Robust mission planning for Autonomous Marine Vehicle fleets

Thompson, Fletcher ; Galeazzi, Roberto

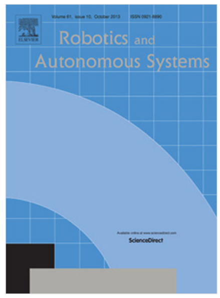
Published in:
Robotics and Autonomous Systems

Link to article, DOI:
10.1016/j.robot.2019.103404

Publication date:
2020

Document Version
Early version, also known as pre-print

Journal Pre-proof



Robust mission planning for Autonomous Marine Vehicle fleets

Fletcher Thompson, Roberto Galeazzi

PII: S0921-8890(19)30335-5
DOI: https://doi.org/10.1016/j.robot.2019.103404
Reference: ROBOT 103404

To appear in: *Robotics and Autonomous Systems*

Received date : 14 May 2019
Revised date : 2 November 2019
Accepted date : 8 December 2019

Please cite this article as: F. Thompson and R. Galeazzi, Robust mission planning for Autonomous Marine Vehicle fleets, *Robotics and Autonomous Systems* (2019), doi: https://doi.org/10.1016/j.robot.2019.103404.

Highlights

1. A complete (end-to-end) framework for Autonomous Marine Vehicle mission planning designed to adhere to vehicle energy capacities and maximise completed mission objectives was defined.
2. The standard marine vehicle dynamics model was developed to forecast energy consumption for possible trajectories in a mission.
3. Existing optimal route selection algorithms were modified to demonstrate improved solution times without compromising on solution quality.
4. Simulation of an offshore wind farm inspection mission shows the mission planner produces unique, collision free routes for each vehicle that conform to energy constraints and maximise the number of inspected turbines.

Robust Mission Planning for Autonomous Marine Vehicle Fleets

Fletcher Thompson[a,*], Roberto Galeazzi[b]

[a] *National Centre for Maritime Engineering and Hydrodynamics, 100 Newnham Dr, Newnham TAS 7248, Australia*

[b] *Danmarks Tekniske Universitet, Building 326, Elektrovej, 2800 Kgs. Lyngby, Denmark*

Abstract

Mission planning for Autonomous Marine Vehicles (AMVs) is non-trivial because significant uncertainty is present when profiling the operating environment, especially for underwater missions. Mission complexity is compounded for each vehicle added to the mission. In practice, fleet operations are formulated as separate temporal problems by the operator and solved using a temporal planner. This paper proposes a planning method that uses energy as the base planning resource instead of time. Unlike temporal planners, energy planners account for physical loads endured by the vehicles. The extent of uncertainty in the vehicle loads is clarified by using the vehicle dynamics model and Monte Carlo simulation on the model parameters. The planning method is a multistage procedure to decompose operator specified task, obstacle, and vehicle data into an energy formulation of the Team Orienteering Problem (TOP) which is then solved using Discrete Strengthened PSO (DStPSO). The DStPSO algorithm has been modified to include a selective swarm size decay method that allows for larger initial swarm sizes to promote early exploration and preserves a percentage of the best performing particles on each iteration to save computational resources. The planner produces near-optimal routes containing feasible trajectories for individual vehicles that maximise tasks completed according to individual vehicle energy constraints. A case-study mission for long-term, large-scale, underwater inspection of a wind turbine array was converted into input data to evaluate the planner. Energy planning presents the opportunity for vehicles to actively monitor the feasibility of their individual plan against their current energy consumption, allowing for advanced reasoning and fault handling to occur *in situ* without operator assistance.

Keywords: Planning AI, Multi-robot Systems, Marine Robotics

1. Introduction

This paper proposes a modular framework for the automated mission planning of a fleet of Autonomous Marine Vehicles (AMVs) for long-term, large-scale missions such as inspection, maintenance, and monitoring of offshore structures. Mission planning for solo AMVs is non-trivial. Factors that need to be considered include the objectives that determine mission success, the requirements of the objectives, the suitability for the chosen vehicle to fulfil these requirements, and the associated risks and consequences that the vehicle and operator crew will be subject to during the mission. The complexities are compounded when multiple vehicles are deployed for different tasks, which increases cognitive load on the operators of the vehicles [1]. By implementing aspects of Artificial Intelligence (AI) in the automation of mission planning such as task prioritisation, feasibility analysis, and path planning, the duties of the operator can be refocused on strategic objectives such as task generation and risk analysis, increasing the robustness of the final mission plan.

The mission planning framework proposed in this paper is the primary novel contribution to multi-AMV planning literature. The framework is designed as a modular pipeline, that can accept methodologies for each module so long as the input and output requirements for the module slot are satisfied (see Fig. 1 for the basic outline of the framework). By using working methods from operations research, path-planning, and optimisation, we demonstrate the functionality and effectiveness of the framework as a mission planner. We have defined ”placeholder” modules to perform the demonstration, but the opportunity exists to improve these modules by replacing the placeholders with better performing algorithms.

The planner formulates the multi-AMV mission plan as the Team Orienteering Problem (TOP) [2] rather than as a resource scheduling problem, with energy as the base planning resource instead of time. Energy resource optimisation is multi-objective in that it represents both the time taken and the loading on the vehicles, and requires dynamic models of the vehicles and the environment to

*Corresponding Author
Email addresses: fletcher.thompson@utas.edu.au (Fletcher Thompson), rg@elektro.dtu.dk (Roberto Galeazzi)

provide a reliable plan. The planner is proposed as an alternative framework to existing temporal logic planners [3] and Hierarchical Task Network (HTN) planners [4, 5] for marine vehicles.

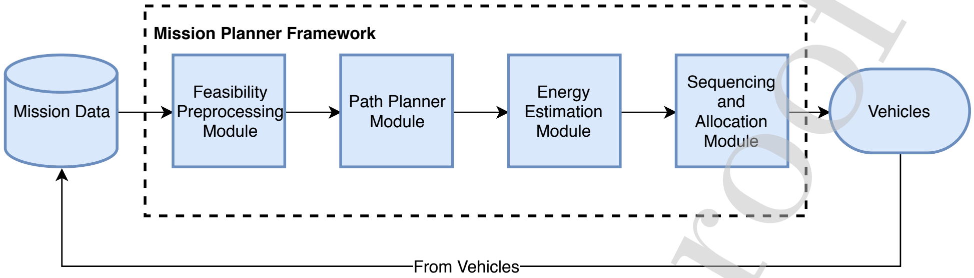


Figure 1: Modules of the proposed mission planner framework.

The implementation of the framework in this paper integrates methodologies that draw from separate fields of research: AI planning and operational research, marine vehicle systems, robot path planning, unsupervised learning, and Particle Swarm Optimisation (PSO). Development of the modules outlined in Fig. 1 resulted in minor contributions and improvements to existing literature in these fields as listed below:

1. A k-means clustering based algorithm that decomposes large-scale mission profiles into feasible operating zones, effectively eliminating large portions of the planner's search space.
2. A swarm size decay algorithm was added to an existing variant of PSO, Discrete Strengthened Particle Swarm Optimisation (DStPSO), to use less computational resources while still achieving comparable quality in the generated plans.
3. The parameters within the marine vehicle dynamics model used for energy estimation were formulated as a stochastic process to accommodate uncertainties from real-world phenomena.

Mission planning for AMVs can be structured into three procedural steps. The first step, referred to as *knowledge-based reasoning*, relies on the planning agent's (human operator or AI) knowledge base to identify tasks relevant to the mission objectives, requisite actions that will complete an identified task, and the sequencing of the identified tasks within a logical hierarchy of dependency succession (e.g. substructure must be cleaned of bio-fouling before inspection tasks can be performed). In AMV literature, HTNs [4, 5] and mixed-initiative planning [3, 6] have had the most success at providing easy methods for the operator to represent abstract tasks as sequences of primitive actions.

The second step, referred to as *task allocation,* is a common problem in multi-robot systems literature [7]. Task allocation is a multi-objective constraint satisfaction problem that considers the feasibility for a vehicle to complete the task (reliability), the vehicle's competence at completing the task to a required standard (quality), and the urgency of the task and the speed at which a vehicle can complete it (utility). To assess the suitability of a vehicle for a task, the task must first be refined down to a sufficient level of detail so that technical assessments can be made. In the case of the proposed planner, tasks are refined into velocity/loading profiles and then assessed for energy costs against the vehicle's energy capacity.

The third step is *risk projection* and requires the planning agent to consider uncertainties associated with allocated resources and mission structure within the context of failure analysis. The amount and magnitude of uncertainties are highly dependent upon the mission, the environment, and the vehicles involved and is difficult to generalise. If the mission proposal generated from steps one and two is considered acceptable in terms of risk and consequence, then the mission can be executed. Planners that can make projections on uncertainty provide the operator with decision support.

Deliberative planners exist in the marine robotics literature for both solo and fleet based operations. The EUROPtus mission planner [3] was developed to assist operators with the scheduling and allocation of oceanographic sampling tasks to a variety of AMVs operating in meso-scale areas (50 km^2). EUROPtus is a general purpose deliberative planner that uses temporal logic to allocate operator specified tasks to vehicles based on availability, producing functional plans that factor in task length uncertainty. Individual vehicles could then use their onboard T-REX planner [8] to repair plans *in situ*. This decentralised configuration is well suited for large sampling missions where vehicles do not have to directly cooperate to achieve mission objectives. The planner has not been trialled on missions that contain tasks with interdependencies. Additionally, the time domain is used to obtain mission plans but the planner itself does not consider the loadings the vehicle must overcome to complete the plan.

Following a HTN approach, [9] produced a task allocation method for mine countermeasure missions that distributes the mission plan onboard each vehicle. New tasks that are generated by vehicles *in situ*, such as mine clearance tasks that result from positive mine detection tasks, are intermittently broadcasted using underwater acoustic communication, which will propagate to neighbouring vehicles. Because each vehicle is aware of the others, allocation is done locally by assessing the vehicle's proximity and ability to perform a new task against the others. This leads to inefficiencies in mission execution, where multiple vehicles may allocate themselves the same task based on out-of-date information. Mission planning tools [10] were developed to parallelise a task into smaller equivalent sub-tasks that are subsequently allocated to individual AMVs. The genetic algorithm was used to allocate sub-areas to AMVs participating in a cooperative survey mission. The planning tool is limited by a uniform speed assumption and does not consider loadings on the vehicle that may reduce run times.

This paper specifies a modular pipeline that formulates operator-specified tasks into the TOP [2] and then solves the TOP. The TOP, which can be described as a combination of the vehicle routing problem and the binary knapsack problem, is formulated into the marine vehicle mission planning domain in Sections 2.2 to 2.3 and is represented as a directed edge graph where the nodes represent locations of tasks that include a reward for being visited and the edges represent the cost of transitioning between two nodes. The objective of the TOP is to specify routes for multiple team members that maximise the combined reward (defined in Section 2.4) and meet the individual cost constraints of the members.

The preprocessing module, presented in Sections 2.6 to 2.7 functions as a preprocessing step that clusters operator-specified task data and vehicle data into feasible operating zones for the vehicles based on their estimated point-of-safe-return (PSR), the furthest distance the vehicle can travel safely from a nominal rendezvous point. The PSR is determined through the energy capacity of the vehicle's battery and the expected energy consumption (calculated in Section 2.5) for the vehicle operating at a steady forward speed.

The path-planning module (Section 2.8 is then applied on each cluster. The module generates collision-free transitions between nodes through an Artificial Potential Field (APF) method established by [11] and modified by [12]. A useful property of the vector field obtained by this particular APF is C^{∞} smoothness [12], meaning that the commanded velocity, acceleration, jerk or snap profiles can be directly determined according to the specifications of the vehicle controllers.

The cost of task transitions in the TOP are specified in terms of energy consumption, which is calculated through the energy prediction module presented in Section 2.5. The module obtains the energy consumption distribution of the vehicle by computing the required thruster output loads to follow the commanded velocity profile for the duration of the transition. Difficult to measure forces such as surface friction due to bio-fouling, thruster output variation due to non-linear voltage drop from battery discharge, wave induced pressure fluctuations, and localised currents can be accounted for by adding noise to and randomly sampling the parameters in the dynamics model. These uncertainties are what we aim to account for in our proposed method for energy estimation, increasing the robustness of the planner.

For this paper, the underwater robot platform called REconfigurable MOdular Robotics for Aquatic environments (REMORA) [13] is considered. The mathematical model of the REMORA vehicle's dynamics in calm water [13] was sampled and used in Monte Carlo simulation to produce the energy distribution. The path planner was tested for suitability with the control model of the REMORA AMV developed in [13] and was shown to quickly stabilise and track the generated path.

With the data formulated into the TOP, a PSO variant formulated for discrete operations was chosen as the solver for the sequencing and allocation module (Section 2.9). DStPSO [14] formulates PSO for the discrete domain and uses a reduced version of variable neighbourhood search as a local improvement to the global leader particle, strengthening the best solution. DStPSO was selected because it has few parameters, is simple to implement, and converges to near optimal global solutions in comparatively faster times than most other swarm optimisation methods [15]. The original DStPSO method has been modified to include a linearly adaptive inertia weight based on the stall counter stopping criterion, and a swarm size decay algorithm (Section 2.9.1) that prunes the swarm over each outermost loop of the worst performing particles.

The individual placeholder modules were given unit test instances to benchmark their effectiveness. The hydrodynamic potential flow path-planner was tested on the dynamic model of the REMORA vehicle and evaluated for reliability in collision avoidance. The preprocessing module's clustering algorithm were evaluated on 387 test datasets from [2, 16] for computational cost. The sequencing and allocation module was also evaluated on the same 387 test datasets without the preprocessing module for computational cost, and sensitivity and consistency of the solution's score to parameters such as problem size, swarm size, and decay rate. To evaluate the mission planner framework with the developed modules in its entirety, a case-study inspection mission of the Anholt wind turbine array using a small fleet of REMORA AMVs was used as input data.

The results show the framework is capable of producing near-optimum mission plans over a variety of problems, and can also be used as a benchmark for modules proposed as improvements to the placeholder modules.

2. Methodology

2.1. Definition of Robustness

We define a robust mission planner as a system capable of generating plans across a variety of missions that are:

1. Efficient in terms of energy consumption.
2. Unlikely to strand any of the vehicles due to energy depletion.
3. Reducing the likelihood of collisions.
4. Maximising the productivity of all the vehicles.

The rest of this section develops a mission planning framework based on these design criteria. The formulation of the TOP (Sections 2.2 to 2.3) addresses criteria 1, 2 and 4 in conjunction with the energy estimation module (Section 2.5) and feasibility preprocessing module (Sections 2.6

to 2.7). The inclusion of the path planning module (Section 2.8) contributes to criteria 3, as well as indirectly contributing to the other criteria by improving the realism of the energy estimate. Finally, we demonstrate that the framework and proposed placeholders can generate feasible, near-optimum results across a large range of mission problems in Section 3.

2.2. Problem Formulation

AMV fleet missions can be described abstractly as allocating and sequencing tasks to be completed at each target to the available vehicles. An applied mathematics problem that has a similar objective to the above is the TOP defined in [2]. In this problem, several agents must be allocated separate routes through a set of targets that represent tasks that yield a reward variable when completed. These routes must satisfy the energy constraints of the vehicles and maximise the collective reward of the team. An optimal solution for the TOP has the following characteristics:

1. Each vehicle has a unique set of tasks.
2. Each vehicle's route does not cross over itself.
3. Each vehicle's route has a predicted energy consumption that is close to the energy capacity of the vehicle.
4. Each vehicle starts and finishes at the nominated starting and finishing points.

2.3. Definition of the TOP Adapted for AMV Missions

The TOP stands as a solid method for allocating sequences of points to team members where the situation may arise that not all points can be visited. Variants of the TOP that consider time-windows [17], stochastic weights (for the single vehicle TOP) [18], time dependent weights [19], and many others (see [20] for more variants) introduce aspects of real-world problems to the TOP. There are aspects of each of these variants that also suit AMV mission planning, but to begin with we present the following definitions that adapt the TOP to fit within the multi-robot systems and marine vehicle domains.

Definition 1. A task, T, is the tuple (g, s, I_t) where

- $g \in \mathbb{R}^3$ is the vector containing the location information of the task.

- $s \in \mathbb{R} \geq 0$ is the scalar reward yielded by completing the task.
- I_t is a tuple containing further information on the type of task, prerequisite tasks, and effects on other tasks.

