermore, the VNS-PRM* is an anytime algorithm which starts with a relatively small number of samples to quickly find a feasible solution that is then continuously improved if more computational time is available, which is shown in Fig. 3.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/LRA.2019.2923949

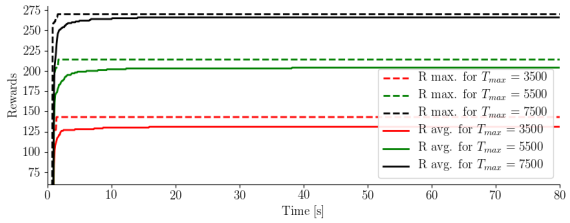


Fig. 3. Evolution of the average and maximal sum of the collected rewards for the *potholes* scenario and selected budgets T_{max}.

The proposed VNS-PRM* has been further examined for curvature-constrained planning with Dubins vehicle, configuration space $SE(2)$, and compared with our implementation of [27] for the POP. In $SE(2)$, the optimal Dubins maneuvers are used as the distance between every two configurations. Because of Dubins vehicle, each target location is considered as 12 configurations with equidistantly spread heading angle θ to allow each target location to be visited using a different vehicle heading. Only a single such sample is, however, allowed to collect the reward associated with the particular target location which transforms the problem into an instance of the Set Orienteering Problem [29]. The implementation of [27] for Dubins vehicle and the POP (denoted as the PRM-MT) uses the navigation PRM roadmap in $\mathbb{R}^2$ with 1000 samples to guide the expansion of the motion tree in $SE(2)$. Since [27] does not address the POP, the following modifications have been made: an ILP OP solver is used instead of the PC-TSP solver, a solution has to reach proximity of the terminal location, the solution length is used instead of the execution time, and the sum of the rewards of unvisited target locations is used instead of the PC-TSP penalty. The results for solved instances with the turning radius of $\rho = 60$ are reported in Table III.

TABLE III
RESULTS ON THE POP INSTANCES WITH DUBINS VEHICLE – $q \in SE(2)$

Pr.	T_{max}	PRM-MT				VNS-PRM*			
		R_m	$R \pm \sigma$	$\mathcal{L}_r$	T_i	R_m	$R \pm \sigma$	$\mathcal{L}_r$	T_i
potholes	1500	**48**	35.6±11.0	0.87	3	**48**	48.0± 0.0	0.89	5
	2500	**89**	70.5±11.0	0.91	21	**89**	89.0± 0.0	0.94	6
	3500	118	94.6±15.7	0.94	145	**127**	122.7± 4.3	0.93	56
	4500	153	107.8±27.4	0.90	263	**168**	161.8± 7.7	0.98	109
	5500	179	109.4±26.9	0.79	363	**204**	190.3±13.3	0.96	124
	6500	221	120.1±35.0	0.74	370	**242**	226.0±10.8	0.97	165
	7500	234	88.5±41.5	0.61	407	**263**	257.1± 5.9	0.95	221
	8500	167	92.4±32.9	0.54	545	**292**	281.3±11.2	0.97	103
	9500	219	91.7±43.1	0.52	552	**299**	295.2± 3.5	0.94	67
dense	2000	80	67.6± 9.5	0.90	219	**108**	107.8± 1.4	0.90	95
	4000	154	74.9±26.4	0.54	2514	**237**	225.2± 4.8	0.96	560
	6000	110	66.9±14.6	0.30	2513	**360**	339.5±10.5	0.98	812
	8000	144	34.3±32.4	0.13	2522	**472**	429.0±11.7	0.98	1065
	10000	130	9.7±21.7	0.03	3042	**564**	510.9±17.0	0.99	1363
	12000	52	2.8±10.2	0.01	3144	**646**	596.7±21.2	0.98	1827
	14000	121	8.1±23.1	0.02	2815	**719**	670.7±20.9	0.99	2356
	16000	84	3.5±13.9	0.01	2510	**806**	742.3±23.9	0.99	2136
	18000	55	8.9±16.2	0.02	2879	**839**	798.8±22.8	0.98	2488

Regarding the reported results, the VNS-PRM* outperforms the PRM-MT in both the maximal achieved rewards R_m and the average rewards R with smaller σ. The average ratio of the used budget $\mathcal{L}_r$ for the PRM-MT indicates that the method is unable to exploit the available travel budget. This is caused by uniform sampling of PRM-MT along the navigation roadmap without considering the ability of the motion tree to reach these random samples. This can be improved by generating samples according to the progress of the motion tree [30]. The average time of the last solution improvement T_i of the VNS-PRM* is also lower than for the PRM-MT in most of the cases. Low values of the collected rewards in *dense* scenarios suggest that the PRM-MT [27] struggles with narrow passages and the guidance along solutions found in static roadmaps becomes less effective for longer T_{max} with the possibility to visit more targets.

The VNS-PRM* is further verified in 3D scenario denoted as *building* that is 20×30×6 large and has seven rooms in each of the two floors, see Fig. 4. One target location is in each room with the reward in the range 5–30, thus 14 targets in the total. The upper floor is accessible only by tight windows and the robot is modeled as a cylindrical object with 0.7 diameter and 0.5 height with the configuration $q \in \mathbb{R}^3$.

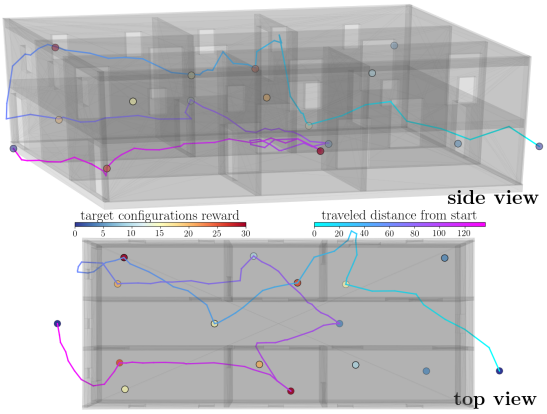


Fig. 4. Example solution of the POP in the *building* environment for T_{max} = 140 with the collected reward $R = 230$ and solution length of 136.3.

The computational results for the *building* scenario are depicted in Table IV, where T_{init} denotes the average time to find initial solution with the average reward R_{init} and i is the average number of the VNS-PRM* iterations.

