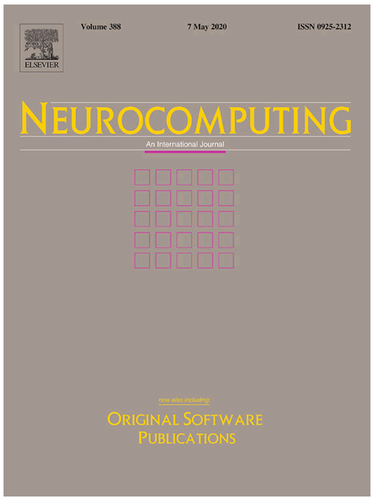


# Unsupervised learning based coordinated multi-task allocation for unmanned surface vehicles

Song Ma, Weihong Guo, Rui Song, Yuanchang Liu

PII: S0925-2312(20)31439-9
DOI: https://doi.org/10.1016/j.neucom.2020.09.031
Reference: NEUCOM 22814

To appear in: *Neurocomputing*

Received Date: 9 January 2020
Accepted Date: 17 September 2020

# Unsupervised learning based coordinated multi-task allocation for unmanned surface vehicles

Song Ma[a], Weihong Guo[b], Rui Song[c], Yuanchang Liu[a,*]

[a]*Department of Mechanical Engineering, University College London, Torrington Place, London WC1E 7JE, UK*
[b]*Department of Industrial and Systems Engineering, Rutgers University, New Jersey 08854, USA*
[c]*Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool L69 3BX, UK*

**Abstract**

In recent decades, unmanned surface vehicles (USVs) are attracting increasing attention due to their underlying capability in autonomously undertaking complex maritime tasks in constrained environments. However, the autonomy level of USVs is still limited, especially when being deployed to conduct multiple tasks simultaneously. This paper, therefore, aims to improve USVs autonomy level by investigating and developing an effective and efficient task management algorithm for multi-USV systems. To better deal with challenging requirements such as allocating vast tasks in cluttered environments, the task management has been de-composed into two submissions, i.e., task allocation and task execution. More specifically, unsupervised learning strategies have been used with an improved K-means algorithm proposed to first assign different tasks for a multi-USV system then a self-organising map (SOM) been implemented to deal with the task execution problem based upon the assigned tasks for each USV. Differing to other work, the communication problem that is crucial for USVs in a constrained environment has been specifically resolved by designing a new competition strategy for K-means algorithm. Key factors that will influence the communication capability in practical applications have been taken into account. A holistic task management architecture has been designed by integrating both the task allocation and task execution algorithms, and a number of simulations in both simulated and practical maritime environments have been carried out to validate the effectiveness of the proposed algorithms.

*Keywords:* `unmanned surface vehicle (USV), multi-task allocation, unsupervised learning, K-means clustering, self-organising map (SOM)`

*Corresponding author. Tel: +44 (0)20 7679 7062;
*Email address:* `yuanchang.liu@ucl.ac.uk` (Yuanchang Liu)

## 1. Introduction

There is an increasing trend in the development of advanced unmanned surface vehicles (USVs) in recent decades to support demanding and complex maritime missions, such as search and rescue in post-disaster scenarios, environmental monitoring and surveying in remote locations and coastal patrol in conflicting areas. It can be predicted that in the near future, with the advance in sensor and robotics technologies, USVs will be capable of executing missions under all kinds of harsh and remote circumstances and effectively, enhancing the mission efficiency as well as mitigating unnecessary casualties and human resource cost. However, compared with other types of unmanned platforms such as unmanned aerial vehicle (UAV), autonomous cars and autonomous underwater vehicle (AUV), the autonomy level of USVs is still relatively low. One solution to address such an issue is to deploy multiple USVs simultaneously and cooperatively. The benefits of using multi-USV systems include wide mission area, improved system robustness and increased fault-tolerant resilience.

The deployment of multi-USV systems can be used to deal with various missions, among which a multi-task mission or a single complicated mission that consists of multiple sub-tasks is of the most significance. One of the crucial requirements for deploying a multi-USV system to autonomously undertake a multi-task mission is to develop and employ an efficient task assignment strategy to properly allocate different tasks to different USVs such that the mission can be executed in an optimised and balanced way. In addition, while the USVs are undertaking missions, other critical aspects including the collision avoidance, the system reliability and the effectiveness of the communication within the system should be taken into account. Based upon these criteria, Liu and Bucknall [1] proposed a hierarchical structure for the control of multi-USV fleets. The structure encourages a cooperative operation among different USVs by employing a three-layer control scheme including the Task Management Layer, the Path Planning Layer and the Task Execution Layer. As elaborated in [2], the Task Management Layer is acting as the central decision making part mainly responsible for allocating tasks to each USV in conjunction with the vehicle's capability.

Presently, a number of studies have been carried out to investigate the multi-task allocation problem for multi-vehicle systems with the interest not only in marine applications but wider operation domains including air, sea and ground. Cunningham *et al.* [3] proposed an adaptive planning algorithm to operate a multi-UAV system in a cooperative way such that a large-scale sensor network can be formed. Shetty *et al.* [4] solved a multi-task allocation problem for UAVs to conduct reconnaissance missions. Missions are prioritised according their importance and the allocation problem is solved by using a heuristic optimisation algorithm. Kurdi *et al.* [5] proposed a bio-inspired multi-task allocation algorithm for a multi-UAV system. The algorithm is proven to have a linear running time making it suitable for on-line mission planning in complex scenarios such as the search and rescue mission. In terms of maritime applications, the majority work has been focused on multi-AUV applications. Deng *et al.* [6] investigated the problem of employing a multi-AUV system for cooperative underwater surveying. Supported by an underwater acoustic network, a two-layered planning and control algorithm has been designed for on-line adaptive mission allocation. Zhu *et al.* [7] proposed a multi-AUV task allocation algorithm using neural network. By considering the influences generated by ocean currents, the algorithm can effectively assign tasks to a group of AUVs in favour of the least energy consumption. Inspired by the work in AUV, algorithms have also been designed for multi-USV systems for practical maritime applications. For example, Raboin *et al.* [8] developed a model predictive control strategy for a multi-USV system for assets guarding. Scerri *et al.* [9] proposed to deploy multiple USVs with high-level decision-making capabilities to resolve demanding real-work challenges such as the flood disaster mitigation mission.