Assumption 1. *For non-hierarchical missions (i.e. tasks are independent from each other), I_t simply points to the type of task, no prerequisite tasks or effects on other tasks need to be considered.*

Additionally, s is mapped to I_t by a time dependent reward function specified by the operator which must also depend upon the importance, urgency, and frequency of the task. We define such a reward function in Section 2.4.

Definition 2. A vehicle, V, is the tuple (e_b, I_v) where

- $e_b \in \mathbb{R}$ is the energy storage capacity of the vehicle's batteries in Joules.
- I_v is a tuple containing further information on the vehicle identifier, type of vehicle, domain of operation, collision boundary, capabilities, and dynamic model.

Assumption 2. *For a homogeneous fleet (i.e. vehicles are of similar type and capability), I_v provides unique identifiers and the type, domain, capabilities and dynamic model are identical for all vehicles.*

Definition 3. An obstacle, O, is the tuple (X_o, r_o, I_o) where

- $X_o \in \mathbb{R}^3$ is the 3D position of the obstacle centroid.
- $r_o \in \mathbb{R}$ is the clearance radius the operator would like to maintain around the obstacle.
- I_o is a tuple containing further information on the obstacle, such as the classification (e.g. buoy, pile, rock, etc.) and the coordinate convention used by X_o.

Definition 4. The open mission, $\mathcal{M}_O$, is the sextuple $(\mathcal{T}, \mathcal{V}, \mathcal{O}, P, Q, E)$ where

- The operator defines N_T number of T which are collected in the N_T-tuple $\mathcal{T}$.
- The operator defines N_V number of V and collects them in the N_V-tuple $\mathcal{V}$.
- The operator defines N_O number of O and collects them in the N_O-tuple: $\mathcal{O}$.

- $P \in \mathbb{N}^{N_T}$ is the set of N_T sequential integers that references an element of $\mathcal{T}$: $P = \{1, \ldots, N_T\}$.
- $Q \in \mathbb{N}_V^N$ is the set of N_V sequential integers that references an element of $\mathcal{V}$: $Q = \{1, \ldots, N_V\}$.
- E is the zero-diagonal matrix of costs for transitioning between $\mathcal{T}_{P_i}$ and $\mathcal{T}_{P_j}$ and performing task $\mathcal{T}_{P_j}$: $E \in \mathbb{R}^{N_T \times N_T} \geq 0$.

$\mathcal{M}_O$ is the search domain of the planner. E is zero-diagonal because the transition $P_i = P_j$ is a forbidden transition. There are a total of $N_T^2 - N_T$ non-zero entries in E. The planner must provide a subset of $\mathcal{T}$ allocated to $\mathcal{V}$ as a proposal that can be evaluated for adherence to the energy constraints of the vehicles and the total reward yielded from completed tasks. The planner's proposal is specified as follows.

Definition 5. The closed mission, $\mathcal{M}_C$, is the quintuple $(\mathcal{T}, \mathcal{V}, R, S, F)$ where

- R is the N_V length set of tuples, where each tuple, R_Q, has an independent length $L_Q \geq 2$. R_Q is an ordered sequence subset of P corresponding to each vehicle's proposed route through $\mathcal{T}$.
- S is the set of rewards collected from completed tasks in $\mathcal{T}$: $S = \left\{s \in \mathcal{T}_{R_Q}\right\}$.
- F is the N_V length set of tuples, each of length $L_Q - 1$, corresponding to the ordered sequence of elements of E accessed by the ordered sequential pairs in R_Q. $F_Q = \{E(R_{Q_i}, R_{Q_j}) \mid 1 \leq i \leq L_Q - 1,\ 2 \leq j \leq L_Q,\ (i, j) \in \mathbb{N}\}$.

The mission planner is then a solver that finds the most effective $\mathcal{M}_C$ according to the following fitness function and constraint:

$$\begin{aligned} \underset{\mathcal{M}_C}{\text{maximise}} \quad & \sum_{x_i \in S} x_i \\ \text{subject to} \quad & \sum_{y_i \in F_Q} y_i \leq e_b \in \mathcal{V}_Q \end{aligned} \tag{1}$$

To introduce real-world components to this framework, E is the sum of the energy consumed traversing from $g \in \mathcal{T}_{P_i}$ to $g \in \mathcal{T}_{P_j}$ along the collision free path $\mathcal{S}_{ij}$, labelled $E_{s,ij}$, the energy consumed completing the task at $\mathcal{T}_{P_j}$, labelled $E_{t,j}$ and the hotel load drain on the vehicle over the time taken to traverse the path and complete the task, labelled E_h. E is provided as the expectation of a stochastic process, which presents the opportunity of planning using stochastic weights. The models underlying the estimation of the energy variable are presented in Section 2.5.

Because the vehicles must navigate around obstacles and take routes that are energy efficient, Euclidean distance calculations for $g \in \mathcal{T}_P$ may result in solutions that underestimate the actual distance travelled by the vehicle, which will subsequently underestimate the energy cost of traversing the path. Therefore, care must be taken with finding $\mathcal{S}_{ij}$. A simple method for projecting a smooth, collision free trajectory for an AMV is presented in Section 2.8.

Similar to the TOP formulation, each vehicle must start and finish at two specified locations by the operator, which are inserted at the beginning and end of $\mathcal{T}$ as two special tasks, $\mathcal{T}_1$ and $\mathcal{T}_{N_T}$ respectively. For the *minimum operator effort* mission, we would like the vehicles to return to their deployment position, $\mathcal{T}_1 = \mathcal{T}_{N_T}$ is the special case called the *home point*. This is because in practice, AMVs are deployed from a central location such as a shore launch point, moored docking station or a vessel. The *home point* conveniently ensures the vehicles return to a position where they are able to recharge, offload collected data and diagnostic information, and be easily accessible for maintenance. In Section 3.2.1, we describe a procedure to determine ideal location of $g \in \mathcal{T}_{\{1,N_T\}}$.

Solving the TOP is well studied and many solutions have been developed, most of which are available in [20]. The Discrete Strengthened Particle Swarm Optimisation (DStPSO) method [21] was selected as a meta-heuristic method for solving the TOP. Compared to other meta-heuristic solvers such as Ant Colony Optimisation, Genetic Algorithm, and Tabu search, DStPSO reaches near-optimum solutions faster with fewer parameters [15].

To summarise, the following components from $\mathcal{M}_O$ are required in order to be solved in a similar fashion to the TOP:

1. The number of tasks, N_T
2. The number of vehicles, N_V
3. Each vehicle's battery energy storage constraint, e_b.

4. The value of each task's reward, s.
5. The cost of moving to and performing a task, E_{ij}.

The TOP requires the mission data to be processed into the above form to find an optimum $\mathcal{M}_C$, which is distributed to the vehicles upon deployment. The proposed planner follows the process in Fig. 2. The following sub-sections detail the steps taken to obtain each of the components of the process.

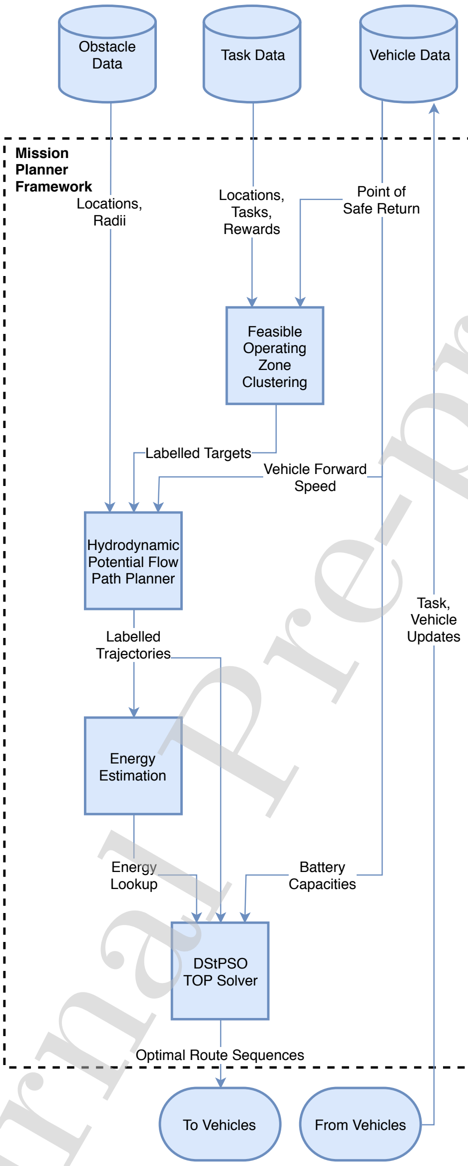


Figure 2: Process flow of the mission planner framework and the proposed placeholder modules. All modules from Fig. 1 are shown along with the expected incoming and outgoing data.

2.4. Modelling Rewards

For the TOP to be solved, rewards for completing a task must be assigned to each target. In the case of recurring tasks such as maintenance and cleaning, reward is primarily a function of time since the task was last completed. Machine learning methods such as linear regression could be used to estimate the reward of a task from a sensor based data set (such as measuring the vibration of a structure). As a simple, parameterised alternative, the sigmoid function, a popular continuous activation function in machine learning, is selected as the candidate function for representing the reward of a task:

$$s(t) = \frac{1}{1 + e^{-t}} \tag{2}$$

The range of the sigmoid function is $[0, 1]$, hence it is useful as a binary activation switch. However, it is desired that tasks can be parameterised in terms of importance, importance growth-rate, and frequency. The generalised logistic function [22] could allow the operator increased flexibility with how reward $s \in T$ grows or decays with time. Given the generalised logistic function:

$$s(t) = A + \frac{K - A}{(C + De^{Bt})^{1/\delta}} \mid (A, B, C, D, K, \delta) \in \mathbb{R} \tag{3}$$

The operator can control the start and end values with A and K, and rate of growth/decay B of $s \in T$ (see Fig. 3). As time progresses and T has not been completed, s can grow or decay. The independent variable, t, can be set to 0 upon completion of a task to restart the reward function.

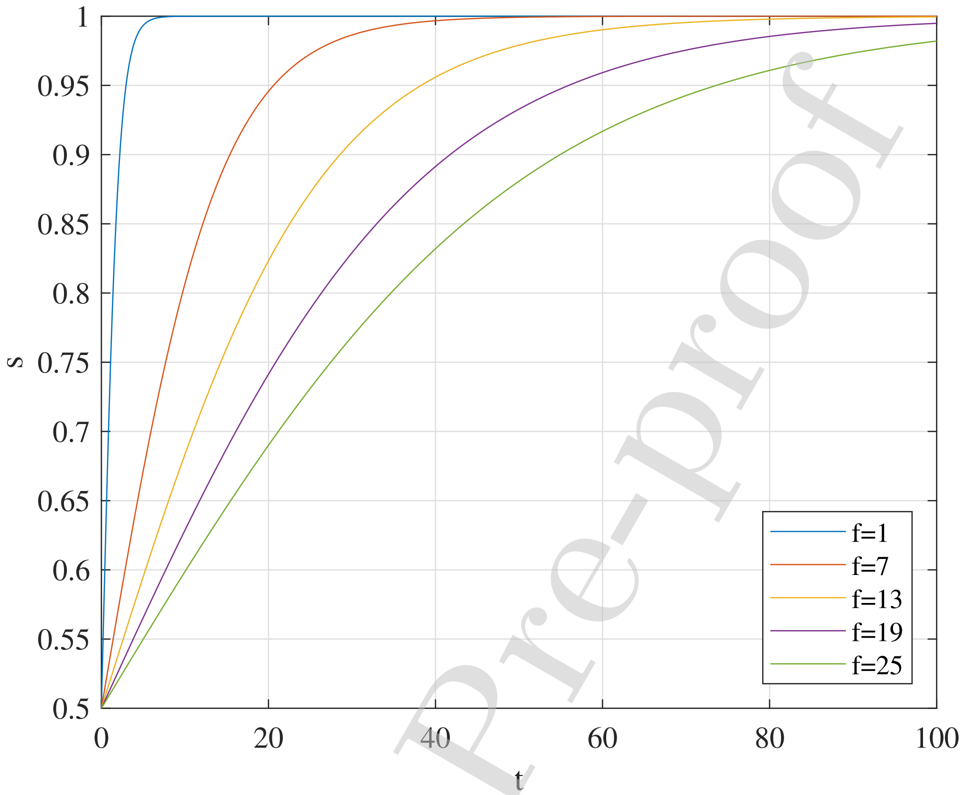


Figure 3: Generalised Sigmoid Reward Over Arbitrary Timescale, $A = 0$, $B = \frac{1}{f}$,$C = 1$, $D = 1$, $K = 1$ and $\delta = 1$

2.5. Energy Consumption Prediction

Let $\mathcal{S}_{ij}$ be the collision free path connecting the task location $g \in \mathcal{T}_{P_i}$ to $g \in \mathcal{T}_{P_j}$. The energy consumed by the vehicle $V \in \mathcal{V}$ to traverse the route and perform the tasks is given by

$$E_{ij} = E_{s,ij} + E_{t,j} + E_h, \quad i \neq j \tag{4}$$

where $E_{s,ij}$ is the energy spent to traverse the path, $E_{t,j}$ is the energy depleted to perform the task T at $g \in \mathcal{T}_{P_j}$ and E_h is the energy consumed by the hotel load. The complete set of calculations for $E_{s,ij}$ (A.1, Eqs. (19) to (23)), $E_{t,j}$ (A.2) and E_h (A.3, Eq. (24)) are located in Appendix A for completeness.

Eq. (23) shows that the energy $E_{s,ij}$ is a function of the hydrostatic and hydrodynamic characteristics of the specific vehicle V through Eq. (21). The coefficients of $\mathbf{D}(\boldsymbol{\nu})$, $\mathbf{C}(\boldsymbol{\nu})$ and $\mathbf{g}(\boldsymbol{\eta})$ are usually estimated from model tests and hence affected by uncertainty [13]. Furthermore, as the vehicle performs missions natural wear and tear will affect its hydrodynamic characteristics resulting in changes of the coefficients. This implies that a pure deterministic description of $\boldsymbol{\tau}$ may cause severe underestimates of the energy consumption associated with traversing the route R_Q. To account for model uncertainties and wear and tear the steady state generalised vector $\bar{\boldsymbol{\tau}}$ is modelled as a Gaussian random vector with mean $\boldsymbol{\mu}_\tau$ and covariance matrix $\boldsymbol{\Sigma}_\tau$, i.e. $\bar{\boldsymbol{\tau}} \sim \mathcal{N}(\boldsymbol{\mu}_\tau, \boldsymbol{\Sigma}_\tau)$. Therefore both the power spent and the energy needed to traverse the path become random variables and their estimates are computed through the expected value operator $E[\cdot]$ as

$$\hat{E}_{s,ij} = E\left[E_{s,ij}\right] = E\left[\int_{\mathcal{S}_{ij}} \bar{\boldsymbol{\tau}}(\mathbf{s})\, \mathrm{d}\mathbf{s}\right] = \int_{\mathcal{S}_{i}ij} \mathbf{T}_c E\left[\bar{\mathbf{t}}(\mathbf{s})\right]\, \mathrm{d}\mathbf{s} = \int_{t_s}^{t_f} E\left[\mathcal{P}(t)\right]\, \mathrm{d}t. \tag{5}$$

Remark. Since the matrices $\mathbf{D}(\boldsymbol{\nu})$, $\mathbf{C}(\boldsymbol{\nu})$ and the vector $\mathbf{g}(\boldsymbol{\eta})$ are linear in the parameters and each parameter is estimated as being normally distributed, then the generalised vector of forces and moments is normally distributed. However as the vehicle ages through operations the wear and tear may determine changes in the parameters such that distributions other than normal will be better suited. This implies that model parameters should be periodically re-estimated in order to reduce errors in the energy consumption estimation.

2.6. Vehicle Range

The range of a vehicle depends on its total energy storage, hotel load, power distribution efficiency, mechanical efficiency, propulsive efficiency, hydrodynamic drag properties, and environmental loadings. Estimating all of these properties, which in reality vary with time, is non-trivial. However, an ideal range for a specified forward speed can be obtained based on approximated constants as described in [23]. For most vehicles, the range is obtained through endurance testing under certain speeds and weather conditions [24]. The range of the vehicle can be inferred for new conditions based on this knowledge.