TABLE IV
RESULTS ON THE POP INSTANCES FOR $q \in \mathbb{R}^3$

Pr.	T_{max}	VNS-PRM*						
		R_{init}	R_m	$R \pm \sigma$	$\mathcal{L}_r$	i	T_{init}	T
building	60	4.8	60	29.2±14.5	0.92	71	0.6	6.7
	80	30.5	100	76.8± 8.5	0.95	97	0.5	14.5
	100	45.4	120	101.4±20.3	0.94	108	0.6	23.5
	120	52.6	150	132.1±13.9	0.96	140	0.6	45.5
	140	73.0	175	163.2±11.0	0.96	143	0.6	52.9
	160	81.1	205	179.4±28.5	0.95	137	0.6	47.6
	180	87.2	215	203.7± 7.9	0.96	150	0.6	68.0
	200	92.0	225	215.7± 7.4	0.96	155	0.6	70.5
	220	100.1	230	224.7± 4.3	0.95	147	0.7	67.1

Table IV shows that the VNS-PRM* finds an initial solution within one second with the average solution quality of 35.3% of the best-found solution. The number of iterations i indicates that the algorithm terminates after the maximum of 50 iterations without improvement. Furthermore, the comparison of computational times for $q \in \mathbb{R}^2$, $SE(2)$, and $\mathbb{R}^3$ show the increased computational requirements of planning in $SE(2)$, which is caused by the nearest neighborhood search of the PRM* where k-d trees are not effective in $SE(2)$.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/LRA.2019.2923949

Finally, the VNS-PRM* has been experimentally verified in a small data collection mission with a hexarotor UAV. The scenario consists of three walls and four cylindrical obstacles representing the indoor- or urban-like environment, see Fig. 1 with the visualization of the results. The environment was about $9\times10\,\mathrm{m}$ large with ten target locations, including initial and terminal locations, with the constant altitude, and thus the VNS-PRM* search space is $\mathbb{R}^2$. The UAV is modeled as a cylindrical object with $1.4\,\mathrm{m}$ diameter and $0.5\,\mathrm{m}$ height which corresponds to $1.75\times$ enlargement of the real physical dimension of the UAV to compensate possible localization inaccuracies. The considered travel budget limit was set to $\mathrm{T_{max}} = 25\,\mathrm{m}$ and the solution found onboard of the UAV before flight by the VNS-PRM* within $T = 8.4\,\mathrm{s}$ is $24.11\,\mathrm{m}$ long with the collected reward $R = 75$. The model predictive trajectory tracking [31] was used to precisely follow the trajectory and visit all six planned target locations.

VI. Conclusions

A novel generalization of the Orienteering Problem (OP) for robotic data collection scenarios is introduced in this paper. The problem is called Physical Orienteering Problem (POP), and it is suitable for cases where collision-free paths in environment with obstacles are required together with the maximization of collected rewards from the given target locations using the limited travel budget. The proposed solution of newly introduced POP is based on the Variable Neighborhood Search (VNS) metaheuristic for the OP that is combined with the asymptotically optimal motion planner PRM*. The proposed VNS-PRM* starts with a low-dense roadmap that is continuously expanded during the VNS-based route optimization by selecting the most promising solutions for shortening the collision-free paths and thus allowing to maximize the collected rewards. The presented results show that the proposed VNS-PRM* is a feasible and vital method and it can provide optimal solutions when compared on 2D instances with a point robot. Furthermore, the proposed roadmap expansion strategy demonstrates computational benefits in comparison to a very dense initial roadmap. The main benefit of the approach rests in the generalization of the OP for more complex configuration spaces demonstrated in a solution of the POP in $\mathbb{R}^3$ and with Dubins vehicle in $SE(2)$.

References

[1] P. Vansteenwegen, W. Souffriau, and D. V. Oudheusden, "The orienteering problem: A survey," *European Journal of Operational Research*, vol. 209, no. 1, pp. 1–10, 2011.
[2] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press, 2006.
[3] I. Jawhar, N. Mohamed, J. Al-Jaroodi, and S. Zhang, "A framework for using unmanned aerial vehicles for data collection in linear wireless sensor networks," *Journal of Intelligent & Robotic Systems*, vol. 74, no. 1, pp. 437–453, Apr 2014.
[4] T. Tsiligirides, "Heuristic methods applied to orienteering," *The Journal of the Operational Research Society*, vol. 35, no. 9, pp. 797–809, Sep. 1984.
[5] R. Pěnička, J. Faigl, P. Váňa, and M. Saska, "Dubins orienteering problem," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1210–1217, April 2017.
[6] P. Hansen and N. Mladenović, "Variable neighborhood search: Principles and applications," *European Journal of Operational Research*, vol. 130, no. 3, pp. 449–467, May 2001.
[7] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.
[8] S. M. Lavalle, "Rapidly-exploring random trees: A new tool for path planning," Iowa State University, Tech. Rep., 1998.
[9] L. E. Kavraki, P. Svestka, J. . Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, Aug 1996.
[10] M. Elbanhawi and M. Simic, "Sampling-Based Robot Motion Planning: A Review," *IEEE Access*, vol. 2, pp. 56–77, 2014.
[11] A. Gunawan, H. C. Lau, and P. Vansteenwegen, "Orienteering problem: A survey of recent variants, solution approaches and applications," *European Journal of Operational Research*, vol. 255, no. 2, pp. 315–332, 2016.
[12] R. Ramesh, Y.-S. Yoon, and M. H. Karwan, "An optimal algorithm for the orienteering tour problem," *ORSA Journal on Computing*, vol. 4, no. 2, pp. 155–165, 1992.
[13] M. Fischetti, J. J. S. González, and P. Toth, "Solving the orienteering problem through branch-and-cut," *INFORMS Journal on Computing*, vol. 10, no. 2, pp. 133–148, 1998.
[14] A. Şevkli and F. Sevilgen, "StPSO: strengthened particle swarm optimization," *Turkish Journal of Electrical Engineering and Computer Sciences*, vol. 18, no. 6, pp. 1095–1114, 2010.
[15] M. Schilde, K. F. Doerner, R. F. Hartl, and G. Kiechle, "Metaheuristics for the bi-objective orienteering problem," *Swarm Intelligence*, vol. 3, no. 3, pp. 179–201, 2009.
[16] Z. Sevkli and F. E. Sevilgen, "Variable neighborhood search for the orienteering problem," in *ISCIS*, 2006, pp. 134–143.
[17] 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*, vol. 79, no. 3, pp. 497–516, July 1957.
[18] A. Bit-Monnot, R. Bailon-Ruiz, and S. Lacroix, "A local search approach to observation planning with multiple uavs," in *ICAPS*, 2018, pp. 437–445.
[19] R. Pěnička, J. Faigl, M. Saska, and P. Váňa, "Data collection planning with non-zero sensing distance for a budget and curvature constrained unmanned aerial vehicle," *Autonomous Robots*, Feb 2019.
[20] J. Faigl and R. Pěnička, "On close enough orienteering problem with dubins vehicle," in *IEEE/RSJ IROS*, 2017, pp. 5646–5652.
[21] D. Thakur, M. Likhachev, J. Keller, V. Kumar, V. Dobrokhodov, K. Jones, J. Wurz, and I. Kaminer, "Planning for opportunistic surveillance with multiple robots," in *IEEE/RSJ IROS*, 2013, pp. 5750–5757.
[22] D. Perez, P. Rohlfshagen, and S. M. Lucas, "The physical travelling salesman problem: Wcci 2012 competition," in *IEEE Congress on Evolutionary Computation*, 2012, pp. 1–8.
[23] D. Devaurs, T. Siméon, and J. Cortés, "A multi-tree extension of the transition-based rrt: Application to ordering-and-pathfinding problems in continuous cost spaces," in *IEEE/RSJ IROS*, 2014, pp. 2991–2996.
[24] J. McMahon and E. Plaku, "Autonomous underwater vehicle mine countermeasures mission planning via the physical traveling salesman problem," in *OCEANS - MTS/IEEE*, 2015, pp. 1–5.
[25] S. Edelkamp, M. Pomarlan, and E. Plaku, "Multiregion inspection by combining clustered traveling salesman tours with sampling-based motion planning," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 428–435, April 2017.
[26] J. McMahon and E. Plaku, "Mission and motion planning for autonomous underwater vehicles operating in spatially and temporally complex environments," *IEEE Journal of Oceanic Engineering*, vol. 41, no. 4, pp. 893–912, Oct 2016.
[27] ——, "Autonomous data collection with limited time for underwater vehicles," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 112–119, Jan 2017.
[28] J. Dezert and C. Musso, "An efficient method for generating points uniformly distributed in hyperellipsoids," in *Workshop on Estimation, Tracking and Fusion:A Tribute to Yaakov Bar-Shalom*, 2001.
[29] R. Pěnička, J. Faigl, and M. Saska, "Variable neighborhood search for the set orienteering problem and its application to other orienteering problem variants," *European Journal of Operational Research*, vol. 276, no. 3, pp. 816 – 825, 2019.
[30] V. Vonásek, J. Faigl, T. Krajník, and L. Přeučil, "RRT-path–a guided rapidly exploring random tree," in *RoMoCo*, 2009, pp. 307–316.
[31] T. Báča, D. Heřt, G. Loianno, M. Saska, and V. Kumar, "Model predictive trajectory tracking and collision avoidance for reliable outdoor deployment of unmanned aerial vehicles," in *IEEE/RSJ IROS*, 2018, pp. 6753–6760.