From mathematical modelling perspective, a multi-task allocation problem is synonymous with the Travelling Salesman Problem (TSP), which is a challenge of finding the shortest yet most efficient route to take given a list of specific destinations and can be mathematically modelled as a complete graph problem. Such a challenge has been widely investigated from the context of using exact algorithms such as the mathematical modelling [10] and dynamic programming [11, 12], and heuristic and approximation algorithms such as the LinâĂŞKernighan heuristics [13, 14], and the metaheuristics [15, 16]. However, it should be noted that TSP is also a NP-hard problem in combinatorial optimisation making the heuristic approach more favourable with regards to the computational efficiency. Especially, with the advance in machine learning in recent decades, it becomes increasingly appealing to adopt some classical machine learning algorithms to solve TSP. For example, self-organising map (SOM), an unsupervised learning algorithm with a 2-layer-neural-network structure which is first proposed by Kohoen [17, 18] for the visualisation of high-dimensional data, has been proven to be capable of solving TSP with the advantages of relative simplicity and promising performance [19, 20]. In essence, the SOM will form a neural network structure with a closed-loop circle topology and iteratively update the neural network to find an optimal solution in a competitive approach [21] (details of the SOM algorithm will be given in Section 4). Such a competitive mechanism can also be used to solve the multi-task allocation for multi-vehicle systems. For example, in [7], the effectiveness of SOM in solving the task allocation problem of multi-AUV has been successfully addressed, where each vehicle can be allocated with certain tasks in a time-varying ocean environment. However, [7] did not consider balancing the tasks, and thus may lead to unbalanced assignment where one vehicle of the system will not be assigned with any task, whereas other AUVs have to undertake multiple tasks that create a workload and a longer executive time.

In terms of the employment of the SOM on multiple USVs task allocation, Liu *et al.* [22] have successfully improved the SOM by integrating a coordination and prioritising scheme to better balance tasks among multi-USV systems. In addition, an important issue, i.e. the robustness of communication within the system has been well addressed, where the developed algorithm can intelligently ensure all the allocated tasks for a USV are well located within the communication range so that information can be transmitted in real-time. However, it is assumed that the communication is only constrained by a simple factor of distance while more complex aspects such as the signal transmission power, the signal noise power and the signal-to-noise ratio were ignored. Also, in [22] and [23], the communication is assumed to be between a USV and a base station with the location of the base station been randomly selected and not optimised. It is evident that a better selection of the base station location can increase the communication capability by minimising the impact caused by obstructions along the communication path.

This paper therefore endeavours to further research on the multi-task allocation for multi-USV systems by specifically addressing the above-mentioned communication issues. The underlying problem still considers a typical marine environment monitoring case, i.e. multiple USVs are deployed for water sampling collection. A real-time information transmission is required through a communication channel between a USV and a base station. The base station can be a mobile platform (for example a mother ship or a UAV) that will find its best location to ensure a robust and reliable communication. Note that the proposed solution can be further applied onto other vital practical applications such as AUVs launch and recovery from USVs, where a bilateral communication between two platforms is required. The task allocation problem in this paper is decomposed into two sub-missions as:

- *Task assignment*: multiple tasks are first grouped into different clusters according to various constraints such as vehicle's capabilities and the communication between USVs and their according base stations.
- *Task execution*: the assigned tasks are further planned to find an optimal task execution sequence for each USV. Path planning capability can be integrated at this stage to prevent any collisions.

In the context of task assignment (the number of tasks is normally larger than the number of vehicles), the clustering concept in machine learning is proven to be effective in fast grouping different tasks. Common clustering algorithms include K-means clustering [24], quality threshold clustering [25] and hierarchical clustering based on the greedy algorithm [26]. Especially, K-means has properties such as robustness and computational efficiency, which underpins the requirement of the task assignment problem mentioned above. Several studies have been conducted in this field, for example Cunningham *et al.* [3] proposed an adaptive path planning algorithm for a formation of UAVs to arrange a sensor network. In this work, the K-means clustering was employed at the initialisation stage for generating several sub-networks. Elango *et al.* [27] also used the K-means to solve the multi-task allocation for a multi-robot system. Due to the nature of the investigated problem, tasks in this work were allocated in a balanced way and the communication problem was not considered. Note that the conventional K-means clustering has the drawbacks such as the tendency to form clusters with similar size, difficulties in dealing with central-symmetric inputs and being apt to local optimisation, which cannot guarantee an expected result for all circumstances.

With the attempt to accelerate the capabilities of USVs to intelligently conduct complex maritime missions, main contributions of this paper can be summarised asïijŽ

- To facilitate an efficient execution of large-scale maritime missions, the task allocation has been de-composed into two submissions, i.e. the task assignment and the task execution with the former focusing on clustering missions into smaller groups by considering physical constraints while the latter mainly solving the refined on-line planning mission;
- The communication constraints between USVs and their base control stations have been explicitly resolved by using practical telecommunication models. Signal loss caused by obstacle obstructions are mainly considered such that missions

can be allocated to maximise the success rate in information communication to ensure important data collected by USVs will not be lost;

- Important case studies including using USVs for environment monitoring, coastal patrol and research and rescue have been investigated using the proposed algorithms. This can well demonstrate the capability of USVs in dealing with practical complex tasks and greatly promote the research into further ambitious maritime missions such as the Ocean of Things project that is well advocated by U.S. Defense Advanced Research Projects Agency (DARPA) [28].

More specifically, in this paper, a new task allocation algorithm has been proposed and designed using unsupervised learning strategy. Referring to the decomposition of task allocation problem (including *task assignment* and *task execution*), an improved K-means clustering algorithm is developed by considering the communication capability between a USV and its base station so that tasks can be better assigned to improve mission efficiency. In terms of task execution, the SOM algorithm has been adopted to effectively calculate an optimised execution sequence. Different sets of computer-based simulations have been undertaken to validate the effectiveness of the proposed algorithm.

The rest of paper is organised as follows. Section 2 introduces problem formulation and assumptions. Section 3 describes the proposed task assignment algorithm using improved weighted competitive K-means clustering method and Section 4 introduces the developed task execution algorithm for multi-USV systems based on self-organising map (SOM). Simulation results are given in Section 5 with detailed discussion on the effectiveness of the proposed algorithms. Section 6 concludes the paper and suggests future works.

## 2. Problem formulation and assumptions

Based upon the authors' previous work in [2], [22], which investigated the task allocation problem of USVs using self-organising map (SOM), a further in-depth study has been conducted by specifically considering the communication requirement within a multi-USV system. Specifically, the system consists of $k$ USVs, and each of them has a corresponding base station, such as a crewed ship, a UAV or any other types of highly mobile platforms that has compatible communication devices onboard. Each of the $k$ vehicles is supposed to visit several targets including its start point, and finally returns to the start point to form a closed path. During the navigation, it is required that USVs have to keep a constant communication with their corresponding base stations. In such a scenario, several underlying assumptions of this work are listed below:

- The distance between every two target points are long enough for the USV to complete a required manoeuvre.
- The obstacle data is captured and integrated into an environment map.
- The number of targets is greater than the number of USVs.