In Section 2.5 we proposed a stochastic approach as an alternative to the above methods to obtain an energy consumption distribution for the vehicle over a velocity profile. We can use the steady state power consumption evaluated for the vehicle travelling at a constant forward speed to estimate the forward distance travelled before the vehicle's energy storage, e_b, is depleted. The battery's capacity (C_b, measured in Ah) and its nominal operating voltage, V_b, are related to its energy storage by:

$$e_b = C_b \times 3600(s/h) \times V_b \tag{6}$$

The point-of-safe-return, PSR, is useful for robust mission planning because it puts an upper bound on the distance the vehicle is allowed to be from its home point. The PSR for the vehicle operating at a constant forward speed of $U = \bar{U}$ m/s is:

$$PSR = \frac{\bar{U} \times E_b}{2(\mathcal{P} + \mathcal{H})} \tag{7}$$

The difficulty in this procedure is that the energy consumption is non-trivial to predict in the marine environment. Dynamic loads from waves, wind, tide, current, thermal currents, and hydrodynamic forces all influence the effort generated by the vehicle's thrusters. Hydrodynamic effects such as turbulence, influenced by bio-fouling and surface degradation, and the design of the vehicles thrusters also affect the efficiency of the vehicle, dependent upon the vehicle's speed and thruster RPM. Small-scale hydrodynamic effects can be captured through the use of parameter variation in the dynamic model. However, large-scale effects such as current, waves, and wind must be added as separate estimator components to the base dynamic model.

2.7. Target Clustering

The selection of points available for the vehicle team to visit must all be within the PSR of the vehicle with the largest range. For target sets that are distributed over large areas, such as offshore wind farm installations or macro-scale marine sampling, some of the targets will always be outside of any of the vehicles' reach for any time instance. Problems that contain many targets will have larger search spaces and will take longer to solve. Removing the infeasible targets will simplify the search domain. We propose grouping target sets into clusters that are sized appropriately so that they are within serviceable range of the vehicles from the centroid of the cluster. The k-means clustering algorithm is a suitable method for obtaining appropriately sized clusters of targets. Alg. 1 details a simple procedure that achieves feasible operating zones via target clustering.

Algorithm 1: Feasible operating zone clustering

```
input  : Target coordinates L ∈ ℝ^{N×3}; Point of Safe Return PSR
output: C sets of indexes, X_b est centroids of each cluster

1  flag = 0;
2  N_C = 0;
3  while ¬flag do
4      N_C ← N_C + 1;
5      id ←zeros(N_C, 1);
6      sumD_best ← 0;
7      for i ← 1 to reps do
8          [IDX, X, sumD, sqdD] ←kmeans(L, N_C);
9          if 1/sumD > sumD_best then
10             sumD_best ← 1/sumD;
11             IDX_best ← IDX;
12             X_best ← X;
13             sqdD_best ← sqdD;
14     for i ← 1 to N_C do
15         id(i) ← PSR^2 > max(sqdD_best(IDX_best == i));
16     flag ←all(id);
17 for i ← 1 to N_C do
18     C(i) ← IDX_best(IDX_best == i);
19 return C,X_best;
```

The set of locations for all non-special tasks, $\{g \in \mathcal{T}_i \mid 2 \leq i \in \mathbb{N} \leq N_T - 1\}$, and the largest calculated PSR of the vehicles are used as inputs to the algorithm. Lines 7-13 replicate the k-means clustering function on the location data $reps$ times, the solution with the best fit (i.e. the lowest *sumD*) is chosen. Lines 14-16 checks that the point furthest from the center in each cluster is less than the specified PSR. If this constraint is not met, then the number of required clusters (N_C) is increased and the process begins again. The returned variable, C, is the tuple of length N_C where each element corresponds to a unique subset of P. C is used to subdivide $\mathcal{M}_O$ into N_C sub-missions, labelled as $\mathcal{M}_O^{(k)}$ where $1 \leq k \in \mathbb{N} \leq N_C$. The special *home point* tasks $\mathcal{T}_{\{1,N_T\}} \in \mathcal{M}_O^{(k)}$ have their location set to $X_{best}^{(k)}$, the centroid of the corresponding k-th cluster in $\mathbb{R}^3$.

2.8. Obtaining Paths

Assuming straight-path distances between targets will lead to underestimation in energy costs when planning routes for vehicles in environments containing obstacles. The vehicle will occasionally take non-straight paths, either as a result of navigating around obstacles or because the dynamics of the vehicle prevent it from instantaneously adjusting to the reference trajectory, and a planner that does not account for this may produce optimistic plans that are unattainable by the vehicle. The planner must have a realistic estimate of the distance of a collision free and dynamically viable path, which is a well studied problem in robot path planning literature [25].

Generating a valid path for a vehicle to transition from one point to another requires consideration of the obstacles between the vehicle's starting and finish points for a transition. For a basic static obstacle avoidance method, the following components are required:

1. Vehicle's starting location and destination, $\{g \in \mathcal{T}_{\{i,j\}}\}$.
2. Vehicle's collision radius, defined in $I_v \in V$.
3. Static obstacle locations and sizes, $\mathcal{O}$.

There are many successful methods available in path planning literature: Probabilistic Road Maps [26], Rapidly exploring Random Tree [27], A* [28], any-angle (Θ*) [29], and Artificial Potential Fields (APF) [30]. APF methods that use hydrodynamic potential flow theory ([11, 31]) can produce smooth, spline-like trajectories efficiently because the search domain is defined in part by analytic equations. A particle pursuit guidance controller was developed specifically for marine vehicles [31] that used the stream function of a hydrodynamic APF to guide a vessel around circular obstacles, but could not guarantee that the particle would not cross an obstacle boundary.

Circular obstacles were modelled as a potential field using the circle theorem [32] that guarantees zero boundary crossflow between inside and outside of the circle. The circle theorem was used for APF path planning for UAVs in [11], where it was also proven that the presence of local minima was guaranteed to be either at the destination sink or exactly on the boundary of the circular obstacle and nowhere else. From the definition of O in Section 2.3, the circle theorem APF method suitably fits as a base path planning model within the AMV mission planner framework. We have adapted this method to generate collision-free routes for marine vehicles.

If the position and velocity of the vehicle is represented in the complex domain $\mathbb{C}$ respectively by

$$z = x + iy \tag{8}$$

$$\frac{dz}{dt} = u + iv \tag{9}$$

where $\{x, y, u, v\} \in \mathbb{R}^4$ are referenced to the planar world frame. The Partial Complex Velocity (PCV) flow field, as derived from the circle theorem used in [11] & [12], is:

$$\frac{dz}{dt} = \frac{Q_s}{2\pi(z - c)} + \frac{Q_s}{2\pi} \frac{r^2}{(b - z)(r^2 + (b - z)(\bar{c} - \bar{b}))}, \tag{10}$$

where Q_s is the strength of the source ($Q_s > 0$) or the sink ($Q_s < 0$), $c \in \mathbb{C}$ is the location of the source/sink (the starting or finishing point), the radius of the obstacle $r \in O$ and $b \in \mathbb{C}$ is the complex variable of $X_o \in O$ in the X-Y plane. The full Complex Velocity (CV) field for an obstacle is the sum of the sink and source PCV fields. For multiple obstacles, simply summing the CV fields will not produce a valid field that represents all of the obstacles. As discussed in [31], the cross flow at the boundary of each obstacle is influenced by the CV flows of all other obstacles. In [12, 11], these influences are eliminated at each obstacle boundary by introducing a weighting term for each obstacle's CV:

$$\alpha_i = \prod_{j \neq i} \frac{d_j^4}{d_i^4 + d_j^4}, \tag{11}$$

where d_i and d_j are the Euclidean distances between the vehicle's current position z and the i-th and j-th obstacle centroids. The complete CV flow for N_O obstacles is then:

$$\text{CV} = u + iv = \sum_{i=1}^{N_O} \alpha_i (\text{PCV}_i^{source} + \text{PCV}_i^{sink}) \tag{12}$$

In effect α_i interpolates the CVs of each obstacle with a weighting from 0 to 1, ensuring that the obstacle closest to the vehicle will have an increasingly dominant flow compared to the other obstacles.

Eq. (12) represents the first order differential equation that can be integrated to obtain the path of the vehicle from a given initial condition. The Dormand-Prince (RKDP) method was selected to evaluate Eq. (12) given a set of obstacles, obstacle radii, and the vehicle's initial and final positions. Compared to the Euler method used by [12], RKDP can solve long trajectories (>1000s) extremely quickly by adapting the step size to minimise calculations whilst retaining an acceptable error tolerance from the real solution.

The method in [12] was developed for non-holonomic vehicles by offsetting the location of the source behind the vehicle position. For holonomic vehicles (i.e. vehicles that can turn on the spot such as ROVs and hovering AUVs), several orientations can be searched through a given starting position using a fitness function to evaluate each solution for shortest travel time, vehicle dynamics, safety, and efficiency. We have used a simple fitness function to determine the shortest path:

$$Z = \frac{1}{(t_f - t_0)} \tag{13}$$

The highest scoring solution will have the shortest path. This ensures that the least energy consuming path is taken given the assumption that the environment is ideal (i.e. no significant changes in wind, wave, or current profiles) and that the vehicle can accurately follow the underlying velocity profile. In practice the shortest path is typically the starting orientation $\psi_0 = \text{atan2}(y_{sink} - y_0, x_{sink} - x_0)$, but if there are many obstacles along this path, other orientations may yield shorter routes. The REMORA's holonomic underwater vehicle model was tested in simulation for following a path generated by integration of Eq. (12) and is presented in Section 2.8.

2.9. Proposal Generation

Our implementation of DStPSO (pictured in Fig. 4) follows the same principles of PSO but has been adapted to work in the discrete domain, strengthened with a local search heuristic on the pioneering particles, and a swarm decay heuristic to save on computational resources. As described in Section 2.3, the search space for the DStPSO algorithm is restricted to $\mathcal{M}_O$. We define a particle by its position $R \in \mathcal{M}_C$ and velocity W. W is the set of points in P that are not in any element of R: $W = P \setminus R$.

From Eq. (1), the position of a particle is subject to the energy constraints of the vehicles. By obtaining F, the feasibility of a route can be determined by checking:

$$\sum_{y_i \in F_i} y_i \leq e_b \in \mathcal{V}_i \forall i \in Q \tag{14}$$

At its core, DStPSO updates its position by inserting random elements from W into elements of R using various insertion method heuristics, constrained by the above energy relation.

A particle is initialised by setting each element of R to $\{1, N_T\}$, corresponding to the special *home point* tasks $\mathcal{T}_1$ and $\mathcal{T}_{N_T}$. The velocity is then $W = P \setminus R = P \setminus \{1, N_T\}$. Each element of R is then sequentially modified by iteratively selecting a random element from W, inserting it using the *cheapest insertion heuristic* [33], and keeping the solution if the updated F still meets the energy constraint. W has the selected element removed and the process repeats until all elements of W have been tried.

The swarm, Q is the set of N_Q initialised particles. Each particle in Q is evaluated for fitness by finding the total collected reward for its current position:

$$\sum_{x_i \in S} x_i \forall i \in R_Q \tag{15}$$

The N_Q long set of particle positions, $pbest$, is initialised by setting each element of $pbest$ equal to the position of the corresponding particle in Q. $pbest$ keeps a running record of the highest scoring position that each particle has visited. $pbest_i$ is only updated when Q_i moves to a position with a fitness higher than the corresponding score of $pbest$. Finally, the particle that has the highest fitness out of $pbest$ is assigned to $gbest$. $gbest$ is only updated if the fittest particle in the updated $pbest$ is higher than the fitness of the current $gbest$.

When $gbest$ is updated, a local search is triggered on $gbest$ using a simplified version of Variable Neighbourhood Search (VNS) [34] called Reduced VNS (RVNS) [14]. RVNS implements three heuristic search methods (or neighbourhoods) on *gbest*: *1. insert for increasing profit*, *2. insert for decreasing cost*, and *3. path inversion* (also known as 2-opt [35]). Each neighbourhood is evaluated for feasibility and improvement, and if the new position meets both criteria then the neighbourhood is set back to neighbourhood 1. If neighbourhood 3 fails to improve the solution several consecutive times, RVNS returns the updated *gbest* and the particle that had pioneered *gbest* is reinitialised to encourage exploration. For further details on RVNS, see [14].

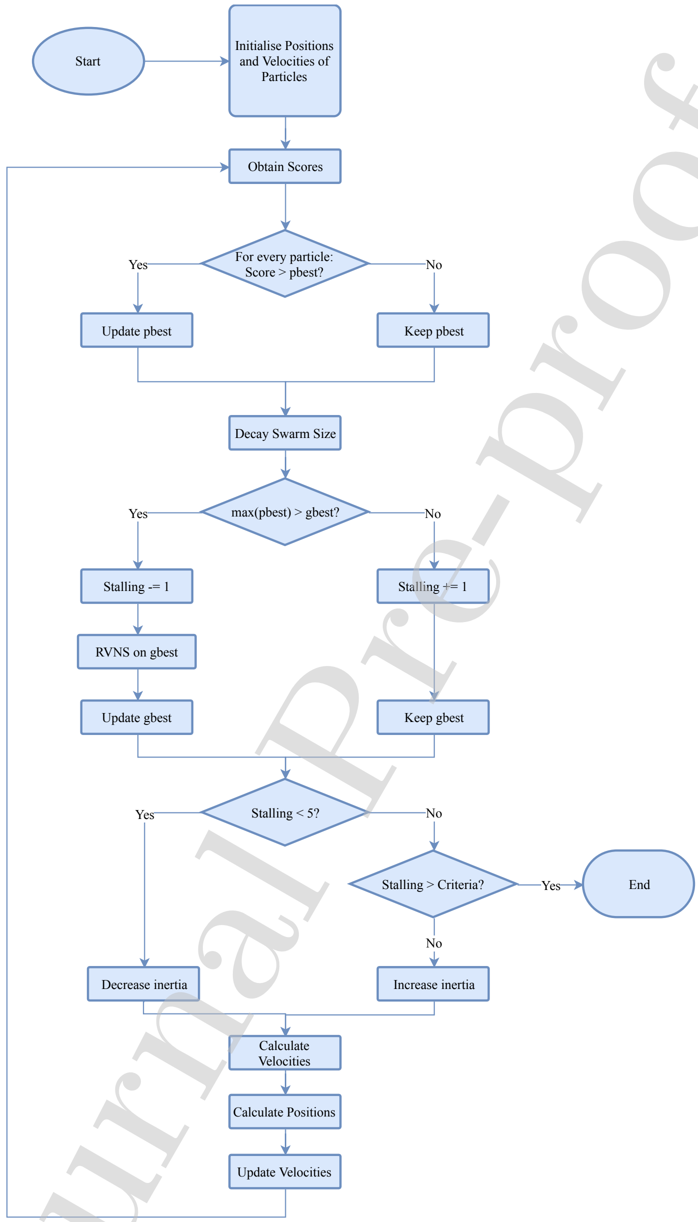


Figure 4: DStPSO procedure with swarm decay.

After the *gbest* local search, or if no *gbest* update occurs, the velocity of the i-th particle in Q is updated in a manner similar to the original PSO,

$$W_i = [w \otimes W_i] \oplus [(c_1 \otimes (pbest_i \ominus R_i)) \oplus (c_2 \otimes (gbest \ominus R_i))], \tag{16}$$

where w represents the typical inertia term used in PSO, and c_1 and c_2 are weighting terms that balance exploration between the particle's best experience and the swarm's best experience. The position and the velocity are subsequently updated by

$$R_i = R_i \circ W_i \tag{17}$$

$$W_i = P \ominus R_i, \tag{18}$$

where each of the special operators, $\otimes$, $\oplus$, $\ominus$, and $\circ$ are defined as follows:

- $\otimes$ Each element of the right hand side (RHS) of the operator is given a random number from 0 to 1. The left hand side (LHS) is a scalar number between 0 and 1. The output is the subset of the RHS that is less than the LHS.
- $\oplus$ Combines two velocity sets. If the LHS contains the *pbest* term and the RHS contains the *gbest* term, then the output is the reordered set $\{RHS \cap LHS, RHS \setminus LHS, LHS \setminus RHS\}$. Otherwise the output is the reordered set $\{RHS, LHS\}$.
- $\ominus$ Is the set difference $LHS \setminus RHS$.
- $\circ$ Apply *insert for increasing profit* from RVNS neighbourhood 1 on the RHS velocity set to the position set on the LHS.