Chapter 8

Results and Discussion

In this chapter, we summarize the contributions of the presented articles as a whole with respect to the initially presented research challenges of the data collection planning for UAVs and this thesis. We further suggest future research objectives for particular challenges.

(1) Feasibility of traveling with nonholonomic fixed-wing UAV or dynamically constrained VTOL UAV has been addressed in all core publications. In general, the selection of motion primitive highly influence the quality of the final path, i.e., the length of a path, its time of flight, or the amount of collected data within limited budget. However, the more complex motion primitives such as Dubins airplane model [12a], Bézier curves [4c] or Hermite curves [11a] increase computational requirements of the planning. Therefore, all the core publications [1c]–[5c] mainly use a rather simple Dubins vehicle model, which creates feasible plans for UAVs traveling a certain speed.

The main challenge of using curvature constrained Dubins vehicle is in the necessity of determining the heading angle of the vehicle for each visited target location to feasibly connect adjacent Dubins maneuvers. In the first core publication [1c], the heading angles at the target locations are equidistantly sampled, and the particular heading angle samples for a given sequence of targets are selected such that the path length is minimized. The pure combinatorial optimization of the SOP presented in the core publication [3c] is also based on the sampling-based approach for selecting the heading angles. In [5c], we use a notion of planning in configuration space. However, for Dubins vehicle, we consider 12 configurations with the equidistantly spread heading angle at each target location. The SOM-based solution presented in [4c] uses an informed sampling-based algorithm for the DTP [97] to find the appropriate heading angles for a given sequence of targets. The advantage of using [97] is the tight lower-bound that can guide the informed sampling of heading angles close to the optimal values for a given sequence. However, for the combinatorial route optimization of the target sequence and target subset selection, the heading samples have to be initially present in order to find some good route where the heading angles can be optimized. The sampling has to be high-dense to obtain solutions with high quality, which, on the other hand, significantly increases computational requirements as shown in [1c]. On the other hand, for a final solution, only a fraction of such samples is ever used.

Therefore, in [2c] a low-dense initial sampling of heading samples is used together with local optimization of headings [56] of high-quality solutions. Such optimized headings are then iteratively inserted into the graph of all samples and further used while exploring the combinatorial part of the data collection routing problem. The results presented in [2c] suggest that for the OP variants, the low-dense initial sampling with further heading optimization increases solution quality and decreases computational time compared to the high-dense sampling approaches.

For future work, we would like to employ the DTP solver [97] for the solution of the OP variants such as for the one in [2c].

(2) Planning with respect to the limited time of flight has been addressed in all core publication with the exception of [4c]. We employ the OP and its novel variants to formulate the data collection planning with maximization of the collected reward within a

limited budget. In [1c], we propose the DOP that generalizes the OP for Dubins vehicle, however, it requires to find the heading angles of the vehicle in each visited target location. We propose the VNS-based method for solving the DOP. The empirical results show that the DOP formulation is necessary and can not be replaced by a straightforward combination of the ordinary OP and the Dubins TSP. The main reason is the fact that the selection of heading angles highly influences the solution length, which is limited by a given budget of the DOP.

We further propose other variants of the OP for data collection with UAV. In [9a] and in the core publication [2c], we generalize the DOP for non-zero sensing range, which we call Dubins Orienteering Problem with Neighborhoods (DOPN). While in [9a], the VNS-based solution is a straightforward extension of the VNS approach for the DOP [1c] with additional neighborhood sampling, in [2c], we propose a novel VNS operators for heading angle and neighborhood visit position optimization. In [3c], we unify both the sampling-based DOP and DOPN as a variant of the Set Orienteering Problem (SOP) and propose a modified VNS-based approach and ILP formulation. Finally, we propose a variant of the OP for environments with obstacles called the Physical Orienteering Problem (POP), which can be understood as OP explicitly deployed in the configuration space [27]. The proposed VNS-PRM* method combines the combinatorial optimization of the VNS for the OP part with asymptotically optimal Probabilistic Roadmaps for collision-free motion planning. In all mentioned OP variants, the limited budget has to be directly considered in the planning method in order to get feasible plans, as shown for the DOP in [1c].