- The energy storage is adaptable to the mission requirement and sufficient to support missions.
- The environment changes such as the variations in currents and winds are not considered in this work.
- The node-to-node telecommunication path is assumed to be a line segment formed by connecting two nodes.

## 3. Task assignment using improved weighted competitive K-means clustering

The task assignment mainly focuses on generating a plan for a multi-USV system considering three key objectives as: 1) to assign specific tasks to each USV, 2) to calculate the distribution of each USV's mission zone and 3) to optimise each base station's location. To solve these problems, an improved K-means algorithm with a new competing strategy has been proposed in this work.

### *3.1. Mechanism of the conventional K-means clustering*

K-means clustering, also known as the Lloyd's algorithm, is a widely used unsupervised learning algorithm initially proposed by MacQueen [24] in 1967 for 'similarity grouping' and 'relevant classification'. K-means algorithm has played an active role in feature representations for computer vision and natural language processing [29].

K-means aims to solve the problem of partitioning $m$ observations or samples into $k$ clusters with each of observations consists of $n$ features. Such a problem is NP-hard [30] and K-means can rapidly converge into an optimum rather than providing an analytical solution. The fundamental of K-means is to minimise the feature difference within clusters, which is in general characterised as the Euclidean distances. Such a concept will be described mathematically in the following paragraph, along with a brief introduction of its implementation.

During the calculation, K-means (pseudo-code shown in Algorithm 1) employs three processes: (1) *Initialisation*, (2) *Assignment* and (3) *Updating*. The *Initialisation* process is implemented at the very beginning of the algorithm, and the other two processes form the algorithm's iterative part. Given $m$ observations ($\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(m)}$), K-means searches for the nearest local optimum as:

$$\underset{Cluster}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in Cluster_i} \left\| \mathbf{x} - \bar{Cluster_i} \right\| \quad (1)$$

where $Cluster$ denotes the sequence containing $k$ clusters ($Cluster_1, Cluster_2, \ldots, Cluster_k$) and $\bar{Cluster_i}$ is the mean of $Cluster_i$. In the first stage, $k$ centroids (or clusters) are randomly initialised. This process can be done either by the Forgy method (Algorighm 1 line 5) [31] or the Random Partition method (Algorighm 1 line 2) [32]. After the initialisation process, iterations start. In the *Assignment* process, each observation $\mathbf{x}^{(\mathbf{i})}$ ($i = 1, 2, \ldots, m$) will be assigned to the centroid that has the smallest Euclidean distance from the observation (Algorighm 1 line 17), such that every single observation will have an assigned centroid and the observations assigned to the same centroid form a cluster $Cluster_j$. When it comes to the *Updating* process, an existing centroid is updated to the mean value of the observations affiliating to its corresponding cluster (Algorighm 1 line 21). The *Assignment* process and *Updating* process will then run in iteration until reaching the maximum iteration number.

**Algorithm 1** K-Means Clustering

```
Require: number of the clusters($k$), maximum times of iterations($iter\_num$), set of $m$
    $n$-dimensional observation vectors ($X$), method of initialisation (either Forgy
    method or the random partition method)
 1: procedure INITIALISATION
 2:     if random partition method then
 3:         $Centroids \leftarrow$ generate a sequence including $k$ random $n$-dimensional vec-
    tors
 4:     end if
 5:     if Forgy method then
 6:
 7:         $Centroids \leftarrow$ generate a sequence by randomly selecting $k$ vectors from $X$
 8:     end if
 9:     $|D| \leftarrow k$                          ▷ Reserve space for sequence with $k$ entities
10:     $|Cluster| \leftarrow k$
11:     $assign \leftarrow 0$
12: end procedure
13: for $iter \leftarrow 1, 2, ..., iter\_num$ do
14:     procedure ASSIGNMENT
15:         for all $\mathbf{x} \in X$ do
16:             $D \leftarrow$ euclidean distances between $\mathbf{x}$ and all $\mathbf{c} \in Centroids$
17:             $assign \leftarrow$ index of $\min(D)$
18:             $Cluster_{assign} \leftarrow \mathbf{x} \cup Cluster_{assign}$
19:         end for
20:     end procedure
21:     procedure UPDATING
22:         for $j \leftarrow 1, 2, ..., k$ do
23:             $Centroids_j \leftarrow$ mean( $Cluster_j$ )
24:         end for
25:     end procedure
26: end for
27: return $Cluster, Centroids$
```

K-means clustering is an algorithm with several advantages, such as the fast convergence speed. Also, as K-means has a compact structure, the computational efficiency is relatively fast compared with other unsupervised learning algorithms [33]. However, the fast convergence feature makes K-means be apt to the local optimum problem, which consequently results in a fact that $k$ clusters returned by K-means turn to be even in dimension. Moreover, when visualising these results using the format of the Voronoi diagram, the clustering borders tend to cross the centre of the work space, which weakens the adaptability of K-means.

## *3.2. Weighted K-means*

The task assignment for a multi-vehicle system should be conducted in a cooperative and competitive way that each task should be assigned to a 'perfect candidate' (a vehicle within the system) by considering the availability of candidates as well as the global objective of the system. Such a strategy is similar to the *Assignment* process of K-means, where the assignment of a specific observation depends on the Euclidean distances from this observation to the $k$ current centroids. Similarly, in a multi-USV system, a target should be assigned to a base station, which is the closet to this target, in regardless of any obstacle or any intrinsic properties of the system. As a result, in an obstacle-free environment, it becomes viable to directly employ K-means algorithm for multi-task assignment for a multi-USV system.

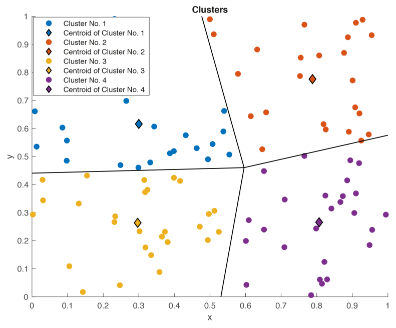
However, such an ideal environment does not exist in real world. For example, if one USV is not as capable as the others due to constraints in fuel, battery or communication, less tasks should be allocated for this USV. However, the conventional K-means tends to generate clusters with even distributions, which will approximately assign the same number of tasks to each USV. One of the possible improvements is to establish a new competing scheme to replace the original strategy that is based upon the Euclidean distances between the centroids and the observation (as shown in Algorithm 1). In order to increase the adaptability of K-means, different weights can be integrated to have a new competing scheme defined in Eq. 2 as:

$$comp = (1 - p) \times ||\mathbf{x} - Clus\bar{t}er_i|| \qquad (2)$$

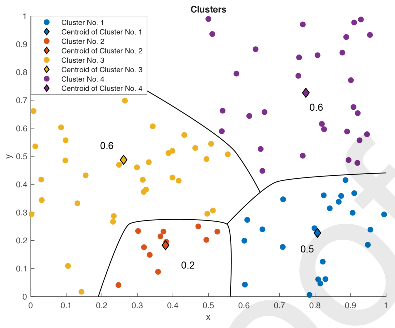
where *comp* is the competing measurement of the *Assignment* process of the algorithm; $p$ is the weight parameter with the value ranging from 0 (smallest) to 1 (largest) to indicate the size of the corresponding clusters, and the optimisation objective now can be redefined as:

$$\underset{Cluster}{\operatorname{argmin}} \sum_{i=1}^{k} \left[ (1 - p_i) \sum_{\mathbf{x} \in Cluster_i} ||x - Clus\bar{t}er_i|| \right] \qquad (3)$$