The DStPSO terminates when no successor to the current *gbest* is found for a consecutive amount of iterations. The proposed set of routes for each vehicle, $R \in \mathcal{M}_C$ is set to *gbest*.

2.9.1. Improvement to DStPSO with Swarm Size Decay

As the swarm size increases, so too does the exploratory power of DStPSO and the computational resources required for particle position updates. A balance between these two outcomes can be exploited by starting out with a large N_Q compared to what is used in practice (usually between 10 to 40 particles for solving the TOP), and then reducing the size of Q on each iteration by keeping the best performing particles until a minimum size is reached. With this modification, DStPSO begins with a wide exploration of the solution space, providing a better chance of pioneering a near optimal *gbest* early. Computational resources are then freed on each iteration as low-scoring particles are selectively removed. The swarm size decay algorithm (Alg. 2) uses a decay factor $0 < \gamma \ll 1$.

Algorithm 2: Swarm size decay algorithm

input : N_Q, Q, $pbest$, γ, minimum swarm size N_{min}

output: N_Q, Q

if $(\gamma > 0 \ \ \& \ \ N_Q > N_{min})$ **then**

  $N_Q \leftarrow \texttt{round}(1 - \gamma \times N_P)$;

  if $N_Q < N_{min}$ **then**

    $N_Q \leftarrow N_{min}$;

  $S_{pbest} \leftarrow \texttt{fitness}(pbest)$;

  $[\_, ID] \leftarrow \texttt{sort}(S_{pbest})$;

  $fittest \leftarrow ID(1 : N_Q)$;

  $Q \leftarrow Q(fittest)$;

return N_Q, Q

3. Results and Discussion

The objective of this section is to demonstrate that the mission planning framework effectively plans multi-vehicle missions. We first evaluate the Feasibility Preprocessing Module and Sequencing and Allocation Module placeholders on several TOP test instance data sets for computation time and quality of the resulting outputs in Section 3.1. Then the case study "Wind Turbine Inspection Mission" data set is presented in (Section 3.2), which we then use to demonstrate the complete mission planning procedure as well as evaluate the Path Planning Module (Section 3.2.2) placeholder for performance and quality of the resulting outputs.

3.1. TOP Test Set Evaluation

The TOP has seven test instance data sets designed by [2, 16] for the purposes of benchmarking TOP solvers. Each test instance contains a mapping of the available tasks in the cost space (i.e. the Euclidean distances between each task is the cost to transition between the tasks in either direction) and a consistent reward score assigned to each task. Each data set contains variations of the maximum allowable cost constraint and the number of team members allocated. The seven data sets have been unpacked into 387 test problems, collected from [36]. A summary of the data is provided in Section B.1. The Feasibility Preprocessing Module was tested on the test instances, and has been summarised in Section B.2. The proposed placeholder using Alg. 1 computes feasible zones for the majority of the test instance variants in under 10 s for problem sizes containing up to 102 tasks.

3.1.1. Swarm Size Decay Evaluation

The DStPSO solver was modified with Alg. 2 that prunes the swarm of the poorest performing particles on each iteration. A performance comparison between the original DStPSO ($\gamma = 0$) and the decayed DStPSO ($\gamma > 0$) was made using the TOP test instances.

The DStPSO algorithm was initialised with inertia weight $w = 0.7$, social bias weight $c_1 = 0.5$, self bias weight $c_2 = 0.5$. Stopping criteria is achieved after 300 consecutive iterations of no improvement (stall). RVNS was set to move from neighbourhood 2 to neighbourhood 3 after 10 consecutive iterations of no improvement, and stopping criteria was set to trigger after 20 consecutive iterations of no improvement from neighbourhood 3. Three solver configurations (varying in γ) were trialled over 10 repeats, measuring the computational time (CPU), the averaged Relative Percentage Error (RPE) from the best found solution of a particular test instance, and the averaged standard deviation of the RPE (σ). The solver was implemented in MATLAB and tested on an Intel i7-8665U 1.9 GHz CPU with 16 GB of memory. Fig. 5 presents the results averaged over the entire test set.

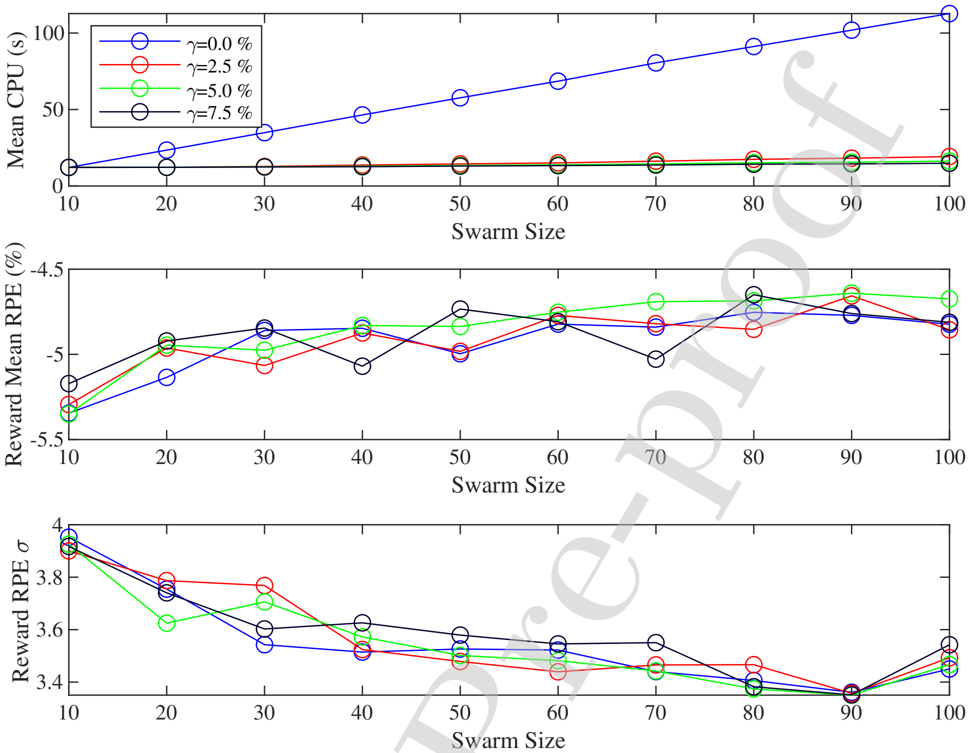


Figure 5: Performance comparison of modified DStPSO algorithm with varying γ. CPU time, reward RPE, and reward RPE σ are averaged over all 387 test sets, repeated 10 times each.

Comparison of RPE and σ across each γ variant shows that larger swarm sizes converge towards a common optimum (σ decreases and RPE increases) with the exception of the largest swarm size (100), whose σ increases. This might be due to the larger swarm producing a wider range of strong particle solutions, and the inertia/social bias/self bias weights need to be retuned to achieve reduced σ. The major performance advantage is observed in CPU time difference. The original DStPSO with a swarm size of 10 has a similar CPU time to the 2.5% DStPSO of swarm size 100, but has a lower RPE and a higher σ. This means that, for the same computation time, DStPSO with swarm decay will provide a better and more consistent solution than DStPSO without swarm decay. The average solving time for the DStPSO algorithm with Alg. 2 ranges between 5.5 s for simple problems and 30 s for more complex problems, and is competitive with other metaheuristic variants (see Section B.3).

3.2. Case Study Application

An example of a structured environment (i.e. an environment where the terrain, static obstacles, and environmental loading conditions are known or can be estimated with a high degree of confidence) are offshore wind farms like the Anholt Wind Turbine Array (Fig. 6), which we use as a case-study application.

Wind turbines require annual inspection of the submerged structure and power cables [37], which is normally completed using ROVs or divers. The distributed inspection mission aims to allocate visual inspection tasks to a fleet of REMORA Autonomous Underwater Vehicles (AUVs), meaning we can use Asms. 1 and 2 for defining $\mathcal{M}_O$. Though the visual inspection of wind turbine substructure and cables is not as difficult a robotic control task as, for instance, underwater valve manipulation on offshore pipelines, the example stands as a proof-of-concept, multi-robot, task allocation and routing problem with variable sea conditions and known obstacles.

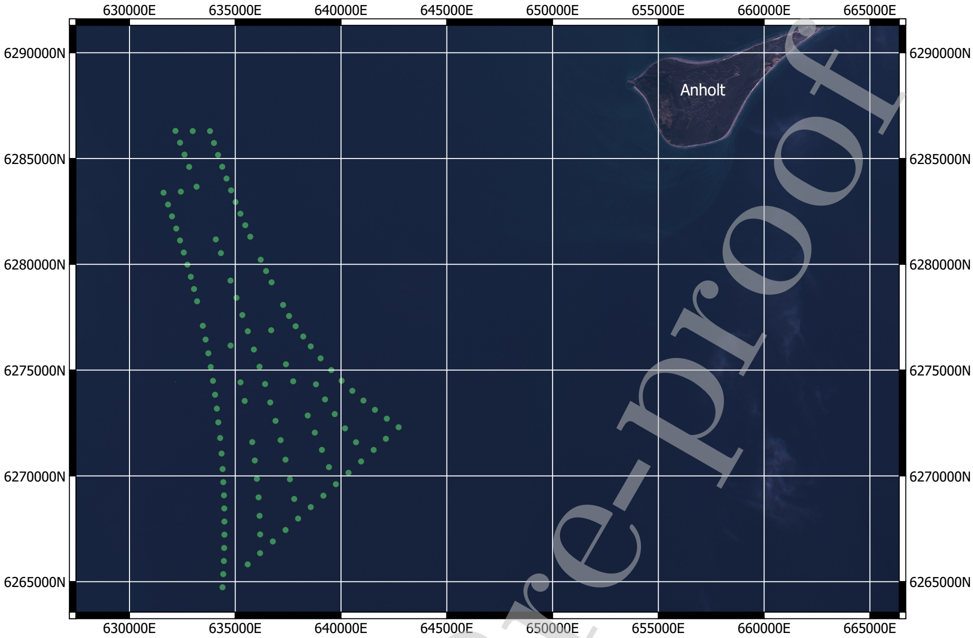


Figure 6: False colour map of Anholt array using infrared satellite imagery from Sentinel-2A (March, 2018) referenced to UTM zone 32N. Green dots indicate the captured centroid of each turbine.

Referring to the AMV mission planning definitions in Section 2.3, the inspection mission needs to first be sub-divided into independent operating zones (defined by N_C instances of $\mathcal{M}_O$) which can then be formulated into the set of inspection tasks and special *home point* tasks, $\mathcal{T}$. The inspection task T is a helical trajectory that the REMORA vehicle must follow to visually inspect the outer surface of a wind turbine substructure from a point close to the waterline to the seabed. All inspection tasks are given an equal reward $s = 1$, and g is set to be 6 m from the centroid of a wind turbine (maintaining a 1 m distance from the exterior of the turbine substructure). There are 111 wind turbines, meaning that $N_T \geq 113$ depending on the feasible operating zone clustering.

One to ten REMORA vehicles will be used for the inspection mission ($N_V = \{1, 2, ..., 9, 10\}$), for the purpose of determining minimum fleet size for complete coverage (i.e. only one deployment per zone is necessary). Each REMORA vehicle can be configured with two 14.8V, 6.2Ah LiPo batteries connected in parallel (12.4Ah total capacity). Each vehicle's available energy capacity, e_b, is then calculated to be approximately 462 kJ from Eq. (6), with 30% of the full capacity kept as an emergency reserve. The parameters of the REMORA dynamic model from Eq. (21) have been empirically determined through model tests by [13]. For the homogeneous fleet assumption (Asm. 2), $\mathcal{V}$ has now been adequately defined.

3.2.1. Evaluation of Feasibility Preprocessing Module

Now that the wind turbine inspection mission is sufficiently defined in terms of task location and vehicle constraint data, the first step of the mission planner procedure is to subdivide the mission area (i.e. the area encapsulated by the location data in $\mathcal{T}$) into feasible operating zones for the vehicles. With a constant forward velocity of $\bar{U} = 0.5$ m/s, the calculated PSR for the REMORA vehicle is 4660.5 m. The PSR is used in Alg. 1 along with the inspection task locations $\mathcal{T}_{2,\ldots,N_T-1}$, to obtain Fig. 7.

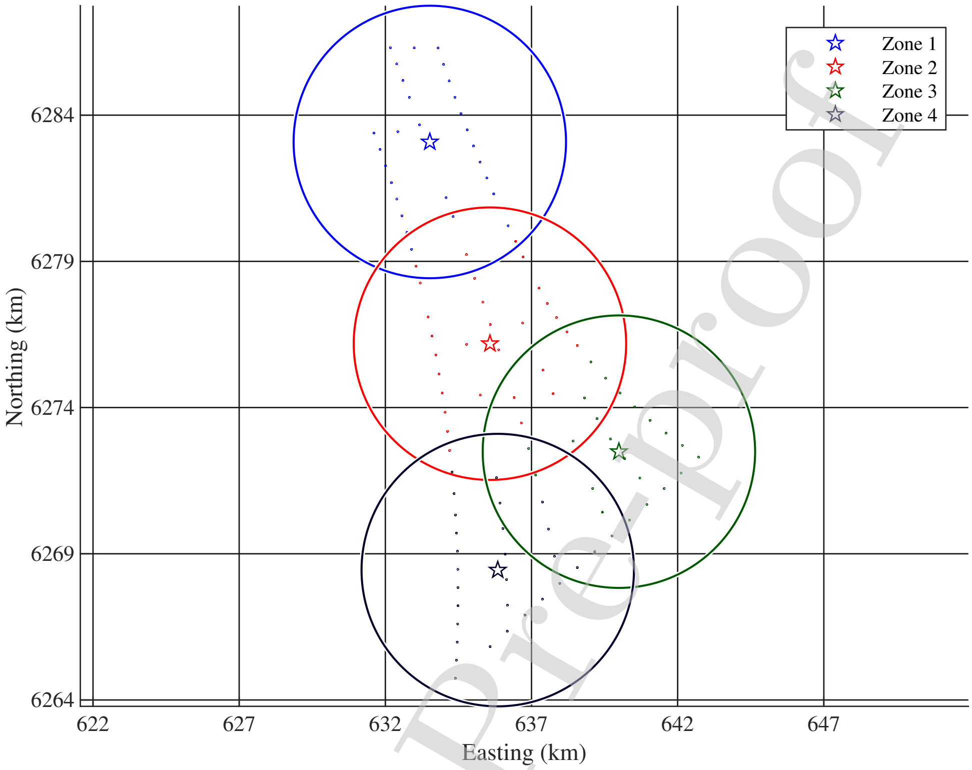
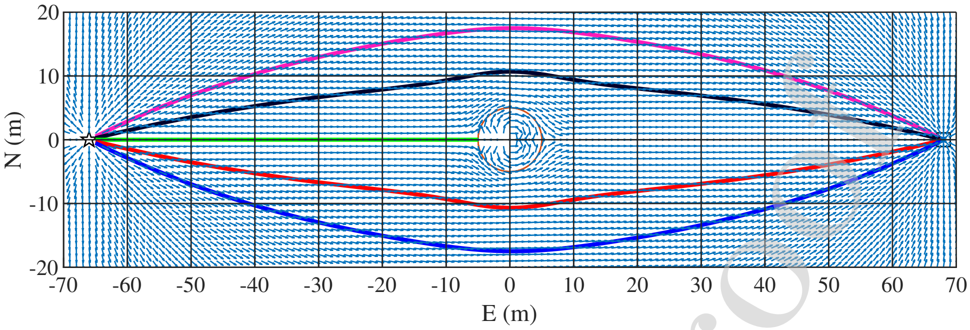


Figure 7: The Anholt wind turbine array, clustered according to the mean PSR of the REMORA vehicle (4.66 km). Each star is a home point and centre of the respective cluster. Overlaps between each zone's PSR and another zone's target set presents the opportunity for inter-zone assistance.