In future work, we would like to focus on online replanning of OP solutions for, e.g., partially known environments with static and moving obstacles.

(3) Data collection planning with non-zero sensing range increases the solution quality in most cases. The ordinary OP can be generalized to the OPN [8a]. The proposed SOM-based planner [8a] is shown to provide higher collected rewards for larger sensing ranges using the same budget. For Dubins vehicle, the DOP can be generalized to the DOPN [2c], [9a]. Similarly to the OPN, the usage of non-zero sensing range in the DOPN increases the collected reward, as shown for the VNS-based planners in [2c], [9a] and SOM-based planner in [10a]. For the data collection planning formulation of the multi-vehicle Dubins TSP with Neighborhoods (m-DTSPN) in [4c], the solution quality for larger sensing ranges also increases, and shorter paths are found using both the VNS-based and SOM-based planners. However, the computational complexity for non-zero sensing range formulations increases due to additional search for locations of visits within the neighborhoods of the target locations.

The VNS based planners for the DOPN in [9a] and for the m-DTSPN in [4c] use a static equidistant sampling of neighborhood positions around each target. The SOM-based planners for the OPN [8a], DOPN [10a] and m-DTSPN [4c], use adaptation of the SOM neurons to find the locations of visits within the neighborhoods. When compared for the DOPN, the best performing approach is the VNS-based method presented in [2c] that uses a low-dense initial sampling of the neighborhood positions that are then iteratively optimized and inserted to the graph of all samples. The approach in [2c] thus uses the same technique for Dubins vehicle heading samples and the samples of neighborhood positions.

The future work might consider recent approaches for solving Generalized Dubins Interval Problem [98] to find near-optimal values of Dubins vehicle headings and neighborhood positions for a given sequence of samples.

(4) Finding optimal solutions for data collection with Dubins vehicle and non-zero sensing range has been addressed in core publication [3c]. We show that both the OPN and the DOP can be solved as a recent variant of the OP from operational research called the Set Orienteering Problem (SOP) [26]. In the SOP, the nodes are grouped in clusters and each cluster has assigned reward. The objective of the SOP is to maximize the collected reward by visiting at least one node in selected clusters using a limited budget path. In the core publication [3c], the SOP is addressed using the VNS-based approach and shown to outperform the existing heuristic. We also propose a novel ILP formulation of the SOP that can be solved significantly faster than the existing one. Most importantly, we show that the OPN and the DOP can be formulated as the SOP where the original (D)OP(N) target locations become clusters of the SOP, and the individual heading angle samples or neighborhood position samples become nodes of the respective clusters. The DOPN, however not shown in [3c], can be similarly formulated as the SOP. Therefore, the data collection with Dubins vehicle and non-zero sensing range can be solved optimally for a given sampling of the problems, using the ILP formulation of the SOP.

For future work, we intend to investigate finding optimal solutions regardless of the sampling, e.g., by combining solutions of the Generalized Dubins Interval Problem [98] with the ILP formulation of the SOP.

(5) Data collection planning in environments with obstacles is particularly challenging due to the necessity of combining multi-goal routing with collision-free planning. The approach proposed in the last core publication [5c] combines the VNS-based method for the routing part with asymptotically optimal sampling-based Probabilistic Roadmap (PRM*) [89] method for collision-free point-to-point planning. We call the novel variant of the OP in configuration space as the Physical Orienteering Problem (POP) and the proposed method as the VNS-PRM*.

The challenging part of combining the two problems in the POP is solved in the VNS-PRM* such that the PRM* creates an initial low-dense roadmap for starting the combinatorial optimization of the POP using the VNS. Afterward, the roadmap is iteratively expanded in every VNS iteration in order to shorten selected collision-free paths in order to further maximize collected reward within the same budget. The selection of the paths for shortening is made during the combinatorial local search of the VNS with the preference of solutions with a length close to the budget and with reward close to the best-found solution. Therefore, only the promising solutions are selected for shortening, and the roadmap is expanded in hyperellipsoids of individual collision-free paths between targets in such solutions. This way, the roadmap is only expanded in promising parts, while the VNS combinatorial optimization part simultaneously uses the roadmap when trying to maximize the sum of the collected rewards.

In future work, we would like to address the approximation factor and optimality of the OP in configuration space.

Chapter 9

Conclusion

This thesis addressed the topic of data collection planning for Unmanned Aerial Vehicles (UAV). In data collection missions, UAVs are typically required to visit a set of target locations in order to collect desired data. The data collection planning for UAVs was formulated either as a problem of minimizing the path visiting all target locations or maximizing the collected data with a restricted budget. The thesis is based on five core publications, where challenges related to the data collection planning for UAVs are tackled. We proposed several data collection planning methods mostly based on the Variable Neighborhood Search (VNS), unsupervised learning of the Self-Organizing Map (SOM), and also based on a solution of the Integer Linear Programming (ILP) problem formulations. We proposed to use Dubins vehicle to model UAV in data collection planning with limited time of flight and address the novel problem using the VNS-based method. Further, we extended the method for cases with non-zero sensing range where the efficiency of data collection can be increased by using, for example, long-range sensors. Both data collection with Dubins vehicle model and non-zero sensing range were formulated as an ILP problem and solved optimally for a given sampling of Dubins vehicle heading angles and sensing positions. We also addressed the multi-robot variant of the data collection with Dubins vehicle and non-zero sensing range using the VNS-based and SOM-based methods. Finally, we proposed a novel formulation of the data collection planning for a limited budget vehicle and environments with obstacles, that is solved by a tightly coupled combination of VNS and asymptotically optimal sampling-based motion planner.

Bibliography

[1c] **R. Pěnička**, J. Faigl, P. Váňa, and M. Saska, "Dubins orienteering problem," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1210–1217, 2017.

[2c] **R. Pěnička**, J. Faigl, M. Saska, and P. Váňa, "Data collection planning with non-zero sensing distance for a budget and curvature constrained unmanned aerial vehicle," *Autonomous Robots*, vol. 43, no. 8, pp. 1937 –1956, 2019.