Fig. 1 provides a comparison result between the conventional K-means and the weighted K-means. In this example, the dimension of workspace ranges from 0 to 1 on x and y axes, and 100 targets are randomly created. According to the nature of the conventional K-means algorithm, the generated 4 clusters can roughly be regarded as 4 sectors with similar size as shown in Fig. 1a. In terms of the weighted K-means, four weight parameters of 0.5, 0.2, 0.6, 0.6 are given to 4 clusters labelled from No.1 to No.4. Note that cluster No. 1 should be larger than cluster No. 2; whereas, cluster No. 3 & 4 are supposed to be approximately identical and larger than cluster No. 1. As shown in Fig. 1b, the clustering result provided by the weighted K-means satisfies the requirement and based upon this, a signal-based competing scheme is further designed by integrating a practical signal-propagation model aiming to solve the communication problem during the task allocation process.



(a)



(b)

Figure 1: Example of clustering 100 target points. Together with the clustering result, the decision boundaries are plotted in both sub-figures with black lines and curves, the weight parameters are also marked. (**a**) Conventional K-means clustering for 100 targets. (**b**) Weighted K-means clustering for 100 targets.

### *3.2.1. Signal propagation mathematical model*

As mentioned earlier, the competitive assignment of multiple tasks is highly related to locations of stations as they are vital for retaining a good communication. K-means algorithm derives $k$ centroids with their X-coordinates and Y-coordinates determined as the mean value of the corresponding coordinates of the assigned targets. The centroid can be regarded as the ideal location for a base station and in this work, the weighted K-means has been further improved by replacing the weight parameter $p$ in Eq. 2 with a new parameter by considering a real-world signal propagation mathematical model as proposed in [34] and [35]. The signal-based competing model is able to measure the probability of successful communication between a base station and its corresponding USV, considering the extra path loss caused by terrain that is not transparent to the electromagnetic wave as:

$$P_r^{ij}\left(\Gamma \geq \gamma\right) = \exp\left(-\frac{\sigma_j^2 \gamma D_{ij}^{\alpha}}{C_{ij} P_i}\right) \tag{4}$$

where $P_r^{ij}$ is the probability subject to the signal-noise ratio (SNR) ($\Gamma$), $\sigma^2$ is the noise power, $D_{ij}$ is the distance between point $i$ and point $j$, $\alpha$ is the propagation loss factor, $C$ is the antenna gain and $P$ is the signal power. According to the general implementation of this signal propagation model, only when the current SNR level ($\Gamma$) is higher than a threshold ($\gamma$) the communication between the node $i$ and the node $j$ can be regarded as a definite success and the probability ($P_r^{ij}$) exists.

Note that within the signal propagation model in Eq. 4, the signal power ($P$) and the antenna gain ($C$) depend mostly on the communication capacity of base stations and can be regarded as constant values. As for the distance ($D_{ij}$) and the propagation loss factor ($\alpha$), they are used to calculate the path loss that will influence the telecommunication capability. In particular, $\alpha$ is essential for evaluating the path loss and affected by several environmental aspects.

### 3.2.2. Signal-based competing parameter

In typical maritime environments, physical obstacles, such as isles, have a significant impact on the telecommunication path loss. It is evident that the communication stability will be largely impaired if the path between a USV and its base station is blocked by many obstacles. In order to quantitatively reflect such an influence, the propagation loss factor ($\alpha$) in Eq. 4 is redesigned and calculated by considering obstacles' locations within an environment.

When there is no obstacle between a base station and a USV, the propagation loss factor ($\alpha$) is set to be a constant value of $\alpha_0$; whereas when obstacles exist, $\alpha$ will increase. Therefore, to determine the probability of a successful connection ($P_r^{ij}$) from a base station $i$ to a USV $j$, it is necessary to determine how much the telecommunication path is obstructed by obstacles. In this work, an obstacle obstruction algorithm (shown in Algorithm 2) has been proposed to calculate how much a path has been occupied obstacles, which subsequently will assist with the calculation of the propagation loss factor ($\alpha$) and the probability of a successful connection ($P_r^{ij}$).

More specifically, in Algorithm 2, it is assumed that the location of obstacle is known and the environment is formatted as a binary bitmap, in which each pixel has logical value of '1' or '0', standing for free and obstacle areas, respectively. As mentioned in Section 2, although the signal propagation is regarded as a line segment between two nodes, in some cases it is difficult to extract a straight line between two nodes in a grid map (for example, as shown in Fig. 2, there is no perfect straight line containing complete pixels between base station B and the USV). Therefore, a linear propagation path algorithm, the Bresenham's line algorithm [36], is employed to pick out all the pixels that lay on a propagation path (Line 3 in Algorithm 2), which subsequently assists with identifying the number of pixels sitting in obstacle areas (Line 4 - 8 in Algorithm 2). An obstacle-based parameter ($terrain_{arg}$) can finally be calculated (Line 9 in Algorithm 2). Note that the calculation of the obstacle-based parameter (($terrain_{arg}$)) can be in essence interpreted as:

$$terrain_{arg} = N_t \cdot W_{tp} \tag{5}$$

where $N_t$ is the number of obstacle pixels and $W_{tp}$ is the obstacle position weight, which is normalised with the minimum length of the bitmap, reflecting the relative distance from obstacles to a base station (higher value means farther distance). Such a consideration is mainly proposed to deal with the situation when multiple paths have been blocked by the same number of obstacles, where relative distances from the detected obstacle to the base station will be considered. For example, as shown in Fig. 2, sections of two paths (illustrated by two dashed red lines) occupied by obstacles are the same, which is 2 pixels. However, the relative positions between obstacles and the base stations are different, i.e. 1 pixel for base station B and 2 pixels for base station A. Based upon Eq. 5, station B will have a smaller $terrain_{arg}$.