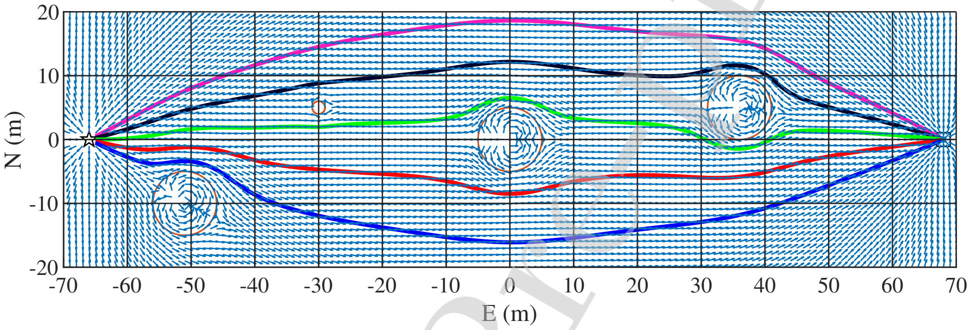
The full $\mathcal{M}_O$ is then decomposed into N_C instances, where $\mathcal{T}$ is distributed to each new $\mathcal{M}_O$ according to the clustering algorithm. The full inspection mission is then formulated into independent N_C sub-missions $\mathcal{M}_O^{(i)} \mid i \in \{1, \ldots, N_C\} \subset \mathbb{N}$. For each $\mathcal{M}_O^{(i)}$, the special *home point* task locations, $g \in \mathcal{T}^{(i)}_{\{1,N_T\}}$, are set to the location of the i-th cluster centroid. Each $\mathcal{M}_O^{(i)}$ can then be digested by the mission planner search algorithm (DStPSO) into a corresponding $\mathcal{M}_C^{(i)}$ for optimisation. But first, $\mathcal{S}$ must be generated for each possible transition in each $\mathcal{M}_O^{(i)}$, so that the corresponding $E^{(i)}$ matrix can be obtained.

3.2.2. Evaluation of Path Planner Module Placeholder

Hydrodynamic potential flow presents an effective solution to obtaining a path $\mathcal{S}$ that navigates around obstacles at a constant forward velocity, but it has two vulnerabilities. Stagnation points on the boundary of an obstacle that cause the vehicle to be trapped in a position of zero velocity, and the generated path having a curvature that cannot be adequately followed due to the vehicle manoeuvrability constraints. Fig. 8a shows an example of the stagnation point causing the vehicle to get stuck in a local minima at the obstacle boundary. This scenario is only likely to happen when there is only one obstacle and its centroid lies on the line between the source and the sink. The influence from multiple obstacles (Fig. 8b), noise from the vehicle's location estimate, and the trajectory tracking error of the vehicle's controller all contribute in reducing the likelihood of the stagnation problem.



(a) Single obstacle.



(b) Multiple obstacles.

Figure 8: Example flow field and vehicle path (streamlines) using hydrodynamic potential flow. The green trajectory in 8a meets with a stagnation point on the surface of the obstacle located at (0,0). In practice this is unlikely to occur as the starting location must lie on the line between source and sink. The influence from multiple obstacles also reduces the likelihood of stagnation as in 8b.

Integrating the CV field (Eq. (12)) from a starting point to a finishing point provides $\mathcal{S}$ based on a mass-less particle drifting along a streamline within the potential field. This ignores the inertial, hydrodynamic, and control components of the vehicle model (see Eq. (21)). The vehicle dynamics may also cause a collision-free trajectory to be invalid because the vehicle is unable to follow the path. This is due to the required turning rate, r, becoming too high for the vehicle's forward velocity, causing an error offset that the vehicle's controller cannot stabilise fast enough. This is likely to happen when the vehicle's trajectory is heading towards the centroid of an obstacle, requiring a large deflection around the obstacle by the integrated CV field. By artificially inflating the size of the obstacles, the radius of curvature of the generated path becomes larger, hence decreasing the magnitude of the required r. Fig. 9 shows a test trajectory generated for the REMORA vehicle model that must navigate around a circular obstacle to reach the position (17,0.1). $\mathcal{S}$ contains the attained x and y positions, commanded forward speed U, and commanded heading ψ for the integrated time series t. Fig. 10 presents the commanded and achieved dynamics of the vehicle for the test trajectory when in autopilot and dynamic positioning modes, showing that the controller is able to adequately track the commands obtained from the integration of the CV field with the inflated obstacle.

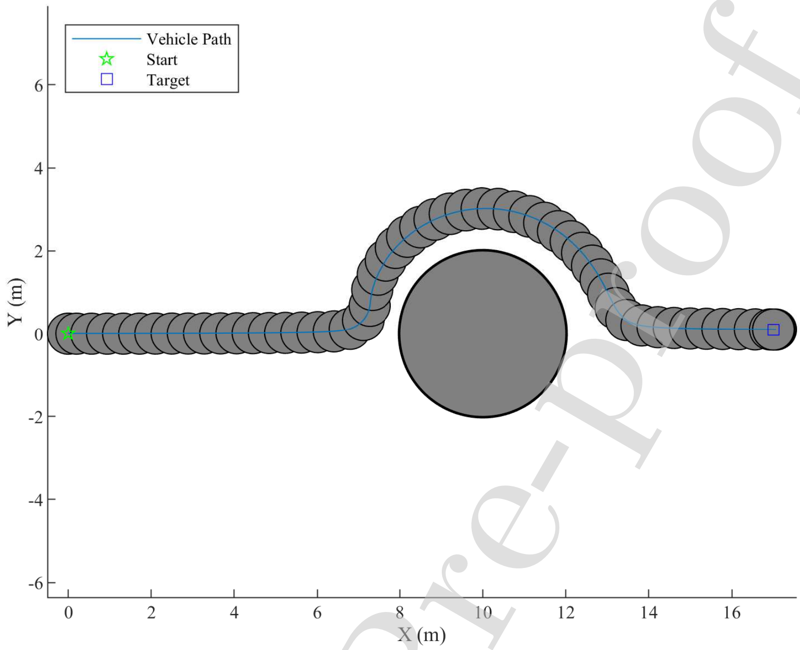


Figure 9: Path navigated by the REMORA Simulink model. Model started at (0,0) with orientation 0 radians (parallel to x axis) and was commanded to navigate to (17,0.1) using the CV flow equation. An obstacle, pictured at actual size, located at (10,0) with radius 5.0 m was inflated by 1.5 m (three times the vehicle's collision radius) for the CV field equations. The resulting path produces a trajectory with curvature suitable for the vehicle to track, avoiding collision with the actual obstacle.

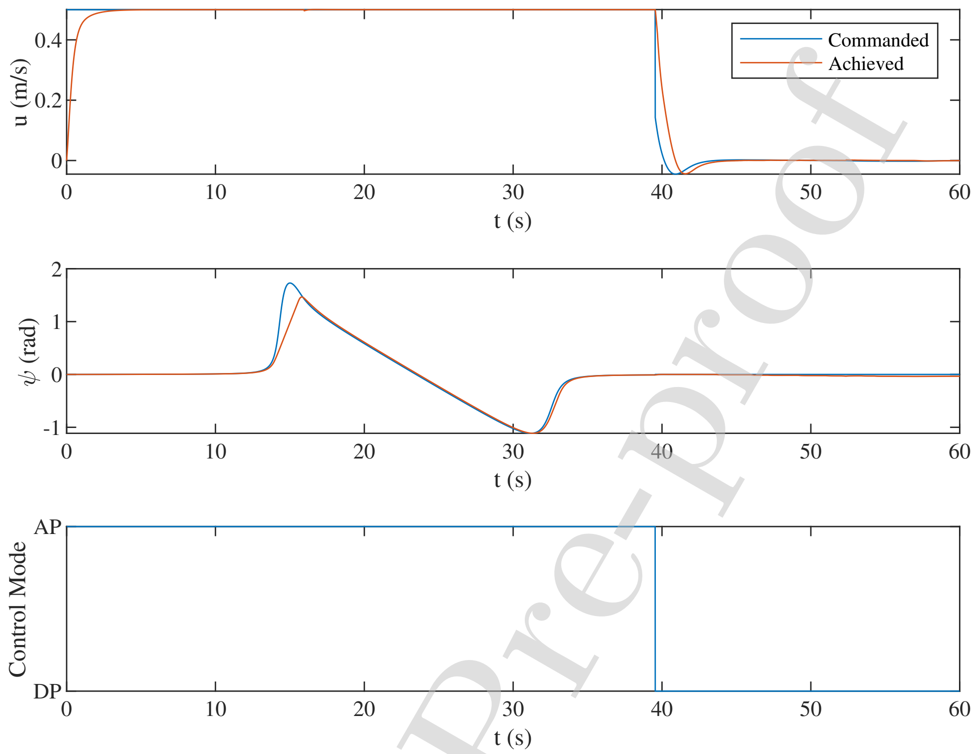


Figure 10: Top: Commanded and actual forward speed of the model during the transition. Middle: Commanded and actual heading of the model during the transition. Bottom: The control method switches between Autopilot (AP) and Dynamic Positioning (DP) mode when the vehicle gets within 0.2 m of the destination. DP enables high manoeuvrability and control but consumes more energy than AP.

3.2.3. Sequencing and Allocation Module Placeholder

As determined in Section 3.2.1, there are four sub-missions that must be solved by the DStPSO algorithm in order to provide a complete plan for the inspection mission of the Anholt array. As shown in Fig. 2, the trajectory generator requires knowledge of the static obstacles' positions and radii (which are provided from the obstacle database as the tuple $O = (X_o, r_o, I_o)$) and the start and end points for the trajectory ($g \in \mathcal{T}_{\{i,j\}}$) for it to produce the requested trajectory $\mathcal{S}_{ij}$.

For the inspection mission case study, each turbine substructure is a pile 5 m in radius, whose cross-section can be represented on the East-North (X-Y) plane as circles of 5 m radius. O is then the collection of 111 X_o coordinates of each turbine location, and r_o is the collection of the corresponding 111 substructure radii, which are all set to 5 m. Given the starting (i-th) and finishing (j-th) coordinates from the k-th sub-mission proposal, $\{g \in \mathcal{T}_{\{i,j\}} | \mathcal{M}_C^{(k)}, O^{(k)}\}$, the trajectory generator can produce $\mathcal{S}_{ij}$ for each sub-mission.

Each $\mathcal{S}_{ij}$ produced by the trajectory generator is mapped to the corresponding element E_{ij} using the method in Section 2.5. $\mathcal{S}_{ij}$ provides the time interval over which $E_{s,ij}$ and E_h are obtained. $E_{t,j}$ is obtained from the nested energy consumption prediction of the helical inspection task, which will have a different $\mathcal{H}$ and $\mathcal{P}$ from the transition phase because special inspection equipment (cameras, sonar, etc.) will be active at this point in the task, and the 3D trajectory taken by the vehicle around the substructure is significantly different from the planar transition trajectory. For the sake of brevity, we have assigned the expected task energy consumption $E_{t,j} = 1\,\mathrm{kJ}\,\forall E_{t,j} \in E$, meaning that a constant is depleted from the vehicle's battery for every task it completes.

Having obtained $E \in \mathcal{M}_O^{(k)}$, the DStPSO algorithm is used to evaluate an optimum $\mathcal{M}_C^{(k)*}$ as described in Section 2.9. The final $gbest$ corresponds to $R \in \mathcal{M}_C^{(k)*}$. The route for the vehicle $l \in Q$, R_l can then be used to access the set of trajectories $\{\mathcal{S}_{ij} \mid (i, j) \in R_l, 1 \leq i \leq L_l - 1, 2 \leq j \leq L_l\}$. The DStPSO algorithm was then calibrated for a swarm size of 1000 and $\gamma = 1\%$, which gives the solver the same expected computation time as a 60-70 particle DStPSO without decay (as interpolated from Fig. 5, given the computational cost presented in Section 3.2.4), but with more than 10x the initial search power. We present the set of trajectories proposed by $\mathcal{M}_C^{(2)}$ in Fig. 11. The full set of solved routes, spanning from a solo REMORA vehicle to a fleet of 10, for each zone are available in Section C.

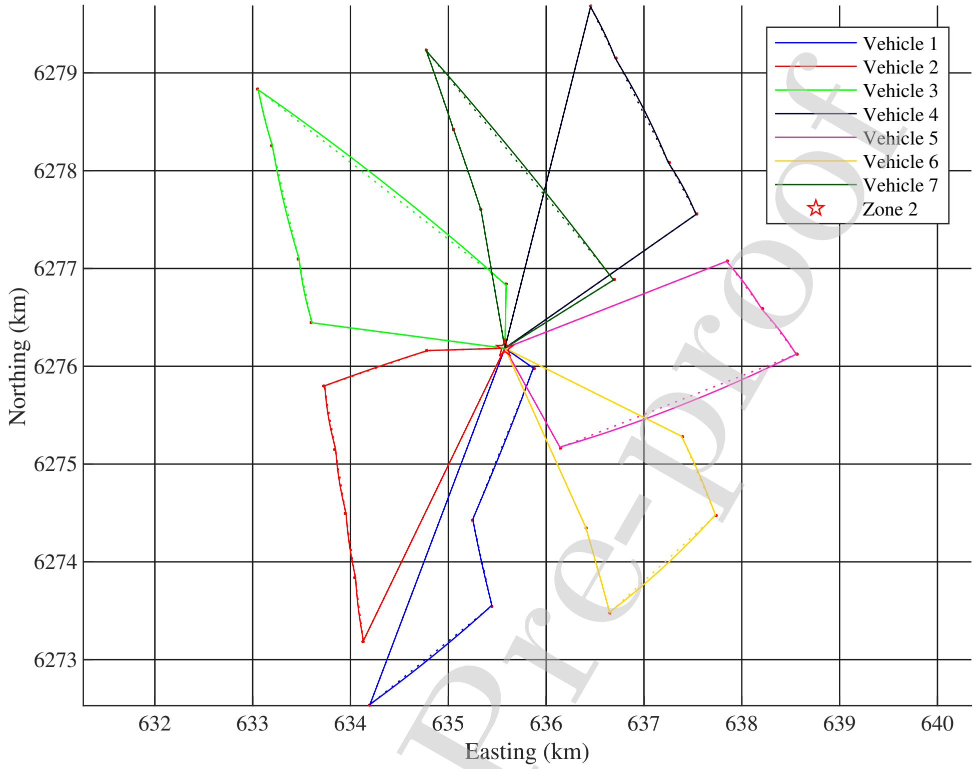


Figure 11: Optimised route for cluster zone 2, with a fleet of seven REMORA vehicles. For visibility, obstacles are not to scale and are larger than actual. Dotted lines represent the Euclidean path of the corresponding vehicle route. As is the case with most of the transitions, the shortest path was a straight line path to the destination, with small distortions in the path that flow around obstacles.

As presented by the planner, at least seven vehicles are required to complete the mission with a single deployment/retrieval (i.e. the smallest number of vehicles required to visit all of the wind turbines). However, it is of more interest to identify how the TOP solver behaves when the fleet size is larger than the bare minimum requirement for complete coverage (i.e. when there is redundancy). Fig. 12 shows that the vehicle within the fleet that is undertaking the largest energy consuming route (i.e. the vehicle with the highest risk of stranding) does not significantly change with increase in fleet size. What can be seen is that, despite the total utilisation of the fleet decreasing with fleet sizes of 7, 8, 9 and 10, the total utilisation of the 'weakest link' vehicle remains largely unchanged at 99.8 % or above of the vehicle's rationed energy capacity (70 % of the theoretical maximum). This is because the DStPSO's node insertion and exchange operations are conditionally implemented either to increase the total achieved reward (maximising tasks completed) or to decrease the total energy cost of a vehicle route so that more potential tasks can be added in future operations.

