CZECH TECHNICAL UNIVERSITY IN PRAGUE

Faculty of Electrical Engineering
Department of Computer Science

Bachelor's Thesis

Hybrid algorithm for the Dubins Traveling Salesman Problem with Neighborhoods

Daniel Váchal

May 2019

Supervisor: doc. Ing. Jan Faigl, Ph.D.

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Váchal** Jméno: **Daniel** Osobní číslo: **420913**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Otevřená informatika**

Studijní obor: **Softwarové systémy**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Hybridní řešení úlohy obchodního cestujícího s Dubinsovým vozidlem

Název bakalářské práce anglicky:

Hybrid algorithm for the Dubins Traveling Salesman Problem with Neighborhoods

Pokyny pro vypracování:

1. Familiarize yourself with the Dubins Traveling Salesman Problem (DTSP) [1] and the DTSP with Neighborhoods (DTSPN) [2, 3].
2. Familiarize yourself with the algorithms [4] and [5] and available implementations.
3. Propose a combination of the memetic techniques [4] with the unsupervised learning [5] to improve the quality of solutions provided by [5] and decrease computational requirements of [4].
4. Implement the proposed solution and evaluate its performance in representative benchmarks of the DTSP(N).

Seznam doporučené literatury:

[1] Savla, K., Frazzoli, E., and Bullo, F: On the point-to-point and traveling salesperson problems for Dubins? vehicle. American Control Conference, 2005, 786-791.
[2] Oberlin, P., Rathinam, S., and Darbha, S.: Today's traveling salesman problem. Robotics & Automation Magazine, 2010, 17(4):70-77.
[3] Isaacs, J. T., Klein, D. J., and Hespanha, J. P. Algorithms for the Traveling Salesman Problem with Neighborhoods Involving a Dubins Vehicle. American Control Conference, 2011, 1704-1709.
[4] Zhang, X., Chen, J., Xin, B., and Peng, Z.: A memetic algorithm for path planning of curvature-constrained uavs performing surveillance of multiple ground targets. Chinese Journal of Aeronautics, 2014, 27(3):622-633.
[5] Faigl, J., Váňa, P.: Unsupervised learning for surveillance planning with team of aerial vehicles. IJCNN 2017: 4340-4347.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

doc. Ing. Jan Faigl, Ph.D., centrum umělé inteligence FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **31.01.2018** Termín odevzdání bakalářské práce: **24.05.2019**

Platnost zadání bakalářské práce: **30.09.2019**

doc. Ing. Jan Faigl, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Ing. Pavel Ripka, CSc. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací.
Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

.

Datum převzetí zadání

Podpis studenta

" Time is the kindest thing of all, it will eventually heal every sorrow. Time is the cruelest thing of all, it will make everything fade away."

-TOKISAKI KURUMI, DATE A LIVE (ENCORE OVA)

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze 24. května 2019

Acknowledgement

My thank you belongs to doc. Ing. Jan Faigl, Ph.D. and Ing. Petr Váňa for not giving up on me, even though i did not make it easy at times. I would also like to thank my family for supporting me and keeping me alive. And i would like to say my thank you to Ondřej Mádlík for his very supportive approach to me.

Abstract

The thesis presents to the reader the Dubins Traveling Problem (DTSP) and its extension to the Dubins Traveling Salesman Problem with Neighborhoods (DTSPN). The thesis focuses on solving the DTSPN and studies two existing algorithms for this problem. One algorithm is the Unsupervised learning algorithm based on Self-Organizing maps (SOM). The second studied algorithm is a Memetic algorithm presenting crossover and mutation operators to solve the DTSPN. The advantages and disadvantages of these two approaches can be considered complementary. Based on that, a novel Hybrid algorithm for solving the DTSPN is proposed, combining the two studied algorithms into one. Adopting the advantages of both.

Keywords: dubins traveling salesman problem; dubins traveling salesman problem with neighborhoods; memetic algorithm; self-organizing map; hybrid algorithm; unsupervised learning

Abstrakt

Tato práce pojednává o problematice Problému obchodního cestujícího s Dubinsovým Vozidlem (DTSP) a jeho rozšíření na Problém obchodního cestujícího s Dubinsovým vozidlem s okolími (DTSPN). Práce se zaměřuje na řešení DTSPN a studuje dva existující algoritmy pro řešení tohoto problému. Algoritmus využívajícího strojového učení postavený na Samo-Organizačních mapách (SOM) a druhý Memetický algoritmus, který k řešení daného problému využívá proces křížení a mutací. Výhody a nevýhody těchto algoritmů se navzájem doplňují, čehož je v práci využito pro navrhnutí hybridního algoritmu pro řešení DTSPN. Tento algoritmus kombinuje výhody obou studovaných.

Klíčová slova: problém obchodního cestujícího s Dubinsovým vozidlem; problém obchodního cestujícího s Dubinsovým vozidlem s okolími ; memetický algoritmus; samo-organizační mapy; hybridní algoritmus; strojové učení

Contents

List of Figures

List of Algorithms

Abbreviations

UAV	Unmanned Aerial Vehicle
TSP	Traveling Salesman Problem
TSPN	Traveling Salesman Problem with Neighborhoods
DTSP	Dubins Traveling Salesman Problem
DTSPN	Dubins Traveling Salesman Problem with Neighborhoods
SOM	Self-Organizing Map
GSOA	Growing Self-Organizing Array
DTP	Dubins Touring Problem
AA	Alternating Algorithm
ATSP	Asymmetric Traveling Salesman Problem
ETSP	Euclidean Traveling Salesman Problem
GTSP	Generalised Traveling Salesman Problem
GTSPN	Generalised Traveling Salesman Problem with Neighborhoods

Symbols Used

ρ	Turning radius limit
n	Natural number
δ	Sensing radius
v	Constant forward velocity
q	State of the Dubins vehicle
p	Position of the Dubins Vehicle in $\mathbb{R}^2$ (waypoint)
(x, y)	Coordinates in $\mathbb{R}^2$
θ	Heading of the Dubins vehicle
$SE(2)$	Set of all possible states of the Dubins Vehicle
S	Sequence of waypoints s_i
P	Set of waypoints p_i
u	Control input
O	Target location
r	Vehicle number
N	Neural network (neuron ring)
v_i	Vehicle configuration in the input space
m	Number of neurons in the ring
σ	Learning gain
μ	Learning rate
α	Gain decreasing rate
v^*	Winner neuron
Θ_v	Set of heading values of the neuron v
p_o	Closest point on the Dubins path to the target o
v_{prev}	Neuron on the path previous to the winner neuron determined so that the length of the Dubins path is minimized
v_{next}	Neuron on the path next to the winner neuron determined so that the length of the Dubins path is minimized

CHAPTER 1

Introduction

The problem addressed in this thesis is motivated by surveillance missions performed by the Unmanned Aerial Vehicles (UAVs). In surveillance missions, the goal is to visit a set of locations to gather information on the objects of interest. An example can be data collection of households energy consumption, where data can be collected remotely from measuring devices. Another instance of a surveillance mission is taking snapshots of the given target locations. In that case, it is not necessary to visit the location of the object exactly, but it is sufficient to reach a location from which the object can be photographed with the requested level of details. There can be several optimization criteria used in surveillance mission planning. The most straightforward way is to shorten the time UAV spends on the mission because keeping the UAV airborne is the most expensive part of the mission. Therefore the mission needs to be planned such that the UAV visits all the required target locations with minimal possible time. If the UAV is moving with constant speed, the problem can be considered as minimization of the total travel cost to visit all the target locations. A common type of the UAV used for the surveillance missions is a fixed-wing aircraft, which is constrained by its turning radius. An example of such fixed-wing aircraft is shown in Figure 1.1.