Using the newly introduced obstacle-based parameter ($terrain_{arg}$), influences to the propagation loss factor ($\alpha$) caused by the size and position of obstacles can be modelled and calculated using a bounded and smooth sigmoud function as:

$$\alpha = \frac{A}{1 + e^{-k_{sa}(terrain_{arg} - \mu_a)}} + \alpha_0 \tag{6}$$

**Algorithm 2** Obstacle Obstruction Generation

```
Require: binary 2-dimensional array of the workspace with terrain information ($\mathbf{M}$),
    base station and target positions ($\mathbf{b}$ and $\mathbf{t}$)
 1: $Path \leftarrow \langle\,\rangle$
 2: $Terrain \leftarrow \langle\,\rangle$
 3: $Path \leftarrow$ Bresenham($\mathbf{b}, \mathbf{t}$)  ▷ Bresenham function returns a sequence of pixles from
    $\mathbf{b}$ to $\mathbf{t}$.
 4: for all $i \in indices(Path)$ do
 5:     if $\mathbf{M}_{Path_i}$ is obstacle then
 6:         $Terrain \leftarrow Terrain \,\|\, \langle i \rangle$        ▷ Append the index of the obstacle pixel.
 7:     end if
 8: end for
 9: $terrain_{arg} \leftarrow |Terrain| \cdot \frac{\sum Terrain}{\min(rows(\mathbf{M}), columns(\mathbf{M}))}$
10: return $terrain_{arg}$
```

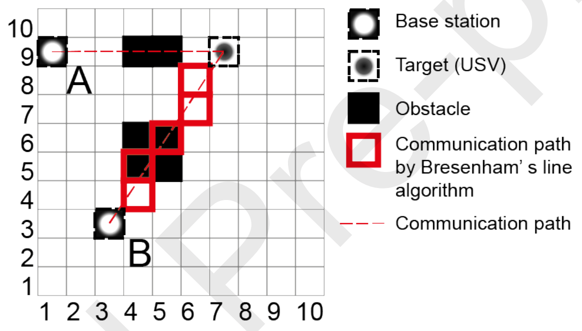


Figure 2: A demonstration of calculating the obstacle-based argument on a 10-by-10 bitmap. Each lattice stands for a single pixel, and the red bold-stroke pixels also show the effect of the Bresenham's line algorithm. In this illustration, the system needs to decide whether the target should be assigned to base station A or B.

where $k_{sa}$, $\mu_a$ are the parameters of the sigmoid function defining the shape of output curve, $A$ is the maximum value of the sigmoud function and $\alpha_0$ is the predefined propagation loss in obstacle-free environments. By setting $k_{sa} << 1$, the logistic function can sensibly reflect the change of $terrain_{arg}$ within a range. In addition, when the value of $terrain_{arg}$ stays beyond the range, the value of the logistic function can rapidly converge to either $\alpha_0$ or $(\alpha_0 + A)$. Eq. 6 will be subsequently integrated with Eq. 4 and Eq. 3 to develop an improved K-means algorithm fully considering the communication capability. The details of the improved K-means will be discussed in the next section.

## *3.3. Multi-task assignment using improved K-means*

Based upon the aforementioned competing strategy, an improved K-means algorithm considering the signal propagation capability has been developed for efficient multi-task allocation.

### 3.3.1. *Requirement and initialisation*

The pseudo-code of the improved algorithm is shown in Algorithm 3 with the majority of the algorithm's inputs being the same to the conventional K-means algorithm including the number of clusters ($k$), the maximum number of iteration ($iter\_num$) and the set of targets ($X$). There are also some other extra inputs for the improved K-means, such as the binary bitmap ($\mathbf{M}$) and the telecommunication parameters for calculating the integrated signal propagation model.

In the initial steps of the improved K-means, the centroids (or the locations of base stations) are initialised using either the Forgy method or the random partition method. Then during the competing procedure, the algorithm first determines the value of the propagation factor ($\alpha$) of each candidate centroids based on the pre-described method introduced in Section 3.2.2.

### 3.3.2. *Competitive Assignment*

Within the competitive assignment, referring to Eq. 2 the parameter $comp$ can be calculated as:

$$comp = D_{ij} \times \left(1 - P_{sr}^{ij}\right) \tag{7}$$

where $P_{sr}^{ij}$ denotes the signal connection measurement between node $i$ and $j$, and is calculated based upon the signal connection probability $P_r^{ij}$ from Eq. 4. The reason for such a further process of $P_r^{ij}$ is to assess the success of the signal propagation in a more deterministic way. As stated in [34], the signal connection between two nodes should be regarded as disconnected if $P_r^{ij}$ is smaller than a threshold. Therefore, in this paper $P_r^{ij}$ is processed by a logistic function to calculate $P_{sr}^{ij}$ (see line 24 in Algorithm 3) as:

$$P_{sr}^{ij} = \frac{1}{1 + e^{-k_{sp}\left(P_r^{ij} - \mu_p\right))}} \tag{8}$$

where $k_{sp}$ is the logistic growth rate controlling the steepness of the output curve (sigmoid curve), and the value of $k_{sp}$ should be $k_{sp} >> 1$ to minimise the blind zone of distinctness, $\mu_p$ is the threshold of the acceptable $P_{sr}^{ij}$.

In summary, $comp$ considers both the euclidean distance and the signal-based competing concept. During each competing assignment episode, a target point will be assigned to the centroid node with the smallest $comp$.

### 3.3.3. *Updating*

As for the updating procedure, it is largely similar to the conventional K-means with the to-be-updated centroid being the mean value of the assigned targets. However, considering both the above newly proposed competing strategy and some practical requirements, two vital aspects influence the robustness of the algorithm: (1) when the base station of a USV is a crewed ship or a fixed platform other than a UAV, the deployment of the base station could be unrealistic if the output localisation is within a terrain. To prevent a centroid being initialised within a terrain, during the initialisation stage, the Forgy method will be employed in this work as it only initialises centroids from the target positions; (2) during the updating process, according to the aforementioned