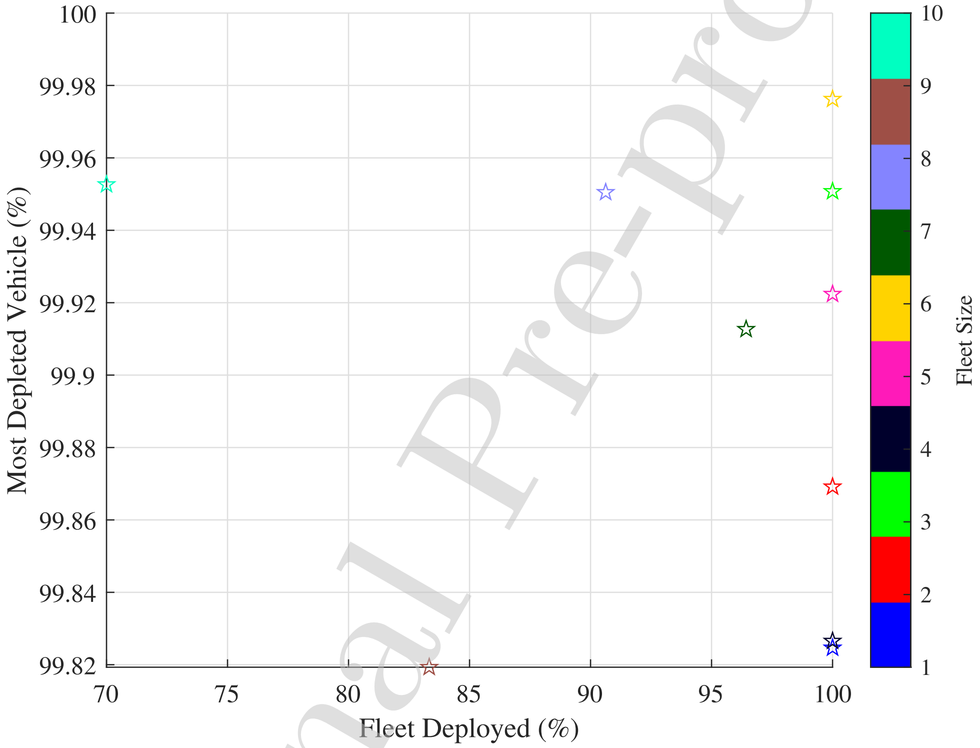


Figure 12: The highest energy consumption (weakest link) of any member of the fleet during the Anholt mission, expressed as the percentage of total rationed energy capacity (i.e. 70% of the theoretical maximum of the battery), versus the utilisation of the fleet, expressed as the percentage of expected energy consumption to the total rationed energy capacity of the fleet.

While this is still a robust solution because the vehicle has a 30 % emergency reserve, it might be advantageous to introduce a penalty based on the standard deviation of the fleet energy cost. In this way, solutions that have similar energy costings for each vehicle will be more likely to be selected than solutions that have a subset of vehicles doing most of the work, minimising the risk of stranding due to energy depletion across the entire fleet. However, this would come with the cost of using more vehicles than absolutely necessary, which increases the complexity of the mission and the risk of other mishaps associated with autonomous deployments (see [38] for real world examples of such problems).

Based on the definition of optimality given in Section 2.2, all of the solved routes found by the planner have the following characteristics, indicating strong solutions:

1. All vehicles have unique tasks (no task is visited more than once).
2. None of the individual vehicle paths cross over themselves.
3. The energy consumptions of the *deployed* vehicles are close to the specified capacity.
4. Each vehicle starts and finished at the designated zone home point.

3.2.4. Computational Cost Breakdown

As an alternative to determining the time complexity of the mission planner framework's placeholder modules, we instead present the computation time benchmark taken for the planner to complete planning for the Anholt mission from start to finish, ranging in fleet size from 1 to 10 (Fig. 13).

The time cost is distributed between the four module placeholders. As can be seen, the feasibility prepocessing, and sequencing and allocation module placeholders are within practical online limits, taking at most under a minute for the DStPSO algorithm to produce a near-optimum M_c. The computational cost is dominated by the path planning and energy estimation modules, taking up to 27 minutes combined to process 1056 potential transitions for the largest zone (zone 2, $N = 33$). The major contributor to this cost is that the step size used to evaluate the trajectory of the REMORA was too small (set to 0.01 s). Over a 4.5 hour long mission, the number of points used to represent the trajectory was perhaps needlessly large. Increasing the step size will significantly reduce the amount of computation time, but at the cost of making the energy estimation less reliable.

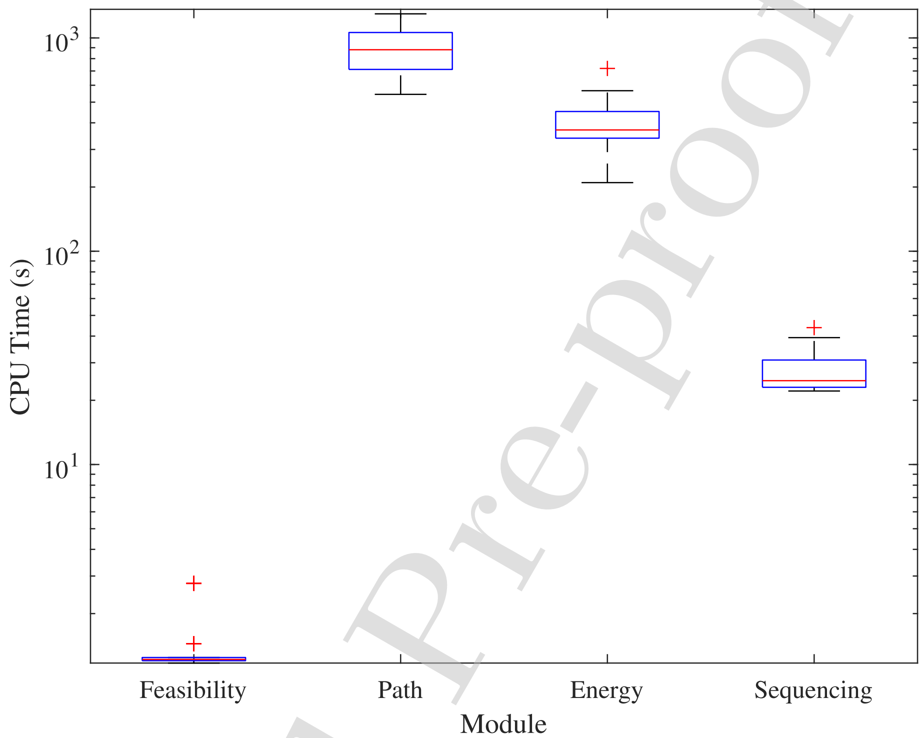


Figure 13: Computation time cost distribution for each module placeholder of the mission planner framework when planning the Anholt wind turbine inspection mission.

The energy estimation module placeholder compounds the cost by performing Monte Carlo of a vehicle model, meaning that the energy for one trajectory is simulated many times (in this case, 100 times). In general, computation time can be reduced for both of these modules by the supporting computer code for parallelisation, either for multi-CPU or GPU acceleration. However, better performing algorithms than the ones proposed within the placeholders could be implemented as modules to improve the time efficiency of the framework.

4. Limitations and Future Work

The proposed module placeholders are basic implementations designed to demonstrate the feasibility of the framework design as well as to provide a benchmark for module design and improvement. The top candidate for improvement is the hydrodynamic potential implementation of the path planning module because of its limitations in the planning domain (only in two dimensional space), modelling all obstacles as circles, assuming that all obstacles are static, the presence of stagnation points, and the long time taken to solve Eq. (12). Future path planning modules should implement existing algorithms (such as are considered in [39]) that can overcome these limitations, which will improve the capabilities of the mission planning framework.

The second place candidate for improvement is the energy estimation module placeholder. Firstly, the dynamic model can be developed further to include environmental parameters such as wind speed and direction, wave spectra, and water current and direction. The resulting energy cost matrix will be more asymmetrical, and result in more interesting route solutions produced by the sequencing and allocation module in order to reduce energy consumption. Secondly, the underlying Monte Carlo simulation assumes Gaussian distributions for the vehicle model but this may not be the case in reality. More appropriate vehicle model distributions may be obtained through machine learning on the mission history of real vehicles.

Finally, the sequencing and allocation module can also be improved in several key ways. The DStPSO solver is currently configured to accept deterministic costs from the energy estimation module. Currently, an upper confidence value (i.e. a value larger than or equal to the mean) of the estimated distribution for each transition is inputted to the DStPSO solver. However, the energy estimation module outputs a Gaussian distribution parameterised by estimated mean and standard deviation for each transition. A stochastic solver could take advantage of the full distribution to provide more reliable solutions.

The search space of the DStPSO solver has only been configured for assumptions of a homogeneous fleet and non-hierarchical mission. The solver can be extended to include heterogeneous fleets of vehicles in three ways:

1. Defining E as the matrix $\mathbb{R}^{N_T \times N_T \times N_V}$ allows for different vehicle dynamic models to be used according to $I_v \in V$.
2. Extending P as a N_V long set of control vectors that reference $\{T_{P_i} \in \mathcal{T} \mid 1 \leq i \leq N_V, i \in \mathbb{N}\}$, essentially defining the set of tasks in $\mathcal{M}_O$ that each vehicle is capable of doing according to $I_v \in V$.
3. Modifying $s \in S$ by a scalar utility variable found in $I_v \in V$ allows for vehicles more effective at completing certain task types than others to evaluate as a higher scoring solution than alternative solutions.

Additionally, the planner can be extended to include hierarchical tasks (i.e. tasks that depend on the completion of other tasks) by specifying a prerequisite variable in $I_t \in T$. This allows for a logical hierarchy, but then must be further extended using temporal logic in order to obtain an energy efficient hierarchical proposal.

Finally, the planner's scope has currently been configured for pre-planning operation (i.e. generating an initial plan for the vehicles). However, it is compelling to obtain online planning for the vehicles as they progress through the mission after deployment, and experience energy consumptions that are almost certainly different from what was estimated in the initial plan. Detecting significant deviation from the expected energy consumption is the primary challenge to be overcome to effectively implement an online replanning system. The computational cost of the path planning and energy planning placeholders also make it impractical to perform a complete replan of the all possible transitions following identification of a replan. Our future objectives are to develop an online replanning component to the mission planner framework which can identify when a replanning situation is necessary, and also to reduce the replanning space to a size that would be practical to implement onboard a vehicle *in situ*.

Conclusions

A new mission planner framework for AMVs was proposed, formulated as the TOP from operational research. The mission domain was first defined in its open form, containing information about the tasks, vehicles, and obstacles as specified by the AMV operator, the *knowledge based* *reasoning* step discussed in section 1. The mission planner searches through the open mission $\mathcal{M}_O$ for an optimum proposal, called the closed mission $\mathcal{M}_C$. The closed mission is an initial plan that contains task allocation and sequencing information for the AMV fleet to execute. Here it can be seen that the *task allocation* step of mission planning has been completed.

The planner differs from temporal planners and task hierarchy planners because it uses energy has the base finite resource. Considering energy means considering the loading of a particular vehicle over the extent of its mission. It requires a trajectory generator to produce viable paths that can be assessed for energy consumption using the dynamic model of the vehicle under consideration. Treating energy consumption as the expected variable of a stochastic process means that uncertainty has been considered by the planner. This is the foundation of the third step in mission planning, *risk projection*. In future development of the mission planner, the level of allowable uncertainty in the mission plan can be specified by the operator as a constraint.

The mission planner is modular in nature because the definition of the open mission requires several separate databases to be processed into the open mission formulation. This means that the components specified in section 2 are interchangeable with different or more advanced methods, depending on the complexity of the mission.

The integration of the components into the mission planner framework also produced 'spillover effects' as minor improvements to the literature concerning some of the components. Most notable is the improvement of the DStPSO algorithm with the swarm decay modification. It is shown in section 3.1.1 that the modification allows for a wider initial exploration of the search space with more particles whilst saving computational resources in the later stages of the search.

Finally, we tested the mission planner framework on simulated operator input data from the case-study inspection mission of the Anholt wind turbine array. Following the homogeneous fleet and non-hierarchical task assumptions stated in section 2.3, we formulate the test data into four separate open missions using the feasible operating zone component. DStPSO was then shown to successfully obtain the closed mission proposal for each instance.

The proposed AMV mission planner stands as the preliminary framework for development towards robust automated planning for more generalised AMV missions. We hope to promote development and comparison of new framework modules through our results benchmark and our identification of the limitations in our implementation.

Acknowledgements

The authors acknowledge the support of the Australian Government's Research Training Programme, the Institute of Marine Engineering, Science and Technology's Laurie Prandolini Research Fellowship, and the Department of Automation and Control at the Technical University of Denmark for hosting the corresponding author during production of this paper.

References

[1] R. R. Murphy, E. Steimle, C. Griffin, C. Cullins, M. Hall, K. Pratt, Cooperative use of unmanned sea surface and micro aerial vehicles at Hurricane Wilma, Journal of Field Robotics 25 (3) (2008) 164–180. doi:10.1002/rob.20235.
URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.20235

[2] I. Chao, B. L. Golden, E. A. Wasil, The team orienteering problem, European Journal of Operational Research 88 (3) (1996) 464 – 474. doi:10.1016/0377-2217(94)00289-4.
URL http://www.sciencedirect.com/science/article/pii/0377221794002894

[3] F. Py, J. Pinto, M. A. Silva, T. A. Johansen, J. Sousa, K. Rajan, EUROPtus: A mixed-initiative controller for multi-vehicle oceanographic field experiments, in: International Symposium on Experimental Robotics, ISER, Tokyo, Japan, 2016, pp. 323–340. doi:10.1007/978-3-319-50115-4_29.
URL https://doi.org/10.1007/978-3-319-50115-4_29

[4] C. Lesire, G. Infantes, T. Gateau, M. Barbier, A distributed architecture for supervision of autonomous multi-robot missions, Autonomous Robots 40 (7) (2016) 1343–1362. doi:10.1007/s10514-016-9603-z.
URL https://doi.org/10.1007/s10514-016-9603-z

[5] C. C. Sotzing, D. M. Lane, Improving the coordination efficiency of limited-communication multi–autonomus underwater vehicle operations using a multiagent architecture, Journal of Field Robotics 27 (4) (2010) 412–429. doi:10.1002/rob.20340.
URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.20340

[6] M. Ai-Chang, J. Bresina, L. Charest, A. Chase, J. C. J. Hsu, A. Jonsson, B. Kanefsky, P. Morris, K. Rajan, J. Yglesias, B. G. Chafin, W. C. Dias, P. F. Maldague, MAPGEN: mixed-initiative planning and scheduling for the Mars exploration rover mission, IEEE Intelligent Systems 19 (1) (2004) 8–12. doi:10.1109/MIS.2004.1265878.

[7] B. P. Gerkey, M. J. Matarić, A formal analysis and taxonomy of task allocation in multi-robot systems, The International Journal of Robotics Research 23 (9) (2004) 939–954. doi:10.1177/0278364904045564.
URL https://doi.org/10.1177/0278364904045564

[8] C. McGann, F. Py, K. Rajan, H. Thomas, R. Henthorn, R. S. McEwen, A deliberative architecture for AUV control, in: 2008 IEEE International Conference on Robotics and Automation, ICRA, Pasadena, USA, 2008, pp. 1049–1054. doi:10.1109/ROBOT.2008.4543343.
URL https://doi.org/10.1109/ROBOT.2008.4543343

[9] C. C. Sotzing, J. Evans, D. M. Lane, A multi-agent architecture to increase coordination efficiency in multi-AUV operations, in: OCEANS 2007 - Europe, Aberdeen, UK, 2007, pp. 1–6. doi:10.1109/OCEANSE.2007.4302393.

[10] G. F. Giger, An operator-centric mission planning environment to reduce mission complexity for heterogeneous

unmanned systems, Ph.D. thesis, University Park, PA, USA, AAI3436143 (2010).

[11] S. Waydo, R. M. Murray, Vehicle motion planning using stream functions, in: 2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422), Vol. 2, Taipei, Taiwan, 2003, pp. 2484–2491 vol.2. doi:10.1109/ROBOT.2003.1241966.

[12] T. Owen, R. Hillier, D. Lau, Smooth path planning around elliptical obstacles using potential flow for non-holonomic robots, in: T. Röfer, N. M. Mayer, J. Savage, U. Saranlı (Eds.), RoboCup 2011: Robot Soccer World Cup XV, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 329–340. doi:10.1007/978-3-642-32060-6_28.

[13] M. Nielsen, O. Eidsvik, M. Blanke, I. Schjølberg, Constrained multi-body dynamics for modular underwater robots — theory and experiments, Ocean Engineering 149 (2018) 358–372. doi:10.1016/j.oceaneng.2017.12.007.

[14] Z. Sevkli, F. E. Sevilgen, Discrete particle swarm optimization for the orienteering problem, in: IEEE Congress on Evolutionary Computation, Barcelona, Spain, 2010, pp. 1–8. doi:10.1109/CEC.2010.5586532.