The problem of finding minimal tour length visiting all the given locations is the Traveling Salesman Problem (TSP), which is known to be NP-hard (in its decision variant) [5], several

Figure 1.1: Carbonix Volanti: fixed wing carbon composite industrial drone, source: [1]

approaches have been proposed [6], [7]. However, the motivational scenario of the problem studied in this thesis is to find a smooth multi-goal trajectory that is suitable for UAVs such as fixed-wing aircraft, which is constrained by its minimal turning radius. Therefore, we focus on a solution of the TSP-like problems with curvature-constrained trajectories for which we consider the vehicle motion constraints modeled as Dubins vehicle [8].

Dubins vehicle models a vehicle that moves only forward with a constant speed and is limited by minimum turning radius ρ. Then, the TSP becomes the Dubins Traveling Salesman Problem (DTSP) [9]. The task is to find the shortest path between given set of points with a curvature constraint, such that the path visits all the given locations. In case, when it is not required to visit the exact locations, we can save additional resources by only visiting their vicinity. That leads to a generalization of the DTSP, where particular waypoints can be chosen from an area surrounding the object of interest. This generalization is called the Dubins Traveling Salesman Problem with Neighborhoods (DTSPN) [10].

A novel hybrid algorighm for the DTSPN is introduced in this thesis. It leverages on to the existing methods: Memetic algorithm [3] and Unsupervised Learning algorighm [4]. The Unsupervised Learning algorithm is based on Self-Organizing Maps (SOM). The algorithm has evolved and the authors have introduced an updated version, called GSOA: Growing Self-Organizing Array - Unsupervised learning for the Close-Enough Traveling Salesman Problem and other routing problems. But we use the original abbrevation (SOM) for the rest of the thesis.

Both of these algorithms have their pros and cons. But these pros and cons can be considered complementary to each other. While the SOM algorithm can be considered a quick constructive heuristic, it does not improve the provided solution with more computational time at its disposal. The memetic algorithm is relatively slow in providing the first competitive solution, but with enough computational time, it can converge to high-quality solutions, eventually to the optimum. These algorithms are presented to the reader in detail, as they are essential for the rest of the thesis.

Chapter 6 presents the proposed hybrid algorithm, combining the benefits of selected approaches. The proposed algorithm is a combination of the SOM and the Memetic algorithm. The solution provided by the SOM is used as initialization of the Memetic algorithm. That way, the first competitive solution can be provided relatively quickly, while it can be improved with more computational time.

In chapter 7, results of proposed solutions are reported and discussed. The proposed hybrid algorithm is compared to the two algorithms from which it was developed. In the last part, the thesis is concluded.

CHAPTER 2

Problem Statement

The problem studied in this thesis is motivated by surveillance missions performed by the UAVs. Where set of locations to visit is given. The problem of connecting all the points in a plane and determining the order of their visits is known as the Traveling Salesman Problem. The target location does not always have to be visited directly. When it is enough to approach the object of interest from a certain distance, then the particular waypoint has to be chosen from an area surrounding the target. With this generalization, the problem becomes the Traveling Salesman Problem with Neighborhoods (TSPN). The shape of this area can vary in real situations, but for the computation, it has to be determined first. For simplicity, this thesis always considers the area around each target to be a disk centered around the target location with a given radius δ. In our case, each location has to be visited by the UAV, the curvature constrained non-holonomic vehicle.

To model the behavior of the UAV, a mathematical model is needed for computations. This model has been proposed in [8]. The model is called the Dubins vehicle, which is a vehicle moving always forward with constant velocity v and is limited by its minimal turning radius ρ. At each point in time, the state of the vehicle is described by its position in the plane and its heading. The represantation of the state is $q = (p, \theta)$, where $p \in \mathbb{R}^2$ is the position $p = (x, y)$ and $\theta \in \mathbb{S}^1$ is the heading of the vehicle, i.e., $q \in SE(2)$. The following mathematical representation is brought from [2].

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = v \begin{bmatrix} cos\theta \\ sin\theta \\ u \cdot \rho^{-1} \end{bmatrix}, \quad |u| \leq 1, \tag{2.1}$$

where u is the control input.

Dubins has shown in [8], that the optimal path connecting two states $q_1 \in SE(2)$ and $q_2 \in SE(2)$ is constructed only of the circular segments (C) with maximal possible turning radius and straight line segments (L). Where the circular segments are either turn to the left (L) or to the right (R). Path constructed from these segments connecting two states of the Dubins vehicle is called the Dubins maneuver. The Dubins maneuver can be of two kinds. One composed of only circular segments (CCC) and the other composed of two circular segments connected by a straight line (CSC). The options for the Dubins maneuver are LRL, RLR for the CCC option and LSL, LSR, RSL, RSR for the CSC option. For demonstration, an example of the Dubins maneuver is shown in Figure 2.1.

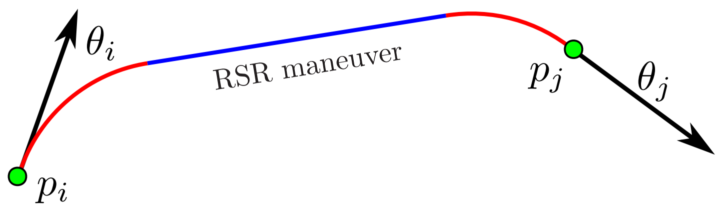


Figure 2.1: An example of the Dubins manuever connecting points p_i and p_j using the departure angle θ_i and arrival angle θ_j.

The problem becomes to find the optimal sequence of the targets to visit as in TSP. Then determine the optimal headings at each target, to be able to connect all the points by a Dubins path, which corresponds to the DTSP. And finally, the Dubins Traveling Problem with Neighborhoods (DTSPN) is the problem of finding the shortest possible curvature-constrained path connecting all the target regions.