[3c] **R. Pěnička**, J. Faigl, and M. Saska, "Variable neighborhood search for the set orienteering problem and its application to other orienteering problem variants," *European Journal of Operational Research*, vol. 276, no. 3, pp. 816 –825, 2019.

[4c] J. Faigl, P. Váňa, **R. Pěnička**, and M. Saska, "Unsupervised learning-based flexible framework for surveillance planning with aerial vehicles," *Journal of Field Robotics*, vol. 36, no. 1, pp. 270–301, 2019.

[5c] **R. Pěnička**, J. Faigl, and M. Saska, "Physical orienteering problem for unmanned aerial vehicle data collection planning in environments with obstacles," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 3005–3012, 2019.

[6a] V. Spurný, T. Báča, M. Saska, **R. Pěnička**, T. Krajník, J. Thomas, D. Thakur, G. Loianno, and V. Kumar, "Cooperative autonomous search, grasping, and delivering in a treasure hunt scenario by a team of unmanned aerial vehicles," *Journal of Field Robotics*, vol. 36, no. 1, pp. 125–148, 2019.

[7a] G. Loianno, V. Spurný, J. Thomas, T. Báča, D. Thakur, D. Heřt, **R. Pěnička**, T. Krajník, A. Zhou, A. Cho, M. Saska, and V. Kumar, "Localization, grasping, and transportation of magnetic objects by a team of mavs in challenging desert-like environments," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1576–1583, 2018.

[8a] J. Faigl, **R. Pěnička**, and G. Best, "Self-organizing map-based solution for the orienteering problem with neighborhoods," in *IEEE International Conference on Systems, Man, and Cybernetics*, 2016, pp. 1315–1321.

[9a] **R. Pěnička**, J. Faigl, P. Váňa, and M. Saska, "Dubins orienteering problem with neighborhoods," in *International Conference on Unmanned Aircraft Systems*, 2017, pp. 1555–1562.

[10a] J. Faigl and **R. Pěnička**, "On close enough orienteering problem with dubins vehicle," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017, pp. 5646–5652.

[11a] A. Dubeň, **R. Pěnička**, and M. Saska, "Information gathering planning with hermite spline motion primitives for aerial vehicles with limited time of flight," in *Modelling and Simulation for Autonomous Systems*, 2019, pp. 172–201.

[12a] P. Váňa, J. Faigl, J. Sláma, and **R. Pěnička**, "Data collection planning with dubins airplane model and limited travel budget," in *European Conference on Mobile Robots*, 2017, pp. 1–6.

[13a] D. Zahrádka, **R. Pěnička**, and M. Saska, "Route planning for teams of unmanned aerial vehicles using dubins vehicle model with budget constraint," in *Modelling and Simulation for Autonomous Systems*, 2019, pp. 365–389.

The core publications of this thesis are referenced with [*c] and other author's publications with [*a].

[14a] V. Vonásek and **R. Pěnička**, "Space-filling forest for multi-goal path planning," in *International Conference on Emerging Technologies and Factory Automation*, 2019, pp. 1587–1590.

[15a] **R. Pěnička**, M. Saska, C. Reymann, and S. Lacroix, "Reactive dubins traveling salesman problem for replanning of information gathering by uavs," in *European Conference on Mobile Robots*, 2017, pp. 1–6.

[16a] T. Báča, P. Štěpán, V. Spurný, D. Heřt, **R. Pěnička**, M. Saska, J. Thomas, G. Loianno, and V. Kumar, "Autonomous landing on a moving vehicle with an unmanned aerial vehicle," *Journal of Field Robotics*, vol. 36, no. 5, pp. 874–891, 2019.

[17a] A. Vick, V. Vonásek, **R. Pěnička**, and J. Krüger, "Robot control as a service — towards cloud-based motion planning and control for industrial robots," in *International Workshop on Robot Motion and Control*, 2015, pp. 33–39.

[18a] V. Vonásek and **R. Pěnička**, "Computation of approximate solutions for guided sampling-based motion planning of 3d objects," in *International Workshop on Robot Motion and Control*, 2019, pp. 231–238.

[19a] ——, "Path planning of 3d solid objects using approximate solutions," in *International Conference on Emerging Technologies and Factory Automation*, 2019, pp. 593–600.

[20] G. Pajares, "Overview and current status of remote sensing applications based on unmanned aerial vehicles (uavs)," *Photogrammetric Engineering & Remote Sensing*, vol. 81, no. 4, pp. 281–329, 2015.

[21] P. Toth and D. Vigo, *The vehicle routing problem*. SIAM, 2002.

[22] A. Otto, N. Agatz, J. Campbell, B. Golden, and E. Pesch, "Optimization approaches for civil applications of unmanned aerial vehicles (uavs) or aerial drones: A survey," *Networks*, vol. 72, no. 4, pp. 411–458, 2018.

[23] 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*, vol. 79, no. 3, pp. 497–516, Jul. 1957.

[24] T. Tsiligirides, "Heuristic methods applied to orienteering," *The Journal of the Operational Research Society*, vol. 35, no. 9, pp. 797–809, Sep. 1984.

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

[26] C. Archetti, F. Carrabs, and R. Cerulli, "The set orienteering problem," *European Journal of Operational Research*, vol. 267, no. 1, pp. 264–272, 2018.

[27] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press, 2006.

[28] H. Ergezer and K. Leblebicioğlu, "3d path planning for multiple uavs for maximum information collection," *Journal of Intelligent & Robotic Systems*, vol. 73, no. 1, pp. 737–762, 2014.

[29] I. Jawhar, N. Mohamed, J. Al-Jaroodi, and S. Zhang, "A framework for using unmanned aerial vehicles for data collection in linear wireless sensor networks," *Journal of Intelligent & Robotic Systems*, vol. 74, no. 1, pp. 437–453, 2014.

[30] C. Wang, F. Ma, J. Yan, D. De, and S. K. Das, "Efficient aerial data collection with uav in large-scale wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 11, no. 11, 2015.

[31] E. Semsch, M. Jakob, D. Pavlicek, and M. Pechoucek, "Autonomous uav surveillance in complex urban environments," in *IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, vol. 2, 2009, pp. 82–85.

[32] M. T. Perks, A. J. Russell, and A. R. Large, "Advances in flash flood monitoring using unmanned aerial vehicles (uavs)," *Hydrology and Earth System Sciences*, vol. 20, no. 10, pp. 4005–4015, 2016.

[33] C. Yuan, Y. Zhang, and Z. Liu, "A survey on technologies for automatic forest fire monitoring, detection, and fighting using unmanned aerial vehicles and remote sensing techniques," *Canadian Journal of Forest Research*, vol. 45, no. 7, pp. 783–792, 2015.