[15] M. N. Ab Wahab, S. Nefti-Meziani, A. Atyabi, A comprehensive review of swarm optimization algorithms, PLOS ONE 10 (5) (2015) 1–36. doi:10.1371/journal.pone.0122827.
URL https://doi.org/10.1371/journal.pone.0122827

[16] T. Tsiligirides, Heuristic methods applied to orienteering, Journal of the Operational Research Society 35 (9) (1984) 797–809. doi:10.1057/jors.1984.162.
URL https://doi.org/10.1057/jors.1984.162

[17] N. Labadie, R. Mansini, J. Melechovský, R. W. Calvo, The team orienteering problem with time windows: An LP-based granular variable neighborhood search, European Journal of Operational Research 220 (1) (2012) 15 – 27. doi:https://doi.org/10.1016/j.ejor.2012.01.030.
URL http://www.sciencedirect.com/science/article/pii/S0377221712000653

[18] L. Evers, K. Glorie, S. van der Ster, A. I. Barros, H. Monsuur, A two-stage approach to the orienteering problem with stochastic weights, Computers & Operations Research 43 (2014) 248 – 260. doi:https://doi.org/10.1016/j.cor.2013.09.011.
URL http://www.sciencedirect.com/science/article/pii/S0305054813002815

[19] J. Li, Model and algorithm for time-dependent team orienteering problem, in: S. Lin, X. Huang (Eds.), Advanced Research on Computer Education, Simulation and Modeling, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 1–7.

[20] A. Gunawan, H. C. Lau, P. Vansteenwegen, Orienteering problem: A survey of recent variants, solution approaches and applications, European Journal of Operational Research 255 (2) (2016) 315 – 332. doi:https://doi.org/10.1016/j.ejor.2016.04.059.
URL http://www.sciencedirect.com/science/article/pii/S037722171630296X

[21] Z. Sevkli, F. E. Sevilgen, Discrete particle swarm optimization for the team orienteering problem, Turkish Journal for Electrical Engineering and Computer Sciences 20 (2) (2012) 231–239. doi:10.3906/elk-1101-1008. URL http://journals.tubitak.gov.tr/elektrik/issues/elk-12-20-2/elk-20-2-4-1101-1008.pdf

[22] F. J. Richards, A flexible growth function for empirical use, Journal of Experimental Botany 10 (2) (1959) 290–301. doi:10.1093/jxb/10.2.290.
URL http://dx.doi.org/10.1093/jxb/10.2.290

[23] M. E. Furlong, D. Paxton, P. Stevenson, M. Pebody, S. D. McPhail, J. Perrett, Autosub Long Range: A long range deep diving AUV for ocean monitoring, in: 2012 IEEE/OES Autonomous Underwater Vehicles (AUV), Southampton, UK, 2012, pp. 1–7. doi:10.1109/AUV.2012.6380737.

[24] B. W. Hobson, J. G. Bellingham, B. Kieft, R. McEwen, M. Godin, Y. Zhang, Tethys-class long range AUVs - extending the endurance of propeller-driven cruising AUVs from days to weeks, in: 2012 IEEE/OES Autonomous Underwater Vehicles (AUV), Southampton, UK, 2012, pp. 1–8. doi:10.1109/AUV.2012.6380735.

[25] T. T. Mac, C. Copot, D. T. Tran, R. D. Keyser, Heuristic approaches in robot path planning: A survey, Robotics and Autonomous Systems 86 (2016) 13 – 28. doi:https://doi.org/10.1016/j.robot.2016.08.001.
URL http://www.sciencedirect.com/science/article/pii/S0921889015300671

[26] L. E. Kavraki, P. Svestka, J. C. Latombe, M. H. Overmars, Probabilistic roadmaps for path planning in high-dimensional configuration spaces, IEEE Transactions on Robotics and Automation 12 (4) (1996) 566–580. doi:10.1109/70.508439.

[27] J. J. Kuffner, S. M. LaValle, RRT-connect: An efficient approach to single-query path planning, in: Proceedings of 2000 ICRA Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065), Vol. 2, San Francisco, USA, 2000, pp. 995–1001 vol.2. doi:10.1109/ROBOT.2000.844730.

[28] R. A. Brooks, T. Lozano-Pérez, A subdivision algorithm in configuration space for findpath with rotation, IEEE Transactions on Systems, Man, and Cybernetics SMC-15 (2) (1985) 224–233. doi:10.1109/TSMC.1985.6313352.

[29] S. Choi, J. Y. Lee, W. Yu, Fast any-angle path planning on grid maps with non-collision pruning, in: 2010 IEEE International Conference on Robotics and Biomimetics, Tianjin, China, 2010, pp. 1051–1056. doi:10.1109/ROBIO.2010.5723473.

[30] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, in: Proceedings. 1985 IEEE International Conference on Robotics and Automation, Vol. 2, St. Louis, USA, 1985, pp. 500–505. doi:10.1109/ROBOT.1985.1087247.

[31] M. D. Pedersen, T. I. Fossen, Marine vessel path planning & guidance using potential flow, IFAC Proceedings Volumes 45 (27) (2012) 188 – 193, 9th IFAC Conference on Manoeuvring and Control of Marine Craft. doi:

https://doi.org/10.3182/20120919-3-IT-2046.00032.
URL https://www.sciencedirect.com/science/article/pii/S1474667016312265

[32] L. Milne-Thomson, Theoretical Hydrodynamics, 5th Edition, Dover Books on Physics, Dover Publications, Mineola, USA, 1996, p. 154.

[33] D. Mester, O. Bräysy, W. Dullaert, A multi-parametric evolution strategies algorithm for vehicle routing problems, Expert Systems with Applications 32 (2) (2007) 508 – 517. doi:https://doi.org/10.1016/j.eswa.2005.12.014.
URL http://www.sciencedirect.com/science/article/pii/S095741740500360X

[34] P. Hansen, N. Mladenović, J. A. Moreno Pérez, Variable neighbourhood search: methods and applications, Annals of Operations Research 175 (1) (2010) 367–407. doi:10.1007/s10479-009-0657-6.
URL https://doi.org/10.1007/s10479-009-0657-6

[35] G. A. Croes, A method for solving traveling-salesman problems, Operations Research 6 (6) (1958) 791–812. doi:10.1287/opre.6.6.791.
URL https://doi.org/10.1287/opre.6.6.791

[36] KU Leuven, The Orienteering Problem: Test Instances, https://www.mech.kuleuven.be/en/cib/op, accessed: 2018-22-05 (2018).

[37] Support structures for wind turbines, Standard, DNVGL-ST-0126, Det Norske Veritas - Germanischer Lloyd, Høvik, Norway (2018).
URL http://rules.dnvgl.com/docs/pdf/DNVGL/ST/2018-07/DNVGL-ST-0126.pdf

[38] R. Stokey, T. Austin, C. Von Alt, M. Purcell, R. Goldsborough, N. Forrester, B. Allen, AUV bloopers or why Murphy must have been an optimist: A practical look at achieving mission level reliability in an autonomous underwater vehicle, in: International Symposium on Unmanned Untethered Submersible Technology, 1999, pp. 32–40.

[39] Z. Zeng, K. Sammut, L. Lian, F. He, A. Lammas, Y. Tang, A comparison of optimization techniques for auv path planning in environments with ocean currents, Robotics and Autonomous Systems 82 (2016) 61 – 72. doi:https://doi.org/10.1016/j.robot.2016.03.011.
URL http://www.sciencedirect.com/science/article/pii/S0921889016301713

[40] L. Furno, M. Blanke, R. Galeazzi, D. J. Christensen, Self-reconfiguration of modular underwater robots using an energy heuristic, in: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, Canada, 2017, pp. 6277–6284. doi:10.1109/IROS.2017.8206530.

[41] C. Archetti, A. Hertz, M. G. Speranza, Metaheuristics for the team orienteering problem, Journal of Heuristics 13 (1) (2007) 49–76. doi:10.1007/s10732-006-9004-0.
URL https://doi.org/10.1007/s10732-006-9004-0

[42] H. Tang, E. Miller-Hooks, A tabu search heuristic for the team orienteering problem, Computers & Operations

Research 32 (6) (2005) 1379 – 1407. doi:https://doi.org/10.1016/j.cor.2003.11.008.
URL http://www.sciencedirect.com/science/article/pii/S0305054803003265

A. Detailed Energy Calculations

A.1. Traverse Energy Cost Calculation

The energy required to compensate drag, centripetal and buoyancy forces/moments while traversing the path $\mathcal{S}_{ij}$ is

$$E_{s,ij} = \int_{\mathcal{S}_{ij}} \boldsymbol{\tau}(\mathbf{s})\,\mathrm{d}\mathbf{s}, \tag{19}$$

where $\boldsymbol{\tau} \in \mathbb{R}^6$ is the generalised vector of control forces and moments acting on the vehicle and $\mathbf{s} \in \mathbb{R}^6$ is the path variable. For a vehicle outfitted with N_{th} thrusters the generalised vector $\boldsymbol{\tau}$ is provided as the linear combination of the thrust command vector $\mathbf{f} \in \mathbb{R}^{N_{th}}$ through the thruster configuration matrix $\mathbf{T}_c \in \mathbb{R}^6 \times \mathbb{R}^{N_{th}}$

$$\boldsymbol{\tau} = \mathbf{T}_c\mathbf{f}. \tag{20}$$

Let $\boldsymbol{\eta} = [N, E, D, \phi, \theta, \psi]^{\mathrm{T}} \in \mathbb{R}^6$ be the generalised pose vector of the vehicle in the inertial North-East-Down (NED) frame and $\boldsymbol{\nu} = [u, v, w, p, q, r]^{\mathrm{T}} \in \mathbb{R}^6$ be the generalised linear and angular velocity vector in a body-fixed frame. For a vehicle manoeuvring at constant speed ($\boldsymbol{\nu} = \bar{\boldsymbol{\nu}}$) the generalised control forces and moments balance the hydrostatic and hydrodynamic contributions, i.e.

$$\bar{\boldsymbol{\tau}} = \mathbf{D}(\bar{\boldsymbol{\nu}})\bar{\boldsymbol{\nu}} + \mathbf{C}(\bar{\boldsymbol{\nu}})\bar{\boldsymbol{\nu}} + \mathbf{g}(\bar{\boldsymbol{\eta}}) \tag{21}$$

where $\mathbf{D}(\boldsymbol{\nu}) \in \mathbb{R}^6 \times \mathbb{R}^6$ is the linear plus quadratic drag; $\mathbf{C}(\boldsymbol{\nu}) \in \mathbb{R}^6 \times \mathbb{R}^6$ accounts for Coriolis and centripetal forces and moments; $\mathbf{g}(\boldsymbol{\eta}) \in \mathbb{R}^6$ is the vector of restoring moments and forces. $\bar{\boldsymbol{\tau}}$ represents an underestimate of the total generalised force spent to traverse the path since it does not account for acceleration and deceleration phases.

The electrical power spent by each thruster to deliver the thrust f_k is generally approximated with a quadratic function of the commanded thrust f_k [40], i.e.

$$\Pi_k = \eta_k |f_k| f_k, \quad k = 1, \ldots, N_{th} \tag{22}$$

where η_k is a thrust efficiency coefficient. Therefore the energy required to traverse the path $\mathcal{S}_{ij}$ can be computed as

$$E_{s,ij} = \int_{\mathcal{S}_{ij}} \bar{\boldsymbol{\tau}}(\mathbf{s})\,\mathrm{d}\mathbf{s} = \int_{\mathcal{S}_{ij}} \mathbf{T}_c\bar{\mathbf{f}}(\mathbf{s})\,\mathrm{d}\mathbf{s} = \int_{t_i}^{t_j} \mathcal{P}(t)\,\mathrm{d}t, \tag{23}$$

where $\mathcal{P} = \sum_k \Pi_k$ is the total electric power consumed to traverse the path $\mathcal{S}_{ij}$, t_i is the time instant the vehicle leaves $g \in \mathcal{T}_{P_i}$ and t_j is the time instant the vehicle arrives to $g \in \mathcal{T}_{P_j}$.

A.2. Task Energy Cost Discussion

In addition to expending energy while transitioning between task locations, the vehicle also expends energy undertaking a particular task at location $g \in \mathcal{T}_{P_j}$. Tasks vary in energy intensity. For example, a vehicle tasked with cleaning substructure from bio-fouling will experience higher loads than a vehicle tasked with visual inspection of the same substructure, meaning a higher energy consumption for the former scenario. Therefore an estimate of the task energy $E_{t,j}$ can be computed after the specific definition of the tuple $I_t \in \mathcal{T}_{P_j}$. For example an inspection task as a seabed survey or a scrutiny of a monopile will be defined as a trajectory and a sequence of actions to be performed while passing at given way points. The energy cost associated with the trajectory tracking will then be estimated by means of equation 5; while the energy spent in carrying out the sequence of actions will be evaluated based on the sensors and actuators to be used and the usage duration.

A.3. Hotel Load Energy Cost Calculation

The energy depleted by the hotel load $\mathcal{H}$ is usually accounted for by considering the nominal power consumption of the guidance, navigation, control, communication, environmental sensing and acting systems that are switched on during the mission. An energy baseline for the hotel load can be estimated by considering those systems that must always be available, i.e. the guidance, navigation and control computer with associated sensors and the communication system. Instead of looking into component data sheets for the nominal power consumption declared by the manufacturers, the hotel load $\mathcal{H}$ can be modelled as a random variable by looking into logged data while the vehicle is idle. For the considered REMORA vehicle study case, recorded data of power consumption shows that the baseline hotel load can be modelled as a normally distributed random variable, i.e. $\mathcal{H} \sim \mathcal{N}(\mu_{\mathcal{H}}, \sigma^2_{\mathcal{H}})$. Hence the energy cost of the hotel load is given by

$$\hat{E}_h = E[E_h] = E\left[\int_{t_{s,k}}^{t_{t,k}} \mathcal{H}\,\mathrm{d}t\right] = \int_{t_{s,k}}^{t_{f,k}} E[\mathcal{H}_k]\,\mathrm{d}t \tag{24}$$

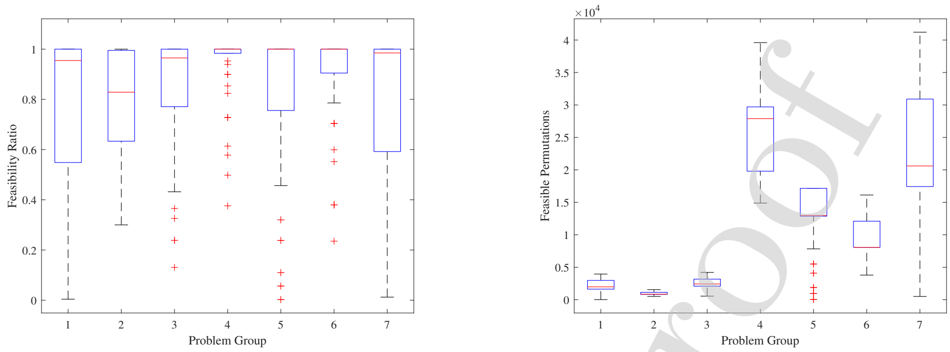
where $t_{s,k}$ and $t_{f,k}$ are the start time of the transition $g \in \mathcal{T}_{P_i} \rightarrow g \in \mathcal{T}_{P_j}$ and the finish time of the task $\mathcal{T}_{P_j}$, respectively.

Remark. As the vehicle executes the mission, different sensors and actuators are powered up in order to fulfil the assigned tasks. This will generate power loads that may change the statistical description of the hotel load towards non-symmetric distributions with heavy tails (e.g. Rayleigh distribution).

B. TOP Test Instance Evaluation

B.1. Data Set Description

The complexity of each data set is represented by two parameters, ”Feasibiliy Ratio” and ”Feasible Permutations”. Feasibility Ratio is calculated by: *the number of transitions whose cost is less than or equal to the total cost constraint, divided by the total number of transitions* ($N^2 - N$). The complexity of each test set is represented as the number of Feasible Permutations (i.e. the Feasibility Ratio multiplied by $N^2 - N$). The distributions of these complexity parameters are shown in Fig. 14.



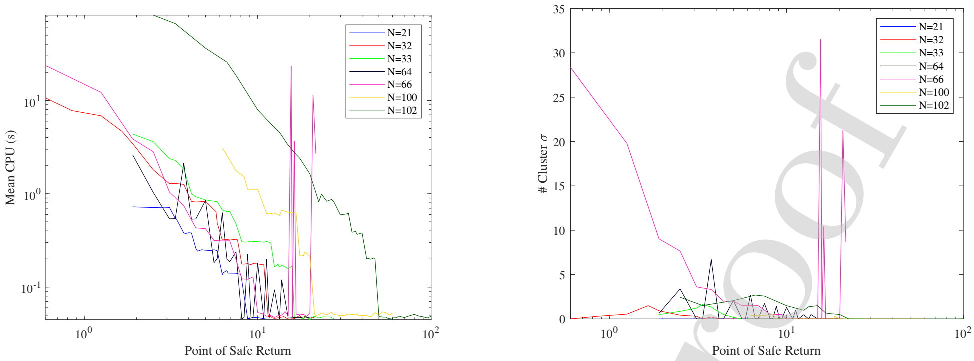
(a) Distribution of Feasibility Ratio.