Formal introduction of the DTSPN is adopted from [10]. Mathematical representation is:

Problem 1 (DTSPN)

$$\begin{aligned} \textit{minimize}_{\Sigma,q} \quad & \mathcal{L}\left(q_{\sigma_n}, q_{\sigma_1}\right) + \sum_{i=1}^{n-1} \left(q_{\sigma_i}, q_{\sigma_{i+1}}\right) \\ \textit{subject to} \quad & \left|p_i \in R_i, q_i\right| < \delta, \end{aligned}$$

where $\mathcal{R} = (R_1, \ldots, R_n)$ is the set of n regions $R_i \subset \mathbb{R}^2$ to be visited by the Dubins vehicle. $\Sigma = (\sigma_1, \ldots, \sigma_n)$ is the ordered permutation of $\{1, \ldots, n\}$. p_i is the visiting point of the region R_i and $q_i \in SE\left(2\right)$ is the state of the Dubins vehicle. And δ is sensing radius. The DTSPN is optimization problem over all possible Σ and all configurations q, where $\mathcal{L}\left(q_i, q_j\right)$ is the Dubins distance between q_i and q_j.

CHAPTER 3

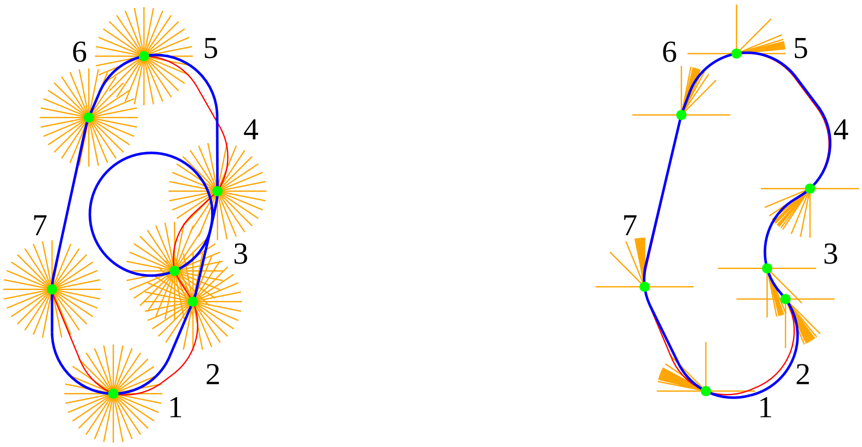
Related Work

In this chapter, existing approaches for the DTSP and the DTSPN are discused. As Dubins shown in [8] the shortest path for Dubins vehicle connecting two points in a plane is one of the six Dubins maneuvers consisting only of straight line segments and arc curves with minimal turning radius. However, the path expects that headings of the vehicle are known for both points connected by the Dubins maneuver. In case of planning the path for the DTSP, the headings at waypoints are unknown. Thus, the solutions for the Euclidean TSP cannot be applied directly, and the problem of finding the optimal headings for each waypoint needs to be solved. For the sequence of n waypoint locations $p_1 \dots p_n$, the problem to find optimal headings $\theta_1 \dots \theta_n$ at each waypoint, in order to minimize the total length of the Dubins path connecting all waypoints is a continuous optimization problem, known as the Dubins Touring Problem (DTP).

One of the first solutions to the DTP is Alternating Algorithm (AA) described in [11]. Next approach to the DTP is used in [12], where the headings are determined using uniform sampling by straight line segments. First for even edges and then optimal Dubins maneuvers are determined for odd edges. Another solution was proposed in [2], which brings a refinement procedure to create informed sampling method for solving the DTP. The comparison between the informed sampling and uniform sampling is shown in Figure 3.1. Using informed sampling method the authors claim to be able to find a solution as close to optimum as 0.1 %.

Approaches to solving the DTSP and the DTSPN found in literature can be divided into four main groups of algorithms. The first group is decoupled approaches, which solve the problem of determining the headings and the problem of finding the sequence separately. The second group is formed by transformation approach algorithms where the idea is to determine the headings first and then transform the problem to Asymmetric TSP (ATSP). The third group is evolutionary algorithms with high computational requirements but with a chance of providing solutions of high quality. The fourth group is the approaches that use unsupervised learning and self-organizing maps.

Decoupled approach to the DTSP is presented in [11], where the authors first obtain the upper bound on the point to point problem. And then consider the corresponding TSP. The effectiveness of decoupling methods mainly relies on the similarities between the DTSP and the ETSP. This makes them unsuitable for situations, where the Euclidean distance between waypoints is not long enough compared to the minimal turning radius.



(a) Uniform sampling

(b) Informed sampling

Figure 3.1: A solution for the DTSP for a given sequence of the targets, using the uniformed sampling and the informed sampling. Source of the picture: [2]

An example of the transformation approach to the DTSP can be a graph-based algorithm presented in [13]. Here, the authors use a sampling method to cast the DTSPN to the Generalised Traveling Salesman Problem (GTSP) with intersecting node sets which can be described by a directed graph. And then, using the Noon-Bean transformation to transform the GTSP into ATSP, they recover the optimal solution for the GTSPN from the solution of the ATSP. Transformation methods are highly dependant on the sampling density of headings. To acquire better results, they usually require tremendous computational resources.

The third group represents genetic algorithms, which improve solutions by mutating existing solutions and creating crossbreeds of these solutions in generations. The fourth group is algorithms based on neural networks. We chose two algorithms from these last two groups to create the hybrid algorithm. As these algorithms are essential for this work, they are described in detail in the following chapter.

CHAPTER 4

Source Algorithms

From the known algorithms for solving the DTSPN, two are chosen in this thesis. One is a Memetic algortihm. This algorithm is a representant of the third group of algorithms for the DTSPN. It has been chosen for its ability to improve the provided solution in time. The second chosen algorithm, representant of the fourth group of algorithms, is the Unsupervised learning algorithm. It was chosen for its ability to quickly provide a competitive solution. The proposed Hybrid algortihm combines these two algorithms into one, adopting advantages of both. That is why the thesis focuses on these algorithms in detail, in this chapter.

Memetic Algorithm

The chosen Memetic algortihm has been presented in [3]. This approach to solving the DTSPN first generates a population of random valid solutions to the DTSPN. Then the individuals in the population can be mutated and crosbred, creating new individuals for the next population. The higher quality solutions are kept for the next generations, thus improving the found solution. This process can be repeated for the time provided to the algorithm, eventually converging to high quality solutions. The main specifics of the chosen Memetic algorithm [3] are presented in this section.