[34] G. Pajares, "Overview and current status of remote sensing applications based on unmanned aerial vehicles (uavs)," *Photogrammetric Engineering & Remote Sensing*, vol. 81, no. 4, pp. 281–329, 2015.

[35] C. Deng, S. Wang, Z. Huang, Z. Tan, and J. Liu, "Unmanned aerial vehicles for power line inspection: A cooperative way in platforms and communications," *Journal of Communications*, vol. 9, no. 9, pp. 687–692, 2014.

[36] G Morgenthal and N Hallermann, "Quality assessment of unmanned aerial vehicle (uav) based visual inspection of structures," *Advances in Structural Engineering*, vol. 17, no. 3, pp. 289–302, 2014.

[37] P. B. Quater, F. Grimaccia, S. Leva, M. Mussetta, and M. Aghaei, "Light unmanned aerial vehicles (uavs) for cooperative inspection of pv plants," *IEEE Journal of Photovoltaics*, vol. 4, no. 4, pp. 1107–1113, 2014.

[38] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1258 –1276, 2013.

[39] H. Choset and P. Pignon, "Coverage path planning: The boustrophedon cellular decomposition," in *Field and Service Robotics*, 1998, pp. 203–209.

[40] R. Bähnemann, N. R. J. Lawrance, J. J. Chung, M. Pantic, R. Siegwart, and J. I. Nieto, *Revisiting boustrophedon coverage path planning as a generalized traveling salesman problem*, 2019. arXiv: `1907.09224 [cs.RO]`.

[41] I. Colomina and P. Molina, "Unmanned aerial systems for photogrammetry and remote sensing: A review," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 92, pp. 79–97, 2014.

[42] J Fernández-Hernandez, D González-Aguilera, P Rodríguez-Gonzálvez, and J Mancera-Taboada, "Image-based modelling from unmanned aerial vehicle (uav) photogrammetry: An effective, low-cost tool for archaeological applications," *Archaeometry*, vol. 57, no. 1, pp. 128–145, 2015.

[43] Y. Lin, J. Hyyppa, and A. Jaakkola, "Mini-uav-borne lidar for fine-scale mapping," *IEEE Geoscience and Remote Sensing Letters*, vol. 8, no. 3, pp. 426–430, 2011.

[44] C. Rego, D. Gamboa, F. Glover, and C. Osterman, "Traveling salesman problem heuristics: Leading methods, implementations and latest advances," *European Journal of Operational Research*, vol. 211, no. 3, pp. 427 –441, 2011.

[45] G. Gutin and A. P. Punnen, *The traveling salesman problem and its variations*. Springer Science & Business Media, 2006, vol. 12.

[46] P. Vansteenwegen, W. Souffriau, and D. V. Oudheusden, "The orienteering problem: A survey," *European Journal of Operational Research*, vol. 209, no. 1, pp. 1–10, 2011.

[47] A. Gunawan, H. C. Lau, and P. Vansteenwegen, "Orienteering problem: A survey of recent variants, solution approaches and applications," *European Journal of Operational Research*, vol. 255, no. 2, pp. 315–332, 2016.

[48] R. Ramesh, Y.-S. Yoon, and M. H. Karwan, "An optimal algorithm for the orienteering tour problem," *ORSA Journal on Computing*, vol. 4, no. 2, pp. 155–165, 1992.

[49] M. Fischetti, J. J. S. González, and P. Toth, "Solving the orienteering problem through branch-and-cut," *INFORMS Journal on Computing*, vol. 10, no. 2, pp. 133–148, 1998.

[50] G. Kobeaga, M. Merino, and J. A. Lozano, "An efficient evolutionary algorithm for the orienteering problem," *Computers & Operations Research*, vol. 90, pp. 42 –59, 2018.

[51] V. Campos, R. Martí, J. Sánchez-Oro, and A. Duarte, "Grasp with path relinking for the orienteering problem," *Journal of the Operational Research Society*, vol. 65, no. 12, pp. 1800–1813, Dec. 2014.

[52] Yun-Chia Liang, S. Kulturel-Konak, and A. E. Smith, "Meta heuristics for the orienteering problem," in *Congress on Evolutionary Computation*, vol. 1, May 2002, pp. 384–389.

[53] Z. Sevkli and F. E. Sevilgen, "Variable neighborhood search for the orienteering problem," in *Computer and Information Sciences*, 2006, pp. 134–143.

[54] N. Mladenović and P. Hansen, "Variable neighborhood search," *Computers & Operations Research*, vol. 24, no. 11, pp. 1097–1100, 1997.

[55] P. Hansen and N. Mladenović, "Variable neighborhood search: Principles and applications," *European Journal of Operational Research*, vol. 130, no. 3, pp. 449 –467, 2001.

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

[57] D. Feillet, P. Dejax, and M. Gendreau, "Traveling salesman problems with profits," *Transportation Science*, vol. 39, no. 2, pp. 188–205, May 2005.

[58] H. Chitsaz and S. M. LaValle, "Time-optimal paths for a dubins airplane," in *2007 46th IEEE Conference on Decision and Control*, 2007, pp. 2379–2384.

[59] D. Thakur, M. Likhachev, J. Keller, V. Kumar, V. Dobrokhodov, K. Jones, J. Wurz, and I. Kaminer, "Planning for opportunistic surveillance with multiple robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 5750–5757.

[60] J. Yu, M. Schwager, and D. Rus, "Correlated orienteering problem and its application to persistent monitoring tasks," *IEEE Transactions on Robotics*, vol. 32, no. 5, pp. 1106–1118, Oct. 2016.

[61] *The mohamed bin zayed international robotics challenge (mbzirc) 2017*, citied on 2019-12-04. [Online]. Available: `https://www.mbzirc.com/challenge/2017`.

[62] W. Cook, *Concorde tsp solver*, citied on 2019-12-05. [Online]. Available: `http://www.math.uwaterloo.ca/tsp/concorde.html`.

[63] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling-salesman problem," *Operations Research*, vol. 21, no. 2, pp. 498–516, 1973.

[64] K. Helsgaun, "An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic," *European Journal of Operational Research*, vol. 126, no. 1, pp. 106–130, 2000.

[65] K. Savla, E. Frazzoli, and F. Bullo, "Traveling salesperson problems for the dubins vehicle," *IEEE Transactions on Automatic Control*, vol. 53, no. 6, pp. 1378–1391, 2008.