(b) Distribution of Feasible Permutations.

Figure 14: Distributions summarising the complexity of each test instance data set. 14a summarises the harshness of the cost constraint for the instance. Values closer to 1 correspond to instance variants that have more feasible transitions than ones with values closer to 0. 14b summarises the search space complexity by the number of feasible transitions that can be searched through for each instance. Problems with larger numbers of permutations will take longer to find optimal solutions within.

B.2. Clustering Evaluation

We evaluated the feasible operating zone clustering algorithm (Alg. 1) for performance (in terms of computation time) and quality (in terms of standard deviation of the number of feasible clusters generated). The algorithm was tested on each instance variant 10 times. Fig. 15 presents the average computation time and standard deviation of the number of feasible clusters identified by the algorithm for unique PSR values specified by each instance variant. In general, larger PSR values represent a more relaxed constraint than smaller PSR values, which translates to higher computational times for smaller PSR values as more cluster zones are required to cover all of the targets. However, for some problems such as $N = 64$ and $N = 66$), certain PSR values result in a larger number of required clusters to adequately cover all tasks, resulting in oscillating spikes of cluster number standard deviation and CPU time.



(a) Mean computation time vs. PSR

(b) Standard deviation of number of clusters vs. PSR

Figure 15: Summary of the performance of Alg. 1 on the TOP test instance data set.

B.3. DStPSO Performance Summary on TOP Test Instance Data

As summarised by [20], DStPSO has not made any improvements on the best known solutions since its introduction in 2010. However, the new swarm size decay modification introduces performance increases in computation time which are summarised in Table 1. Several metaheuristic algorithms tested by [41], these are: Generalised Tabu search with a Penalty heuristic (GTP), Generalised Tabu search with a feasible solution heuristic (GTF), Fast Variable neighbourhood search with Feasible solution heuristic (FVF) and Slow Variable neighbourhood search with Feasible solution heuristic (SVF). Additionally, [41] included results from the algorithm proposed by [2] (CGW) and [42] (TMH), which we have also included for convenience. As can be seen, the DStPSO reaches solutions faster on more complex problems (problem sets 4-7) when compared to the other algorithms.

Table 1: Average CPU time comparison between DStPSO with decay rates 0.0 to 7.5 %, and average CPU time results reported by [41] for various metaheuristic algorithms designed to solve the TOP problem instances. DStPSO is less affected by problem complexity compared to others, but takes longer to solve simpler problems

	Average CPU (s)									
	$\gamma = 0.0$	$\gamma = 2.5$	$\gamma = 5.0$	$\gamma = 7.5$	GTP	GTF	FVF	SVF	TMH	CGW
Set 1	39.45	6.99	5.94	5.48	4.67	1.63	0.13	7.78	N/A	15.41
Set 2	37.32	6.53	5.54	5.14	0.00	0.00	0.00	0.03	N/A	0.85
Set 3	52.00	9.06	7.69	7.09	6.03	1.59	0.15	10.19	N/A	15.37
Set 4	161.57	27.63	23.63	21.57	105.29	282.92	22.52	457.89	797.70	934.80
Set 5	142.99	24.36	20.60	18.87	69.45	26.55	34.17	158.93	71.30	193.70
Set 6	143.70	23.86	20.19	18.39	66.29	20.19	8.74	147.88	45.70	150.10
Set 7	173.01	29.57	24.82	22.47	158.97	256.76	10.34	309.87	432.60	841.40

C. Anholt Mission Plan Evaluation

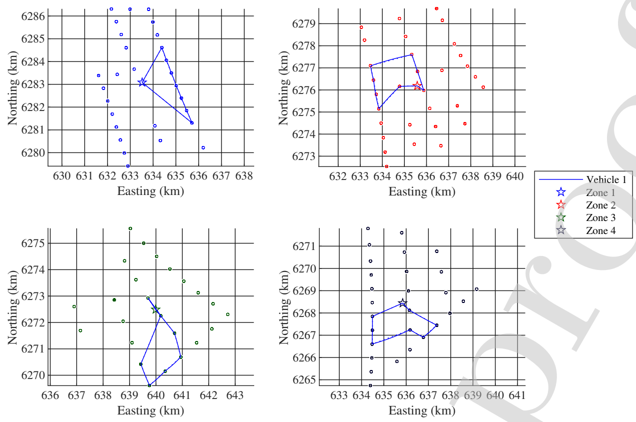


Figure 16: Solved sequences for the Wind Turbine Inspection Mission with a fleet containing 1 member. Feasible zones arranged as follows, Top Left: Zone 1, Top Right: Zone 2, Bottom Left: Zone 3, Bottom Right: Zone 4.

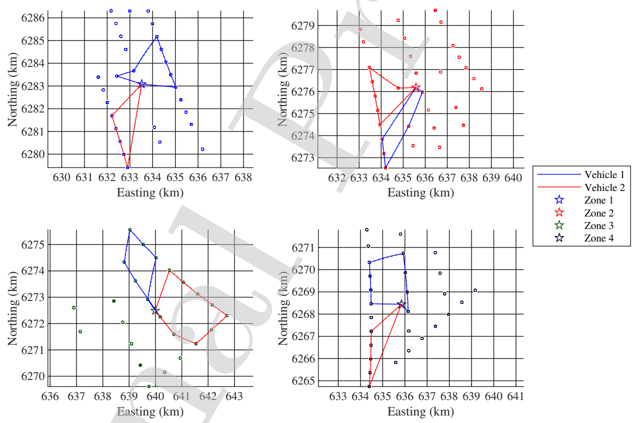


Figure 17: Solved sequences for the Wind Turbine Inspection Mission with a fleet containing 2 members. Feasible zones arranged as follows, Top Left: Zone 1, Top Right: Zone 2, Bottom Left: Zone 3, Bottom Right: Zone 4.

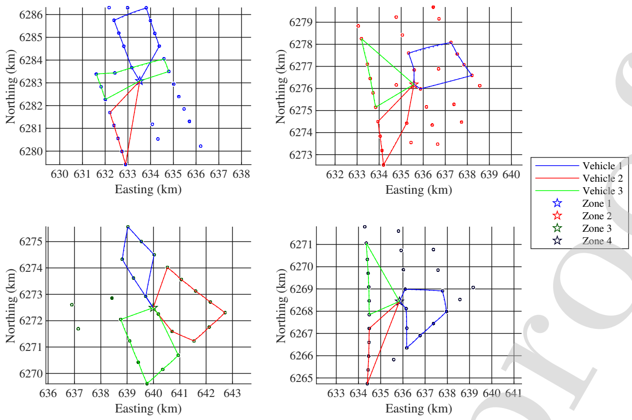


Figure 18: Solved sequences for the Wind Turbine Inspection Mission with a fleet containing 3 members. Feasible zones arranged as follows, Top Left: Zone 1, Top Right: Zone 2, Bottom Left: Zone 3, Bottom Right: Zone 4.

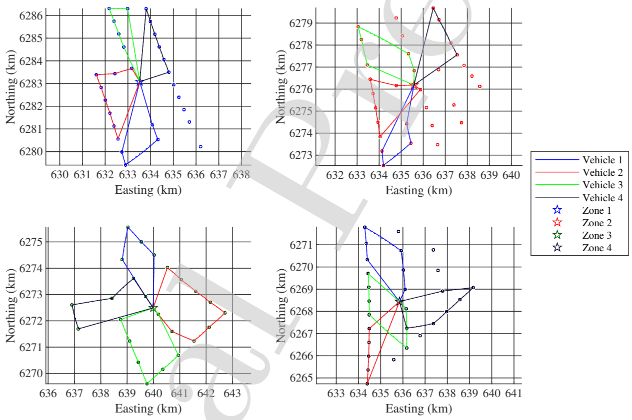


Figure 19: Solved sequences for the Wind Turbine Inspection Mission with a fleet containing 4 members. Feasible zones arranged as follows, Top Left: Zone 1, Top Right: Zone 2, Bottom Left: Zone 3, Bottom Right: Zone 4.

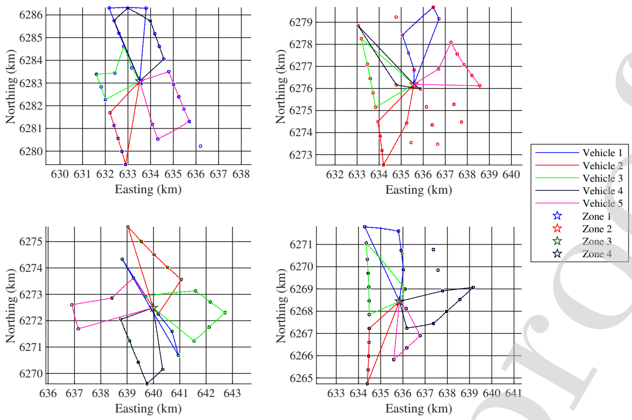


Figure 20: Solved sequences for the Wind Turbine Inspection Mission with a fleet containing 5 members. Feasible zones arranged as follows, Top Left: Zone 1, Top Right: Zone 2, Bottom Left: Zone 3, Bottom Right: Zone 4.

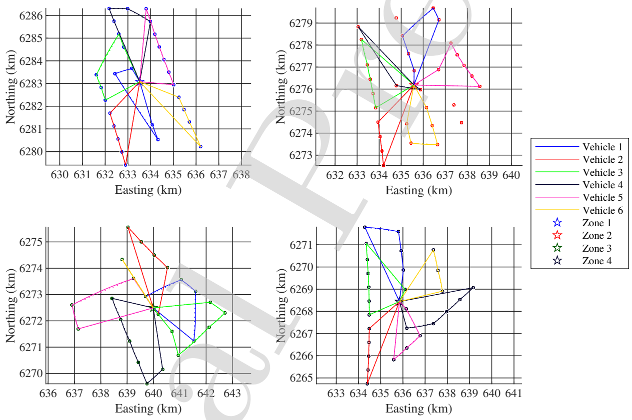


Figure 21: Solved sequences for the Wind Turbine Inspection Mission with a fleet containing 6 members. Feasible zones arranged as follows, Top Left: Zone 1, Top Right: Zone 2, Bottom Left: Zone 3, Bottom Right: Zone 4.

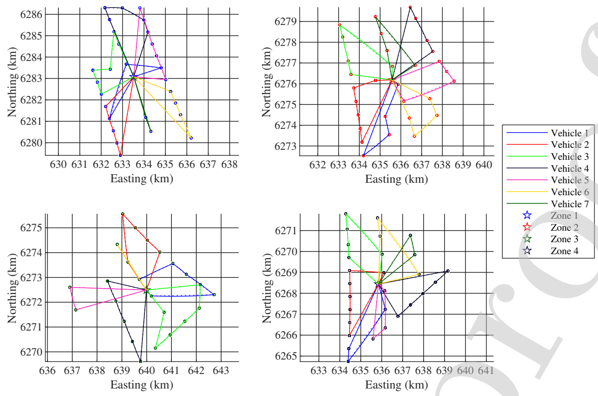


Figure 22: Solved sequences for the Wind Turbine Inspection Mission with a fleet containing 7 members. Feasible zones arranged as follows, Top Left: Zone 1, Top Right: Zone 2, Bottom Left: Zone 3, Bottom Right: Zone 4.

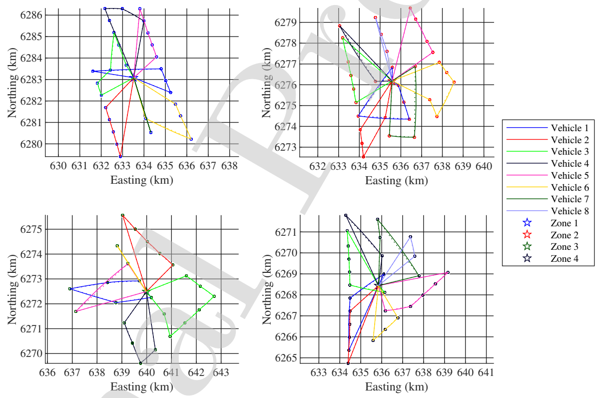


Figure 23: Solved sequences for the Wind Turbine Inspection Mission with a fleet containing 8 members. Feasible zones arranged as follows, Top Left: Zone 1, Top Right: Zone 2, Bottom Left: Zone 3, Bottom Right: Zone 4.

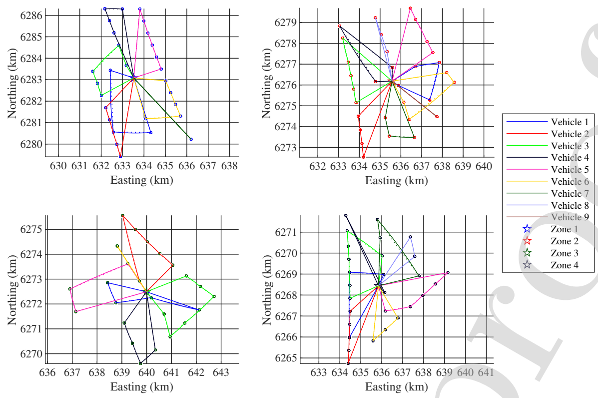


Figure 24: Solved sequences for the Wind Turbine Inspection Mission with a fleet containing 9 members. Feasible zones arranged as follows, Top Left: Zone 1, Top Right: Zone 2, Bottom Left: Zone 3, Bottom Right: Zone 4.

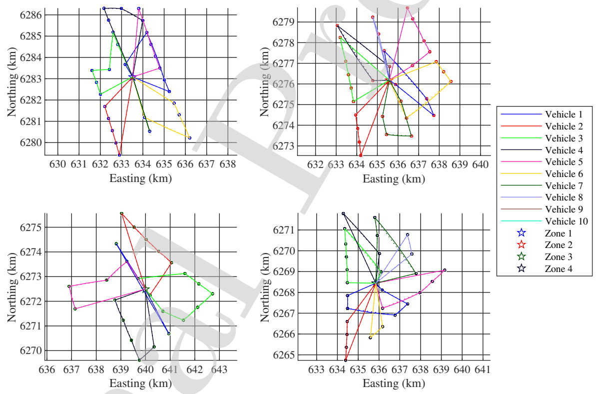


Figure 25: Solved sequences for the Wind Turbine Inspection Mission with a fleet containing 10 members. Feasible zones arranged as follows, Top Left: Zone 1, Top Right: Zone 2, Bottom Left: Zone 3, Bottom Right: Zone 4.

Fletcher Thompson
University of Tasmania (Australian Maritime College)
100 Newnham Drive
Newnham TAS 7248
Australia
Fletcher.Thompson@utas.edu.au

Prof. Karsten Berns, Prof. Maria Gini, Prof. Jun Ota
Editors-in-Chief
Robotics and Autonomous Systems

28-Apr-19

Dear Professors,

As the corresponding author my responsibility is to provide a statement of any conflicts of interest. Myself and my fellow authors can confirm that this article has not been published and is not under consideration for publication elsewhere. We have no conflicts of interest to disclose. All supporting funding bodies have been acknowledged and details have been provided with the attached application.

Thank you for your consideration!

Sincerely,

Fletcher Thompson, PhD Candidate
National Centre for Maritime Engineering and Hydrodynamics
University of Tasmania (Australian Maritime College)

Fletcher Thompson Biography

Fletcher Thompson is a marine engineering PhD candidate at the University of Tasmania's National Centre for Maritime Engineering and Hydrodynamics. His research focusses on automated planning, machine learning techniques including deep learning for forecasting and control, and autonomous marine systems.

Roberto Galeazzi Biography

Roberto Galeazzi is an associate professor with the Department of Electrical Engineering at the Technical University of Denmark. He is a member of IEEE and IFAC; he is the Chair of the IFAC Technical Committee on Marine Systems and he seats on the IEEE Technical Committee on Aerospace Control. His research interests include nonlinear and adaptive control and estimation, fault diagnosis and prognosis, fault-tolerant and reconfigurable control. His focus areas are safety critical systems, autonomous systems and wind turbines. He is the co-author of 7 ISI journal papers, 27 peer-reviewed conference papers and 2 book chapters. Further he is the coeditor of one IFAC conference proceedings.

Fletcher Thompson

Roberto Galeazzi