The authors introduce several optimization difficulty reductions as the main contributions of the article. The first introduced optimization is terminal heading relaxation. The article shows that reaching the target from an initial state with a fixed position and initial heading always reduces to circular arcs and straight line segments. Also, these paths to the target are symmetric to both sides from the initial position. This is a special situation of the paths with terminal headings studied by Dubins [8]. This improvement allows determining the terminal heading from its relative position to the initial heading, once that is fixed. Therefore only initial heading needs to be found, reducing the overall difficulty of the DTSPN. The article presents and uses boundary based encoding scheme. Using the fact that the UAV has to always pass through the boundary of the target region, this point on the boundary can be used as the visiting point of each region. Using the position of the target as center of the disk area of the neighborhood, every point on the boundary can be described using the polar angle. According to the authors of [3], the fact, that only the sequence of visits of each target and initial heading at the entry point needs to be determined, the overall complexity

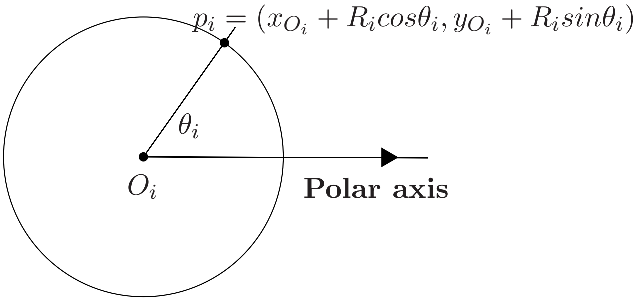


Figure 4.1: Representation of the encoding scheme for the visiting point of the target region. Source [3]

of the DTSPN reduces from $4n$ to $2n$, where n is the number of targets to be visited. The authors also mention the fact that in real situations, the data collection is not instant but can take some time. When the visiting point of each region is on the border, the UAV is not guaranteed to spend enough time in the target region. This problem is solved by calculating with neighborhoods where the diameter of the disk area is reduced by a constant. Scheme for the polar angle represenation of the visiting point is shown in Figure 4.1. Using this encoding scheme, the waypoints $P = (p_1, p_2, \ldots, p_n)$, where $p_i = (x_i, y_i)$ are described. Their sequence is $S = (s_1, s_2, \ldots, s_n)$ and θ_i represents the initial heading at the waypoint. Thus the solution to the DTSPN can be described as $\binom{s_1}{\theta_{s_1}} - \binom{s_2}{\theta_{s_2}} - \cdots - \binom{s_n}{\theta_{s_n}}$, where θ_{s_i} corresponds to the visiting point of the s_ith target region.

Another reduction to the computational difficulty of the DTSPN presented in [3] is an approximate gradient-based search. The authors present that if the visiting sequence of two solutions is the same, then the difference is only in visiting points of each region. While changing the visiting point of one target region affects all the other target regions, the further from the changed point, the effect weakens. Thus to reduce the computational cost, only part of the solution is adapted, when one visiting point is changed.

The evolutionary algorithm has two mutation operators and a crossover. Swapping mutation operator changes the sequence of the visited targets. Randomly choosing two indices $i, j \in 1, 2, \ldots, n, i \neq j$. It swaps the genes in the chromosome. Creating a new solution. For example, in chromosome

$$\begin{pmatrix} 2 \\ \theta_2^i \end{pmatrix} - \begin{pmatrix} 6 \\ \theta_6^i \end{pmatrix} - \begin{pmatrix} 1 \\ \theta_1^i \end{pmatrix} - \begin{pmatrix} 5 \\ \theta_5^i \end{pmatrix} - \begin{pmatrix} 3 \\ \theta_3^i \end{pmatrix} - \begin{pmatrix} 4 \\ \theta_4^i \end{pmatrix},$$

indices $i = 2$ and $j = 5$ are chosen. The operator swaps corresponding genes in the chromosome, creating new chromosome

$$\begin{pmatrix} 5 \\ \theta_5^i \end{pmatrix} - \begin{pmatrix} 6 \\ \theta_6^i \end{pmatrix} - \begin{pmatrix} 1 \\ \theta_1^i \end{pmatrix} - \begin{pmatrix} 2 \\ \theta_2^i \end{pmatrix} - \begin{pmatrix} 3 \\ \theta_3^i \end{pmatrix} - \begin{pmatrix} 4 \\ \theta_4^i \end{pmatrix}.$$

The other mutation operator is shifting the visiting point of the region. Index i is again randomly chosen and the polar angle of the gene is reset within the interval $(0, 2\pi]$. The i-th gene in the chromosome is changed by changing the visiting point of the region.

The crossover operator is very well demonstrated by the example in [3], which is also shown here. Assuming the ith individual in the population to be

$$\begin{pmatrix} 2 \\ \theta_2^i \end{pmatrix} - \begin{pmatrix} 6 \\ \theta_6^i \end{pmatrix} - \begin{pmatrix} 1 \\ \theta_1^i \end{pmatrix} - \begin{pmatrix} 5 \\ \theta_5^i \end{pmatrix} - \begin{pmatrix} 3 \\ \theta_3^i \end{pmatrix} - \begin{pmatrix} 4 \\ \theta_4^i \end{pmatrix}$$

where the superscript i indicates the index of the individual in the population. The parent individuals are constituted by the ith individual and another randomly selected individual from the population.

$$\begin{pmatrix}4\\ \theta_4^i\end{pmatrix} - \begin{pmatrix}6\\ \theta_6^i\end{pmatrix} - \begin{pmatrix}2\\ \theta_2^i\end{pmatrix} - \begin{pmatrix}5\\ \theta_5^i\end{pmatrix} - \begin{pmatrix}3\\ \theta_3^i\end{pmatrix} - \begin{pmatrix}1\\ \theta_1^i\end{pmatrix}$$

Auxiliary vector of the same dimensions as parent individuals is randomly generated with its elements being 1 or 2. In example the vector $v = [1, 2, 2, 1, 1, 2]$. The elements of this vector determine from which parent individual the values will be inherited in the offspring. In this example the first component of the auxiliary is 1, which means that the first gene in parent 1 will be selected to construct the offspring. The offspring:

$$\begin{pmatrix}2\\ \theta_2^i\end{pmatrix} \dots$$

Then, the selected gene is removed from both parents. Removing the component numbered two leaves the parent individuals as
Parent 1:

$$\begin{pmatrix}6\\ \theta_6^i\end{pmatrix} - \begin{pmatrix}1\\ \theta_1^i\end{pmatrix} - \begin{pmatrix}5\\ \theta_5^i\end{pmatrix} - \begin{pmatrix}3\\ \theta_3^i\end{pmatrix} - \begin{pmatrix}4\\ \theta_4^i\end{pmatrix}$$

Parent 2:

$$\begin{pmatrix}4\\ \theta_4^i\end{pmatrix} - \begin{pmatrix}6\\ \theta_6^i\end{pmatrix} - \begin{pmatrix}5\\ \theta_5^i\end{pmatrix} - \begin{pmatrix}3\\ \theta_3^i\end{pmatrix} - \begin{pmatrix}1\\ \theta_1^i\end{pmatrix}$$

The second component of the auxiliary vector is 2, which means the next gene in construction of the offspring will be inherited from parent 2. After this step the generated offspring is:

$$\begin{pmatrix}2\\ \theta_2^i\end{pmatrix} - \begin{pmatrix}4\\ \theta_4^i\end{pmatrix} \dots$$

At the end of this cycle, the fully generated offspring will be:

$$\begin{pmatrix}2\\ \theta_2^i\end{pmatrix} - \begin{pmatrix}4\\ \theta_4^i\end{pmatrix} - \begin{pmatrix}6\\ \theta_6^i\end{pmatrix} - \begin{pmatrix}1\\ \theta_1^i\end{pmatrix} - \begin{pmatrix}5\\ \theta_5^i\end{pmatrix} - \begin{pmatrix}3\\ \theta_3^i\end{pmatrix}$$

This memetic algorithm has been chosen in this thesis mainly for its ability to converge to high-quality solutions when given enough computational time. Its main disadvantage is that first competitive solution cannot be provided quickly. We can change that, by initialising the starting generation with a feasible solution generated by a faster method rather than using random values.