**Algorithm 3** Improved K-means Clustering

```
Require: number of USVs ($k$), maximum times of iterations($iter\_num$), task set ($X$), binary 2-dimensional array of the workspace with terrain information ($\mathbf{M}$), telecommunication parameters($\alpha_0, \sigma^2, \gamma, C, P$), parameters of sigmoid function for $\alpha$ ($A, k_{sa}, \mu_{sa}$), parameters of sigmoid function for $P_{sr}$ ($k_{sp}, \mu_{sp}$), method of initialisation (either Forgy method or the random partition method)
1: $D, P_r, P_{sr}, Comp, Cluster \leftarrow \langle\ \rangle$    ▷ Initialise utility sequences.
2: initialise $\mathbf{c} \in \mathbb{R}^2$, $assign \in \mathbb{N}$
3: $|Centroids| \leftarrow k$    ▷ Initialise centroids using either Forgy method or the random partition method
4: for $iter \leftarrow 1, 2, ..., iter\_num$ do
5:     for all $\mathbf{x} \in X$ do
6:         for $i \leftarrow 1, 2, ..., k$ do
7:             $terrain_{arg} \leftarrow$ Algorithm 2($\mathbf{M}, B, \mathbf{x}$)
8:             if $terrain_{arg} = 0$ then
9:                 $\alpha \leftarrow \alpha_0$
10:            else
11:                $\alpha \leftarrow A/\left(1 + \exp\left(-k_{sa} \times \left(terrain_{arg} - \mu_a\right)\right)\right) + \alpha_0$
12:            end if
13:            $D \leftarrow D \,\|\, \langle \|\mathbf{x} - Centroids_i\|_2 \rangle$    ▷ Append $\|\mathbf{x} - Centroids_i\|_2$ to $D$.
14:            $P_r \leftarrow P_r \,\|\, \left\langle \exp\left(-\frac{\sigma^2 \gamma D_i^\alpha}{C_i P}\right)\right\rangle$
15:            $P_{sr} \leftarrow P_{sr} \,\|\, \left\langle 1/\left(1 + \exp\left(-k_{sp}\left(P_r^i - \mu_p\right)\right)\right)\right\rangle$
16:            $comp = D_i\left(1 - P_{sr}^i\right)$    ▷ See Eq. 7
17:            $Comp \leftarrow Comp \,\|\, \langle comp \rangle$
18:        end for
19:        $assign \leftarrow$ index of min$(Comp)$
20:        $Cluster_{assign} \leftarrow \{\mathbf{x}\} \cup Cluster_{assign}$
21:        if $Cluster_i = \emptyset$ then
22:            reinitialise $Centroids$
23:            Break For Loop
24:        end if
25:    end for
26:    for $i \leftarrow 1, 2, ..., k$ do
27:        $\mathbf{c} \leftarrow$ mean( $Cluster_j$ )
28:        if $\mathbf{M_c}$ is NOT obstacle then
29:            $Centroids_i \leftarrow \mathbf{c}$
30:        else
31:            reinitialise $Centroids$
32:            Break For Loop
33:        end if
34:    end for
35: end for
36: return $Cluster, Centroids$
```

competing strategy, if a centroid is located in a terrain, it becomes difficult that this centroid can be assigned with any target because its propagation factor, $\alpha$, is higher than the other centroids. As a consequence, the centroid can hardly get out of this circumstance, which will eventually result in a problem of local minima. In order to solve this problem, all the centroids will be reinitialised using the Forgy method as stated in line 28 -30 in Algorithm 3.

## 4. Task execution for Multi-USV systems based on self-organising map (SOM)

The focus of task execution is to generate a reasonable execution sequence for each USV in the system to visit the assigned tasks. Such a problem, for each USV, is intrinsically a graph traversal problem. To entirely execute the tasks, USVs are supposed to visit all tasks at least once, so the graph traversal problem can be mathematically subdivided into the travelling salesman problem (TSP). In this work, based upon author's previous work in [2] and [22], an unsupervised learning approach, the self-organising map (SOM), is adopted.

### *4.1. Introduction to the self-organising map (SOM)*

Self organising map (SOM) is a two-layer artificial neural network firstly proposed by Teuvo Kohonen [17]. It is an unsupervised learning algorithm and is widely used for clustering problem with the feature of being capable of dimensionality reduction. SOM can efficiently abstract the relationships among high-dimension analytical data. SOM can also be employed to solve the 2-dimension problems like the TSP, which is an NP-hard problem. A notable highlight of the SOM employed in this work is that its output topology could be predefined, which distinguishes SOM from other similar self-organising networks like Neural Gas [37] and Growing Neural Gas Network [38], both of which generate adaptive output topology according to the input layer.

### *4.2. Implementation of SOM for USV Task Planning*

#### *4.2.1. Requirement and Initialisation*

When employing SOM network for TSP (the pseudocode is provided in Algorithm 4), the input is a set of the location vectors, $\mathbf{x} \in \mathbb{R}^2$, of the tasks ($X$), and the output neurons ($W$) is composed of neurons with a specific topology. For this case, a circular topology is used to conduct the network to derive a closed-loop path. The structure of network deployed in this work is illustrated in Fig. 3. The working space of this USV planning algorithm is a 2-D plane, so entities in both target set $X$ and sequence $W$ that forms the result execution order should be two-dimensional, consisting of x and y coordinates.

After initialisation, the following heuristic iterations including the winner neuron selection and the weight updating will be activated. Neurons in the output layer compete to be the winner neuron, which will be updated along with their neighbour neurons according to the chosen neighbourhood function.

**Algorithm 4** Self Organising Map for TSP

```
Require: targets set ($X \subseteq \mathbb{R}^2$), maximum times of iterations ($iter\_num$), multiplier for
    output layer size ($multi$), learning rate and its decay ($\lambda$, $\Delta_\lambda$), decay of neighbour-
    hood radius ($\Delta_G$)
 1: procedure Initialisation
 2:     $neuron\_num \leftarrow multi \cdot |X|$
 3:     $|W| \leftarrow neuron\_num$    ▷ Randomly initialise a set of output neurons, $W \subseteq \mathbb{R}^2$.
 4:     $R \leftarrow (W, E)$    ▷ Create a RING-topology graph with vertices $W$ and edges $E$.
 5:     $d \leftarrow 0$
 6:     $G \leftarrow 0.1 \cdot neuron\_num$    ▷ Initialise neighbourhood radius.
 7: end procedure
 8: for $iter \leftarrow 1, 2, ..., iter\_num$ do
 9:     procedure Winner Neuron Selection
10:         randomly select one $\mathbf{x} \in X$
11:         find the winner neuron ($W_{winner}$), with Eqs. 9 and 10
12:     end procedure
13:     procedure Neighbourhood updating
14:         for all $\mathbf{w} \in W$ do
15:             $d \leftarrow \min(|path\,(\mathbf{w} \rightsquigarrow W_{winner})\,|)$
16:             $\mathbf{w} \leftarrow \mathbf{w} + \cdot h(d, G) \cdot (\mathbf{x} - \mathbf{w})$
17:         end for
18:     end procedure
19:     $G \leftarrow G \cdot \Delta_G$
20:     $\lambda \leftarrow \lambda \cdot \Delta_\lambda$
21: end for
22: return $W$
```