[66] ——, "On the point-to-point and traveling salesperson problems for dubins' vehicle," in *American Control Conference*, 2005, 786–791 vol. 2.

[67] G. Laporte, H. Mercure, and Y. Nobert, "Generalized travelling salesman problem through n sets of nodes: The asymmetrical case," *Discrete Applied Mathematics*, vol. 18, no. 2, pp. 185–197, 1987.

[68] C. E. Noon and J. C. Bean, "An efficient transformation of the generalized traveling salesman problem," *INFOR: Information Systems and Operational Research*, vol. 31, no. 1, pp. 39–44, 1993.

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

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

[71] D. J. Gulczynski, J. W. Heath, and C. C. Price, "The close enough traveling salesman problem: A discussion of several heuristics," in. Boston, MA: Springer US, 2006, pp. 271–283.

[72] J. T. Isaacs and J. P. Hespanha, "Dubins traveling salesman problem with neighborhoods: A graph-based approach," *Algorithms*, vol. 6, no. 1, pp. 84–99, 2013.

[73] K. J. Obermeyer, P. Oberlin, and S. Darbha, "Sampling-based roadmap methods for a visual reconnaissance uav," in *AIAA Conference on Guidance, Navigation, and Control*, 2010.

[74] K. J. Obermeyer, "Path planning for a uav performing reconnaissance of static ground targets in terrain," in *AIAA Conference on Guidance, Navigation, and Control*, 2009, pp. 10–13.

[75] D. Guimaraes Macharet, A. Alves Neto, V. Fiuza da Camara Neto, and M. Montenegro Campos, "An evolutionary approach for the Dubins' traveling salesman problem with neighborhoods," in *Conference on Genetic and Evolutionary Computation*, 2012, pp. 377–384.

[76] T. Bektas, "The multiple traveling salesman problem: An overview of formulations and solution procedures," *Omega*, vol. 34, no. 3, pp. 209–219, 2006.

[77] P. M. França, M. Gendreau, G. Laporte, and F. M. Müller, "The m-traveling salesman problem with minmax objective," *Transportation Science*, vol. 29, no. 3, pp. 267–275, 1995.

[78] S. C. Sarin, H. D. Sherali, J. D. Judd, and P.-F. J. Tsai, "Multiple asymmetric traveling salesmen problem with and without precedence constraints: Performance comparison of alternative formulations," *Computers & Operations Research*, vol. 51, pp. 64 –89, 2014.

[79] Y. GuoXing, "Transformation of multidepot multisalesmen problem to the standard travelling salesman problem," *European Journal of Operational Research*, vol. 81, no. 3, pp. 557 –560, 1995.

[80] P. Junjie and W. Dingwei, "An ant colony optimization algorithm for multiple travelling salesman problem," in *Conference on Innovative Computing, Information and Control*, vol. 1, 2006, pp. 210–213.

[81] J. Li, Q. Sun, M. Zhou, and X. Dai, "A new multiple traveling salesman problem and its genetic algorithm-based solution," in *IEEE International Conference on Systems, Man, and Cybernetics*, 2013, pp. 627–632.

[82] S. Somhom, A. Modares, and T. Enkawa, "Competition-based neural network for the multiple travelling salesmen problem with minmax objective," *Computers & Operations Research*, vol. 26, no. 4, pp. 395–407, 1999.

[83] B. Soylu, "A general variable neighborhood search heuristic for multiple traveling salesmen problem," *Computers & Industrial Engineering*, vol. 90, pp. 390–401, 2015.

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

[85] D. G. Macharet, J. W. Monteiro, G. R. Mateus, and M. F. Campos, "Time-Optimized Routing Problem for Vehicles with Bounded Curvature," in *Latin American Robotics Symposium and Brazilian Robotics Symposium*, Oct. 2016, pp. 145–150.

[86] J. Faigl, "Gsoa: Growing self-organizing array - unsupervised learning for the close-enough traveling salesman problem and other routing problems," *Neurocomputing*, vol. 312, pp. 120 –134, 2018.

[87] S. M. Lavalle, "Rapidly-exploring random trees: A new tool for path planning," Iowa State University, Tech. Rep., 1998.

[88] L. E. Kavraki, P. Svestka, J. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, Aug. 1996.

[89] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[90] D. Perez, P. Rohlfshagen, and S. M. Lucas, "The physical travelling salesman problem: Wcci 2012 competition," in *IEEE Congress on Evolutionary Computation*, 2012, pp. 1–8.

[91] D. Devaurs, T. Siméon, and J. Cortés, "A multi-tree extension of the transition-based rrt: Application to ordering-and-pathfinding problems in continuous cost spaces," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 2991–2996.

[92] J. McMahon and E. Plaku, "Autonomous underwater vehicle mine countermeasures mission planning via the physical traveling salesman problem," in *MTS/IEEE OCEANS*, 2015, pp. 1–5.

[93] S. Edelkamp, M. Lahijanian, D. Magazzeni, and E. Plaku, "Integrating temporal reasoning and sampling-based motion planning for multigoal problems with dynamics and time windows," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3473–3480, Oct. 2018.

[94] J. McMahon and E. Plaku, "Mission and motion planning for autonomous underwater vehicles operating in spatially and temporally complex environments," *IEEE Journal of Oceanic Engineering*, vol. 41, no. 4, pp. 893–912, 2016.

[95] ——, "Autonomous data collection with limited time for underwater vehicles," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 112–119, 2017.

[96] P. Vansteenwegen, *The Orienteering Problem: Test Instances – Department of Mechanical Engineering, University of Leuven*, citied on 2019-12-05. [Online]. Available: `http://www.mech.kuleuven.be/en/cib/op\#section-2`.

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

[98] P. Váňa and J. Faigl, "Optimal solution of the generalized dubins interval problem," in *Robotics: Science and Systems*, Jun. 2018.

Appendix A

Publications of the author

Below are listed all author's publications. Each publication is displayed with the number of citations based on Web of Science (WoS), Scopus, and Google Scholar (GS). The percentage contribution of the author on the core publications is displayed as agreed upon publication. In other publications, the contributions of all co-authors are considered as equal. Journal articles also contain information about the Impact Factor (IF) by Thomson Reuters and the CiteScore (CS) by Scopus. The publications [1c], [5c], [7a] are only reported with CS due to the novelty of journal that is expected to receive IF in June 2020. The core publications of the thesis are referenced with [*c] and other author's publications with [*a].