Unsupervised learning algorithm

The second chosen algorithm is the Unsupervised learning algorithm based on Self-Organizing Maps (SOM) [4]. This approach to the DTSPN uses an artificial neural network to solve the DTSPN. The neurons in the network, which is called a neuron ring, represent the state of the Dubins vehicle. The ring is then adapted towards the target locations and once each target is covered by the path connecting the neurons, the final path can be determined. Thus solving the DTSPN. The Unsupervised learning algorithm is able to provide a competitive solution in a short amount of time, which is why it was chosen in the thesis.

In this section, the thesis presents the specifics of the chosen Unsupervised learning algorithm. The approach to solve the DTSP [14] is presented and its extension to the DTSPN. This algorithm implements a method to find the waypoint in the disk area around the target during the selection of the winner neuron. This approach has been first proposed in [15] and then used in [16] and further improved in [17]. The article, in which this algorithm has been proposed [4], also brings a way to solve the DTSPN for multiple vehicles. Using a separate neuron ring for each vehicle and in case any target could be covered by more than one vehicle,it chooses the one with the shortest path in order to minimize the longest path. The approach has been proposed in [18]. The thesis does not focus on this part, as it is not in its scope. The main specifics of the Unsupervised learning algortihm [4] are presented in following sections.

Self Organizing Maps for the TSP and the DTSP

SOM is a type of an artificial neural network using Unsupervised learning to adapt its neurons. It can be used for mapping high-dimensional data into an ordinary low dimensional grid [19]. Therefore it is a very good tool for the visualization of data, that could not be displayed otherwise. It is also used for clustering data and other classification problems. SOM for such types of problems is typically 2D maps. However, SOM for the TSP maps the input space and the targets into the neural network. The output is a one-dimensional array of the output units [20]. The neuron weights and the input signals share the same space. Thus the connected neuron ring represents the path between the target locations. [21].

When using SOM for solving the ETSP, neural network $N = (v_1, \ldots, v_m)$ is created. Where v_i is a neuron represeting the location of the vehicle in the input space $\mathbb{R}^2$. And m is the number of neurons in the ring. When solving the ETSP, the final solution is found by connecting the neurons by straight lines [21]. For the SOM-based solution of the DTSP, the neuron also has to contain the information about the Dubins vehicles heading. Therefore each neuron v_i represents the state of the Dubins vehicle $v_i \in SE(2)$. The final Dubins path has to be constructed by connecting the neurons by Dubins maneuvers. While the final solutions of the ETSP and the DTSP differ in constructing the final path and in what the neurons represent, the same framework is utilized to find the final state of the neural network. To complete the information, the framework is sourced directly from [4].

1. *Initialization*: For n target locations O, create a ring of neurons with randomly initialized weights, e.g., with $2n$ neurons [16]. Initialize the learning parameters as follows: the learning gain $\sigma = 10$, the learning rate $\mu = 0.6$, the gain decreasing rate $\alpha = 0.1$, and set the learning epoch counter $i = 1$.
2. *Randomizing*: Create a random permutation of locations $\Pi(O)$ to avoid local minima.
3. *Learning epoch*: For each $o \in \Pi(O)$
   (a) *Select winner* neuron v^* for o as the best matching neuron i.e., the closest neuron to o.
   (b) *Adapt* v^* and its neighbors to o using the neighbouring function (4.1), i.e., set the neuron weights to the new locations v determined as:
   $$v' = v + \mu f(\sigma, d)(o - v)$$
4. *Ring regeneration*: to improve headings associated with the neurons to optimize the length of the Dubins path represented by the ring. (For solving DTSP and DTSPN)

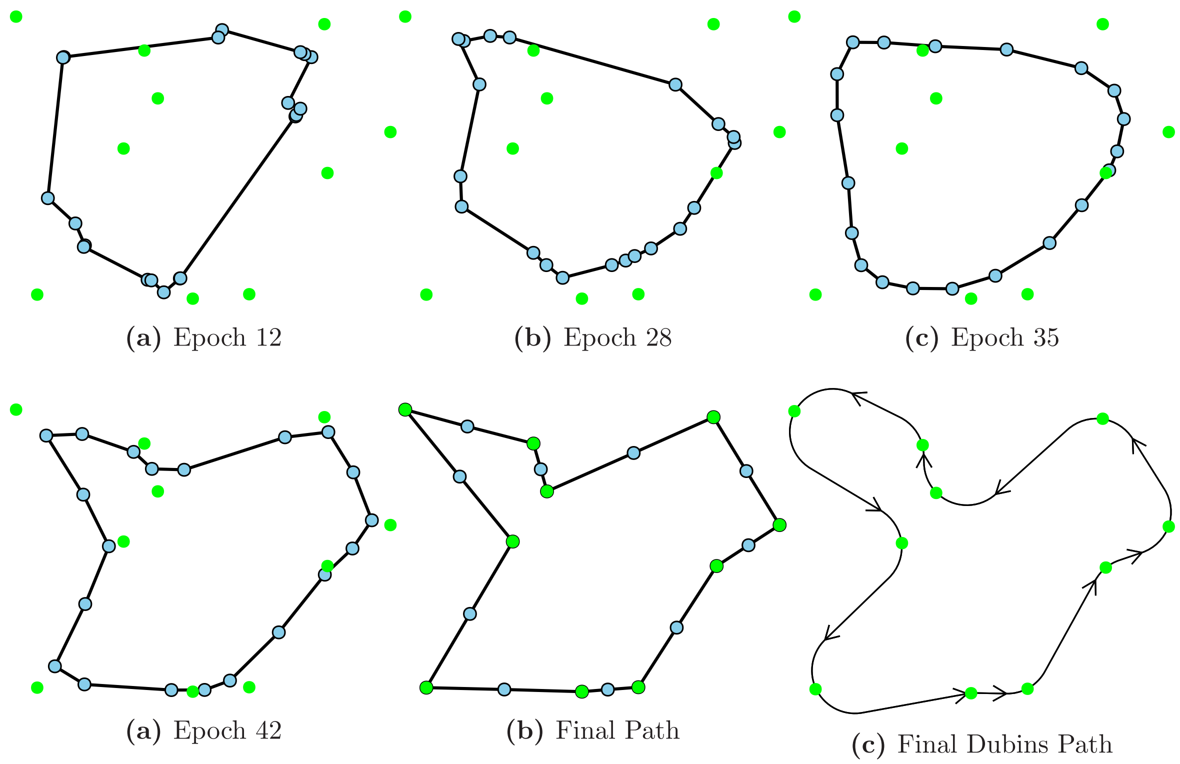


Figure 4.3: Example of the ring evolution towards the targets. After the adaptation process is finished the headings are determined, and the final Dubins path is constructed, source [4]

5. *Update learning parameters*: $\sigma = \sigma(1 - i\alpha), i = i + 1$
6. *Termination condition*: If solution is not improving or $i \geq i_{max}$ Stop the adaptation. Otherwise go to step 2.
7. *Construct the final (Dubins) tour using the last winners*

The used neighbouring function follows the existing SOM for the TSP [21] and it has the following form.

$$f(\sigma, d) = \begin{cases} e^{\frac{-d^2}{\sigma^2}} \text{ for } d < 0.2m^r \\ 0 \text{ otherwise} \end{cases}, \tag{4.1}$$

which decreases the power of adaptation of the neighbouring nodes to the winner neuron v^* with increasing distance d of the neuron to v^* counted in the number of neurons of the ring. The adaptation can be viewed as a movement of the neurons to new position v' which replaces the neuron weights v. The Figure 4.3 shows the evolution of the ring of neurons towards the target locations.