### *4.2.2. Winner Selection and Neighbourhood Function*

Winner neuron selection is the beginning procedure of the iteration process. Firstly, an input target $\mathbf{x} \in \mathbb{R}^2$ is randomly selected from $X$. Then, its euclidean distances from all existing neurons in $W$ is calculated with Eq. 9 as:

$$Dist_j = \|\mathbf{x} - W_j\| \tag{9}$$

where the weight or coordinate of the specific $W_j$ corresponding to the minimum Euclidean distance, $dist_{ij}$, is determined as the winner neuron, as:

$$W_{winner} = \underset{W_j \in W}{\operatorname{argmin}} \left(Dist_j\right) \tag{10}$$

### *4.2.3. Updating*

After the selection of a winner neuron, the neighbourhood function is employed to process the updating. There are two major types of neighbourhood function for SOM including the Gaussian neighbourhood function and the Bubble neighbourhood

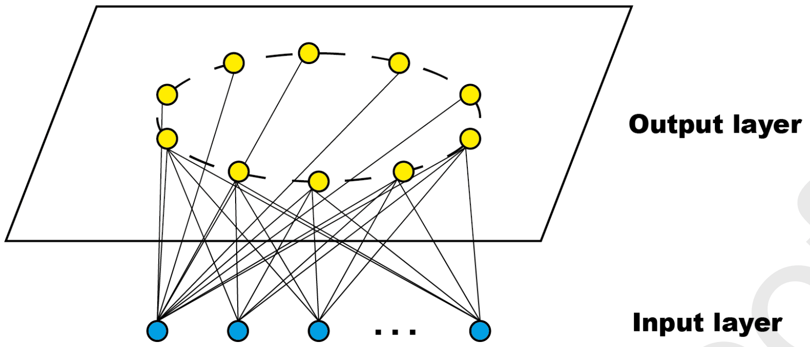


Figure 3: Structure of the deployed SOM network with a circular-topology output layer and an input layer.

function [39], which are presented in Eq. 11 and Eq. 12, respectively:

$$h(d,\, G) = \exp\left(\frac{-d^2}{G^2}\right) \tag{11}$$

and

$$h(d,\, G) = \begin{cases} 0 & d > G \\ 1 & d \leq G \end{cases} \tag{12}$$

where $d$ is the topological lattice distance between the updated neuron and the winner neuron, and $G$, the gain parameter could be regarded as the radius of the neighbourhood. Mathematically, the Gaussian neighbourhood function, Eq. 11, has a smoother kernel-shape distribution, compared with the Bubble neighbourhood function. The Bubble neighbourhood function is, in a sense, an approximation of the Gaussian neighbourhood function with less demand for computational resources. In this work, the Gaussian neighbourhood function is used.

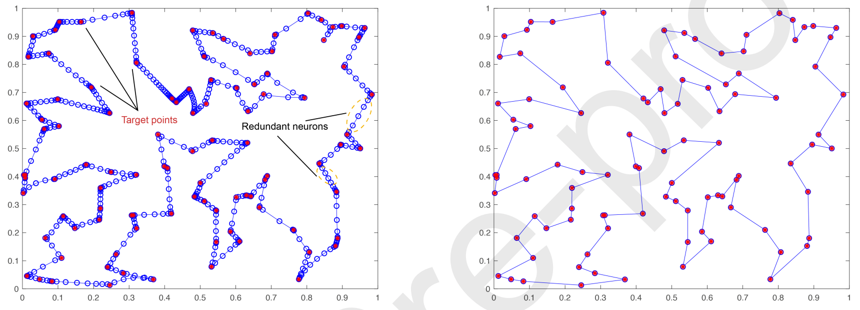
Weightings of the output layer neurons will be updated as shown in Line 16 in Algorithm 4, where $\lambda$ is the learning rate with a decaying rate of $\Delta_\lambda$. Similarly, the gain parameter $G$ also decays in iterations with a decaying factor of $\Delta_G$. The tuning of these four parameters in SOM ($\lambda$, $\Delta_\lambda$, $G$ and $\Delta_G$) has been well studied in [40], [41] and [20], which include a number of invaluable empirical parameter tuning works. According to the provided recommendations in these works, four aforementioned parameters configured as shown in Table 1. It is noteworthy that these parameters are configured based on the assumption that the number of input targets for the SOM is no more than 1000 (namely maximum 1000 missions are required to be executed by USVs), and parameters should be reconfigured if the demand for input targets is different.

### *4.2.4. Redundant neuron deletion*

SOM returns the order which a USV will follow to visit their assigned tasks. Such a task execution order is extracted from the output neurons of SOM with the neurons keeping their initialised topology, a ring topology. In regard to the updating mechanism of SOM, after convergence, each task should coincide with a corresponding neuron. However, as the number of output neurons of SOM is larger than the number of the

Table 1: Configuration of the parameters of SOM for TSP. $\lambda$ and $\Delta_\lambda$ are learning rate and its decay; $G$ and $\Delta_G$ are neighbourhood radius and its decay; $neuron_num$ indicates the number of neurons of the output layer of the SOM.

| SOM parameters | Value |
|---|---|
| $\lambda$ | 0.7 |
| $\Delta_\lambda$ | 0.999 |
| $G$ | 1/10 of the output neuron number |
| $\Delta_G$ | 0.995 |



(a) (b)

Figure 4: Redundant neuron deletion in a regularised workspace. The red stars are the target points and the blue circles are the output layer neurons of the SOM. (**a**) Output of the SOM with redundant neurons. (**b**) Output of the SOM with redundant neurons deleted.

tasks, there will be redundant neurons in the output layer, as illustrated in Fig.4a. To address such an issue, the neuron deletion method proposed in [22] has been used and the refined result is presented in 4b.

## 5. Simulation results and discussions

### 5.1. Simulations set-up

In this section, the proposed task assigned algorithm using the improved K-means and the task execution algorithm based upon SOM are validated in computer-based simulations. In order to carry out the simulations in a systematical way, two algorithms have been interconnected using a system structure illustrated in Fig. 5, which consists of four primary layers:

- **Green layer**: The requirement & input layer.
- **Blue layer**: The global task assignment where the improved algorithms developed in section 3 are executed to assign tasks to each USV-base-station pair and also optimise the location of the base stations.

- **Orange layer**: The task execution and planning layer where the task allocation and the collision avoidance processes are carried out.
- **Purple layer** After all above is completed, the reference paths and optimised localisation of the base stations will be spread to every pair within the system from the central control.