A.1 Thesis-related publications

A.1.1 Thesis core journal publications

Thesis core publications in peer-reviewed journals with Impact Factor

[2c] **R. Pěnička**, J. Faigl, M. Saska, and P. Váňa, "Data collection planning with non-zero sensing distance for a budget and curvature constrained unmanned aerial vehicle," *Autonomous Robots*, vol. 43, no. 8, pp. 1937 –1956, 2019,
65% contribution, IF 3.634, CS 4.21, citations: 0 in WoS, 2 in Scopus, 13 in GS.

[3c] **R. Pěnička**, J. Faigl, and M. Saska, "Variable neighborhood search for the set orienteering problem and its application to other orienteering problem variants," *European Journal of Operational Research*, vol. 276, no. 3, pp. 816 –825, 2019,
70% contribution, IF 3.806, CS 4.98, citations: 2 in WoS, 2 in Scopus, 14 in GS.

[4c] J. Faigl, P. Váňa, **R. Pěnička**, and M. Saska, "Unsupervised learning-based flexible framework for surveillance planning with aerial vehicles," *Journal of Field Robotics*, vol. 36, no. 1, pp. 270–301, 2019,
20% contribution, IF 4.345, CS 5.70, citations: 1 in WoS, 4 in Scopus, 6 in GS.

Thesis core publications in peer-reviewed journals with CiteScore

[1c] **R. Pěnička**, J. Faigl, P. Váňa, and M. Saska, "Dubins orienteering problem," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1210–1217, 2017,
55% contribution, CS 4.56, citations: 22 in WoS, 26 in Scopus, 64 in GS.

[5c] **R. Pěnička**, J. Faigl, and M. Saska, "Physical orienteering problem for unmanned aerial vehicle data collection planning in environments with obstacles," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 3005–3012, 2019,
70% contribution, CS 4.56, citations: 1 in Scopus, 12 in GS.

A.1.2 Other thesis-related publications

Other thesis-related articles in peer-reviewed journals with Impact Factor

[6a] V. Spurný, T. Báča, M. Saska, **R. Pěnička**, T. Krajník, J. Thomas, D. Thakur, G. Loianno, and V. Kumar, "Cooperative autonomous search, grasping, and delivering in a treasure hunt scenario by a team of unmanned aerial vehicles," *Journal of Field Robotics*, vol. 36, no. 1, pp. 125–148, 2019,
IF 4.345, CS 5.70,citations: 5 in WoS, 6 in Scopus, 35 in GS.

Other thesis-related articles in peer-reviewed journals with CiteScore

[7a] G. Loianno, V. Spurný, J. Thomas, T. Báča, D. Thakur, D. Heřt, **R. Pěnička**, T. Krajník, A. Zhou, A. Cho, M. Saska, and V. Kumar, "Localization, grasping, and transportation of magnetic objects by a team of mavs in challenging desert-like environments," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1576–1583, 2018,
CS 4.56, citations: 21 in Scopus, 47 in GS.

Thesis-related conference proceedings in WoS

[8a] J. Faigl, **R. Pěnička**, and G. Best, "Self-organizing map-based solution for the orienteering problem with neighborhoods," in *IEEE International Conference on Systems, Man, and Cybernetics*, 2016, pp. 1315–1321,
citations: 9 in WoS, 17 in Scopus, 21 in GS.

[9a] **R. Pěnička**, J. Faigl, P. Váňa, and M. Saska, "Dubins orienteering problem with neighborhoods," in *International Conference on Unmanned Aircraft Systems*, 2017, pp. 1555–1562,
citations: 9 in WoS, 12 in Scopus, 39 in GS.

[10a] J. Faigl and **R. Pěnička**, "On close enough orienteering problem with dubins vehicle," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017, pp. 5646–5652,
citations: 5 in WoS, 8 in Scopus, 11 in GS.

[11a] A. Dubeň, **R. Pěnička**, and M. Saska, "Information gathering planning with hermite spline motion primitives for aerial vehicles with limited time of flight," in *Modelling and Simulation for Autonomous Systems*, 2019, pp. 172–201,
citations: 0 in WoS, 0 in Scopus, 0 in GS.

[12a] P. Váňa, J. Faigl, J. Sláma, and **R. Pěnička**, "Data collection planning with dubins airplane model and limited travel budget," in *European Conference on Mobile Robots*, 2017, pp. 1–6,
citations: 0 in WoS, 0 in Scopus, 1 in GS.

[13a] D. Zahrádka, **R. Pěnička**, and M. Saska, "Route planning for teams of unmanned aerial vehicles using dubins vehicle model with budget constraint," in *Modelling and Simulation for Autonomous Systems*, 2019, pp. 365–389,
citations: 0 in WoS, 0 in Scopus, 0 in GS.

[14a] V. Vonásek and **R. Pěnička**, "Space-filling forest for multi-goal path planning," in *International Conference on Emerging Technologies and Factory Automation*, 2019, pp. 1587–1590,
citations: 0 in WoS, 0 in Scopus, 0 in GS.

[15a] **R. Pěnička**, M. Saska, C. Reymann, and S. Lacroix, "Reactive dubins traveling salesman problem for replanning of information gathering by uavs," in *European Conference on Mobile Robots*, 2017, pp. 1–6,
citations: 0 in WoS, 2 in Scopus, 25 in GS.

A.2 Unrelated publications

Articles in peer-reviewed journals with Impact Factor

[16a] T. Báča, P. Štěpán, V. Spurný, D. Heřt, **R. Pěnička**, M. Saska, J. Thomas, G. Loianno, and V. Kumar, "Autonomous landing on a moving vehicle with an unmanned aerial vehicle," *Journal of Field Robotics*, vol. 36, no. 5, pp. 874–891, 2019,
IF 4.345, CS 5.70, citations: 1 in WoS, 3 in Scopus, 22 in GS.

Conference proceedings in WoS

[17a] A. Vick, V. Vonásek, **R. Pěnička**, and J. Krüger, "Robot control as a service — towards cloud-based motion planning and control for industrial robots," in *International Workshop on Robot Motion and Control*, 2015, pp. 33–39,
citations: 18 in WoS, 25 in Scopus, 35 in GS.

[18a] V. Vonásek and **R. Pěnička**, "Computation of approximate solutions for guided sampling-based motion planning of 3d objects," in *International Workshop on Robot Motion and Control*, 2019, pp. 231–238,
citations: 0 in WoS, 0 in Scopus, 0 in GS.

[19a] ——, "Path planning of 3d solid objects using approximate solutions," in *International Conference on Emerging Technologies and Factory Automation*, 2019, pp. 593–600,
citations: 0 in WoS, 0 in Scopus, 0 in GS.