SOM for the DTSPN

In solving the DTSPN, the neurons in the ring represent the waypoints on the Dubins path. Unlike in the SOM-based solution of the TSP, to connect the neurons with Dubins maneuvers into the resulting path, each neuron needs to keep the expected headings [14]. The headings have major effect on the resulting path after connecting the neurons. Thus each neuron $v_i \in N$

holds a set of headings $\Theta_i = \{\theta_i^{-k}, \theta_i^{-k+1}, \ldots, \theta_i^2, \theta_i^k\}$. To find the optimal solution, the heading that matches to the shortest possible path needs to be determined. To acquire this heading the neuron ring can be treated as search graph, where the headings of the neighboring neurons are connected. The heading can be determined from this graph by a feedforward search bound by $O\left(ms^3\right)$, where m is the current number of neurons in the ring and s is the number of headings that each neuron keeps in its structure. The search graph representation is shown in Figure 4.4.

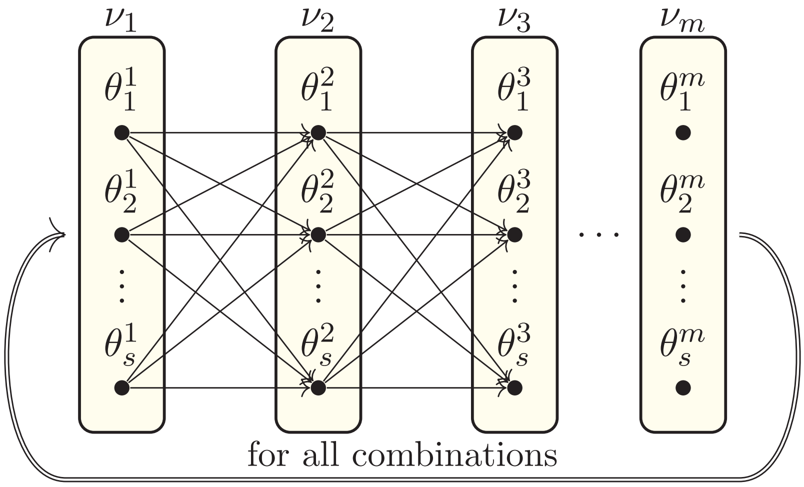


Figure 4.4: A search graph showing how the headings are connected in the neuron ring, source: [4]

The path aquired after this procedure is then used when searching for the winner neuron v^*. The procedure of adapting the network towards a target o used in the algorithm starts by finding the closest point of the current path p_o to the target o. If there is no neuron in this point, new neuron is created and its state is set to this exact location. The vehicle heading θ_p from the point p_o is set as the main heading for the new neuron and the other headings are set around θ_p as $\Theta_{v^*} = \{\theta_p, \theta_p^1, \ldots, \theta_p^i, \theta_p^{-1}, \ldots, \theta_p^{-i}\}$, where $\theta_p^i = \theta_p + i\pi/l, 1 \leq i \leq l$ and l is set to $l = 12$. Next the point o_p is found on the bound of the region of target o as the point on the line connecting point p_o and target o. The point o_p is in the distance equal to the radius δ from o. Then the network adapts towards this point o_p rather than o to save travel distance. In case that the winner neuron already is within the sensing radius δ from o, the network is not adapted at all, as the target can already be covered from p_o. After the adaptation of the winner neuron, the neighboring neurons are adapted as well. The neighborhood is defined by two neurons v_{prev} and v_{next}. These neurons are determined to minimize the expected Dubins path towards o. According to the following equation:

$$L_g = l\left(v_{prev}, (o, \theta)\right) + l\left((o, \theta), v_{next}\right), \tag{4.2}$$

The neurons v_{prev} and v_{next} are determined from equation 4.1. Where θ is one of the headings of the winner neuron v^*. v_{prev} and v_{next} are from the activation bubble around v^*. For the neuron to belong into the activation bubble, its neighbouring function needs to be above the activation limit, which is set to 10^{-5}. As the neighbouring function depends on the learning gain σ, which decreases in the process, the neurons v_{prev} and v_{next} may not be found. In that case, only the winner neuron v^* is adapted towards o_p. Otherwise all neurons between v_{prev} and v_{next} are adapted towards o_p including the winner neuron. The adaptation is made so that the winner neuron v^* is moved to the location of o_p and then

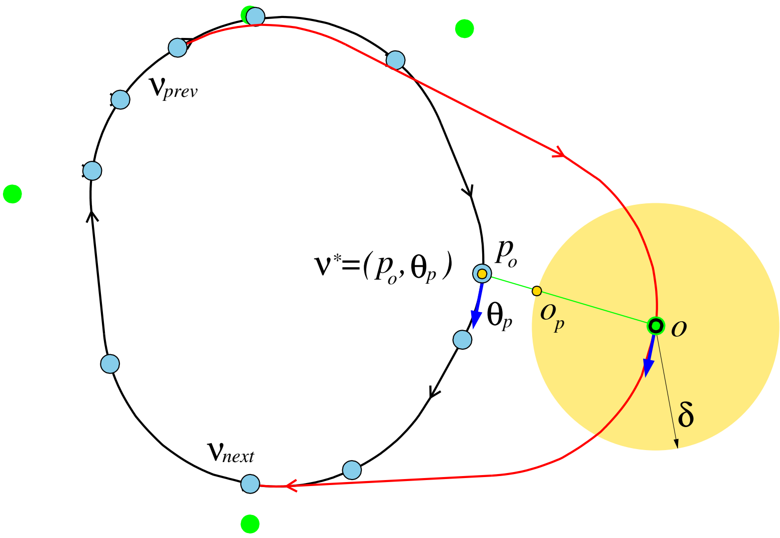


Figure 4.5: Graphic showing the winner selection procedure and the point o_p towards which the network is adapted, source: [4]

Dubins maneuvers are determined between v_{prev}, v^* and v_{next}. If there is such neuron, one neuron is removed between v_{prev} and v_{next} as during the winner selection one neuron could have been added, this procedure ensures that the number of neurons will not exceed $2n$ and limits the computational burden. The process of the winner selection and adaptation of the neuron ring is shown in Figure 4.5.

The Unsupervised learning algorithm has been chosen for its ability to provide a competitive solution quickly. However, its main disadvantage is that it converges to one solution and cannot improve this solution with more computational time at its disposal.

CHAPTER 5

Propsed Hybrid Algorithm

In this chapter, the novel hybrid algorithm for solving the DTSPN is presented. The hybrid algorithm is a combination of the Memetic algorithm proposed in [3] and the Unsupervised learning algorithm proposed in [4]. Details of both algorithms are studied in the previous chapter of the thesis. The proposed hybrid algorithm combines advantages of both its base algorithms to cover for the disadvantages that they have. While the quality of the solution provided by the Memetic algorithm improves with more computational time provided, it can take a long time, before it provides the first competitive solution. This is caused by the Memetic algorithm being initialized by random valid solutions. These random solutions are very ineffective, and it takes time before the evolutionary algorithm improves them. The described disadvantage of the memetic approach is eliminated in the hybrid algorithm by initializing the starting population by solutions provided by the SOM. Using this method, the Hybrid algorithm can provide a competitive solution in a short amount of time as even its initial population contains quality solutions. The comparison of the initial best solutions from the Memetic and the Hybrid algorithms can be seen in Figure 5.1.

The main disadvantage of the Unsupervised learning algorithm is that once the solution is provided, it is final, and this approach alone is not able to improve it any further. While the solutions provided by this algorithm are relatively good, they are not optimal, and we might

(a) Initial solution by Memetic **(b)** Initial solution by Hybrid

Figure 5.1: Ilustration of the best solutions provided by the Memetic algorithm and the Hybrid algorithm in their initial population

want to improve them with more computational resources available. The Hybrid algorithm provides a way to improve the final solutions of Unsupervised learning as it can take several final solutions generated by SOM and use them in the initial population of the evolutionary approach. Using the crossover operator and mutation operators presented in [3], the Hybrid can find further improvements to the solutions found by SOM.

To create the hybrid, the implementations for the two source algorithms has been provided by the supervisor of the thesis. The encoding used in both algorithms is different. The hybrid algorithm presents a function converting the best solution found by SOM into the encoding used my the Memetic algorithm. At first, the initial population needs to be created. To balance the diversity of the initial population and the computational complexity of finding the solutions needed, the Unsupervised learning algorithm is run five times. Setting the best solution found by the Hybrid algorithm to be the best of the five solutions provided by SOM. Then the initial population is initialized. As all the solutions provided by SOM are similar, the Memetic algorithm could get stuck in the local optima and would not be able to provide better solutions. To keep the diversity in the initial population, SOM provided solutions take turns in populating the initial generation, and each time such a solution is added to this generation, it is shadowed by a random solution. After the initialization, the evolutionary algorithm is run for the rest of the computational time provided to improve the best-found solution.

The pseudocode of the Hybrid algorithms is shown here for demonstration:

Algorithm 1: Pseudocode for the Hybrid algorithm

```
Result: Solution of the DTSPN
for i;
where i is the desired number of SOM solutions do
    som = generate SOM solution;
    transform into used encoding;
    if som < best then
        best = som;
    end
    save som for initialization;
end
while initial population < size do
    insert SOM generated solution;
    insert random solution;
end
while has computational time do
    run evolutionary algorithm;
    if better solution found then
        save best solution;
    end
end
```

CHAPTER 6

Results

This chapter consults the results of the proposed Hybrid algorithm in solving the DTSPN. It is compared to the two algorithms from which it was created. This comparison best shows the differences that this approach brings to solving the DTSPN. In each graph, the median provided by the solver is shown, with the 80% confidence interval displayed around the median. The density of targets in each tested instance is 0.03 Every instance used for testing has 30 objectives. All the instances were randomly generated. The first test performed with these three algorithms is on a single instance, and both Memetic and Hybrid algorithms were run ten times on this instance and were given 1 hour to solve the problem. The graph showing the results can be seen in Figure 6.1. This graph shows that the Memetic algorithm takes time to converge to a competitive solution, but once it does converge, the length of the solution is close to the solution provided by the hybrid and is better than the solution generated by SOM. For the detailed view of the changing part of the graph, see Figure 6.2. After 30 seconds, both the Memetic algorithm and the Hybrid algorithm found the solution that remained the best for the rest of the hour. Also, all ten instances of both the Memetic and the Hybrid algorithms gave the same results. Only the results given by SOM vary. As SOM does not improve once provided solution. The median of these solutions is copied to all times for comparison.

To show how the algorithms match in different instances, the second test is shown. In this test, there are used ten randomly generated instances of the DTSPN. Each algorithm was run once and given twenty seconds on each instance. The overall view of the test can be seen in Figure 6.3. This view shows that the Memetic algorithm takes about twenty seconds to provide a solution of similar length to the one provided by SOM. In the detailed view shown in Figure 6.4, the results displayed suggest, that after twenty seconds, the Memetic algorithm provides solution of comparable quality to the SOM. The Hybrid algorithm already has a better solution, that is not further improved in this interval.

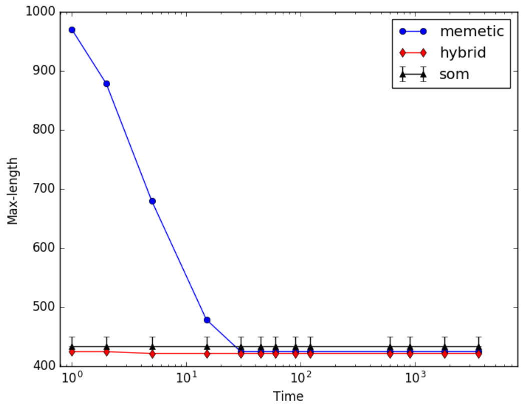


Figure 6.1: The overall result of the three algorithms given one hour to solve a random instance of the DTSPN. On the x axis time is shown in seconds in logaritmic scale. The Unsupervised learning algorithm has been run 10 times on the same problem. While it does not improve the provided solution over time, the solution provided was copied to all times for comparison with the other two algorithms.

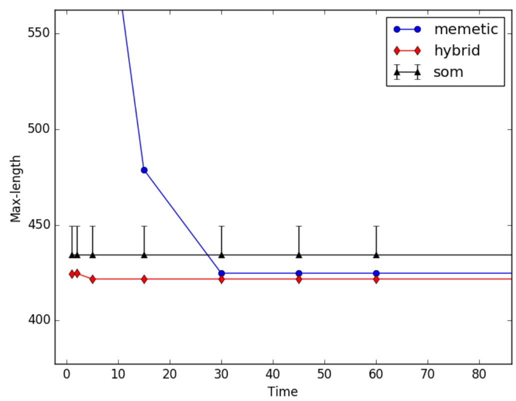


Figure 6.2: Detailed view of the length of the final solutions provided by the compared algorithms. Time displayed in seconds.

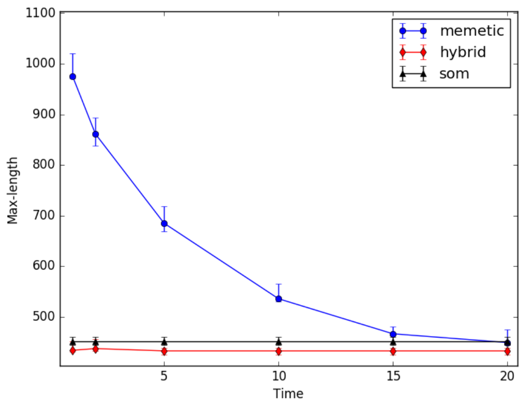


Figure 6.3: Overall view of the compared algorithms on the test with ten instances. The time is displayed in seconds.

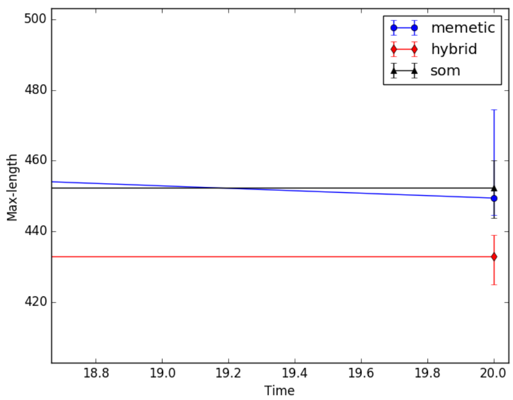


Figure 6.4: Detailed view of the second test. The time is displayed in seconds. This detail shows that after twenty seconds the Memetic algorithm can provide a solution that surpases the solution provided by SOM. However the Hybrid algorithm provides better solution in this particular test.

CHAPTER 7

Conclusion

The thesis focuses on solutions of the Dubins Traveling Salesman Problem with Neighborhoods (DTSPN). The motivation for solving the DTSPN are surveillance missions performed by the Unmanned Aerial Vehicles (UAV). First, the Traveling Salesman Problem (TSP) and the Traveling Salesman Problem with Neighborhoods (TSPN) are presented. These are problems of connecting locations of interest by a path visiting all of the locations minimizing the traveled distance. The model for vehicle constrained by its turning radius, the Dubins vehicle [8] is presented. When searching for a curvature constrained path connecting all locations of interest, the problem is known as the Dubins Touring Problem (DTP) arises. This problem of connecting two points in a plane by a curvature constrained path is presented, and some existing methods are mentioned [2] [11]. After presenting these prerequisites, the thesis presents the Dubins Traveling Salesman Problem (DTSP) and Dubins Traveling Salesman with Neighborhoods (DTSPN) to the reader.

The main contribution of this thesis is a novel Hybrid algorithm for solving the DTSPN. This algorithm is a combination of the Unsupervised learning algorithm based on Self Organizing Maps (SOM) [4] and Memetic algorithm [3]. Both these algorithms are presented to the reader as they are essential for the Hybrid algorithm. The SOM algorithm can provide a competitive solution quickly but is not able to improve this solution with more computational time. While the Memetic algorithm takes longer time to provide a first competitive solution, it can converge to high-quality solutions with more time. The Hybrid algorithm combines the advantages of both mentioned algorithms by initializing the first generation by quality solutions provided by SOM, while also being able to improve these solutions over time.

The results of the proposed Hybrid algorithm are compared to the algorithms used for its creation. It is shown that the algorithm can provide a competitive solution quickly while also being able to further improve this solution.

For future work, The Hybrid algorithm could be improved by further combining the two used approaches. During the run of the evolutionary part of the algorithm, more solutions provided by SOM could be injected into next generations to enhance the diversity and quality of the population, thus improving its ability to generate higher quality offsprings.

Bibliography

[1] `https://newatlas.com/carbonix-volanti-vtol-fixed-wing-industrial-uav/48253`. Accesed: 17.5.2019.

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

[3] B. X. X. Zhang, J. Chen 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.

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

[5] V. C. D. Applegate, R. Bixby and W. Cook, *The Traveling Salesman Problem: A Computational Study*. Princeton University Press, 2007.

[6] S. Gupta and P. Panwar, "Solving travelling salesman problem using genetic algorithm," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 3, pp. 376–380, 01 2013.

[7] S. Saud, H. Kodaz, and İ. Babaoğlu, "Solving travelling salesman problem by using optimization algorithms," *KnE Social Sciences*, vol. 3, no. 1, pp. 17–32, 2018.

[8] 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, 1957.

[9] J. Ny, E. Feron, and E. Frazzoli, "On the dubins traveling salesman problem," *IEEE Transactions on Automatic Control*, vol. 57, pp. 265–270, Jan 2012.

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

[11] K. Savla, E. Frazzoli, and F. Bullo, “On the point-to-point and traveling salesperson problems for dubins’ vehicle,” in *Proceedings of the 2005, American Control Conference, 2005.*, pp. 786–791 vol. 2, June 2005.

[12] R. Pěnička, J. Faigl, P. Váňa, and M. Saska, “Dubins orienteering problem,” *IEEE Robotics and Automation Letters*, vol. 2, pp. 1210–1217, April 2017.

[13] J. Isaacs and J. Hespanha, “Dubins traveling salesman problem with neighborhoods: A graph-based approach,” *Algorithms*, vol. 6, pp. 84–99, 02 2013.

[14] J. Faigl and P. Váňa, “Self-organizing map for the curvature-constrained traveling salesman problem,” in *Artificial Neural Networks and Machine Learning – ICANN 2016* (A. E. Villa, P. Masulli, and A. J. Pons Rivero, eds.), (Cham), pp. 497–505, Springer International Publishing, 2016.

[15] J. Faigl, “Approximate solution of the multiple watchman routes problem with restricted visibility range,” *IEEE Transactions on Neural Networks*, vol. 21, pp. 1668–1679, Oct 2010.

[16] J. Faigl and L. Přeučil, “Self-organizing map for the multi-goal path planning with polygonal goals,” in *Artificial Neural Networks and Machine Learning – ICANN 2011* (T. Honkela, W. Duch, M. Girolami, and S. Kaski, eds.), (Berlin, Heidelberg), pp. 85–92, Springer Berlin Heidelberg, 2011.

[17] J. Faigl and G. A. Hollinger, “Unifying multi-goal path planning for autonomous data collection,” in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2937–2942, Sep. 2014.

[18] J. Faigl, “An application of self-organizing map for multirobot multigoal path planning with minmax objective,” *Computational Intelligence and Neuroscience*, no. 10, 2016.

[19] T. Kohonen, “The self-organizing map (som),” 2001.

[20] B. Angéniol, G. de La Croix Vaubois, and J.-Y. L. Texier, “Self-organizing feature maps and the travelling salesman problem,” *Neural Networks*, vol. 1, no. 4, pp. 289 – 293, 1988.

[21] E. Cochrane and J. Beasley, “The co-adaptive neural network approach to the euclidean travelling salesman problem,” *Neural Networks*, vol. 16, no. 10, pp. 1499 – 1525, 2003.