The algorithms have been validated on a 64-bit laptop with a 2.2 GHz Intel® Core$^{TM}$ i7-8750H processor. The source codes are developed and tested on MATLAB R2019a. Two different sets of simulations are carried out as: 1) a comparison between the conventional and the improved algorithms is undertaken in a simulated environment; 2) simulations are carried out in two practical maritime environments validating algorithms' capability in dealing with two important tasks, i.e. the coastal patrol and the research and rescue missions.

Note that to test algorithms' capability in dealing with challenge missions, which is normally the case in maritime environments, the mission points have been randomly configured with some extreme cases existed, where the conventional K-means algorithm is incapable of producing a reasonable clustering result by considering communication constraints. Also, different to other works ([2], [22], [23]) where only small-scale maps are used, high-resolution maps are used in this paper with the results showing that the proposed algorithm is able to efficiently to work.

### *5.2. Comparison results between the conventional and the improved algorithms*

In this section, tests will be carried out in a $500 \times 500$ pixels map. 70 target points together with two isle-shaped obstacles are randomly placed. The distribution of the target points is shown in Fig. 6. It can be seen that all the targets stay in obstacle-free areas. However, in order to test the performance of the algorithm in some extreme cases, several target points are configured to be laid in bay areas as circled out by blue dash lines. In this tests, 4 USVs are used to cooperatively execute 70 tasks, and each USV needs to maintain a communication link with a base station for transmitting information. It is assumed that 4 USVs are identical in terms of vehicle's capability and are equipped with sufficient energy to support missions.

#### *5.2.1. Parameter configurations*

Precise tuning of two logistic functions (Eq. 6 and Eq. 8) in the improved K-means algorithm (Line 11 and 15 in Algorithm 3) is important for solving the problem in the map above as they determine the telecommunication capabilities between USVs and their base stations. As highlighted previously, the telecommunication model is adopted from [34] and their model parameters are first used as the basis in this paper to describe the communication capacity in obstacle-free environments. Then the sigmoid function is used to evaluate how the communication will be affected by obstacles.

More specifically, in [34], the propagation factor $\alpha$ has the value of 3 to reflect the communication capability unaffected by any obstacles, therefore $\alpha_0$ in Eq. 6 is 3. Then by setting $A_{sa} = 3$, $\alpha$ will vary within a range of [3, 6], where the higher the value is, the more loss the communication propagation. Note that the maximum value of $\alpha$ does not influence the final results in a drastic way as long as there is an evident difference

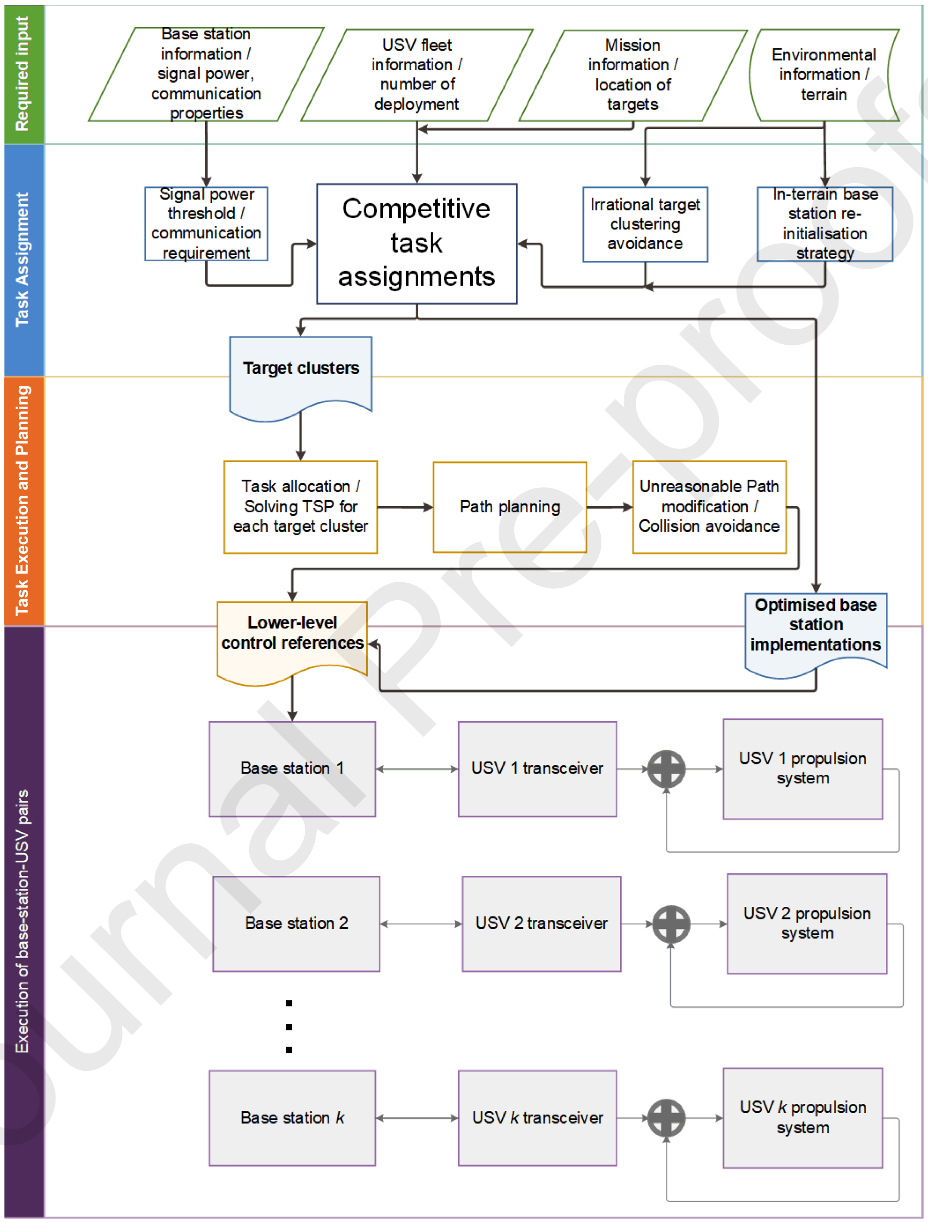

Required input
Task Assignment
Task Execution and Planning
Execution of base-station-USV pairs
Base station information / signal power, communication properties
USV fleet information / number of deployment
Mission information / location of targets
Environmental information / terrain
Signal power threshold / communication requirement
Competitive task assignments
Irrational target clustering avoidance
In-terrain base station re-initialisation strategy
Target clusters
Task allocation / Solving TSP for each target cluster
Path planning
Unreasonable Path modification / Collision avoidance
Lower-level control references
Optimised base station implementations
Base station 1
USV 1 transceiver
USV 1 propulsion system
Base station 2
USV 2 transceiver
USV 2 propulsion system
Base station k
USV k transceiver
USV k propulsion system

Figure 5: System integration structure.

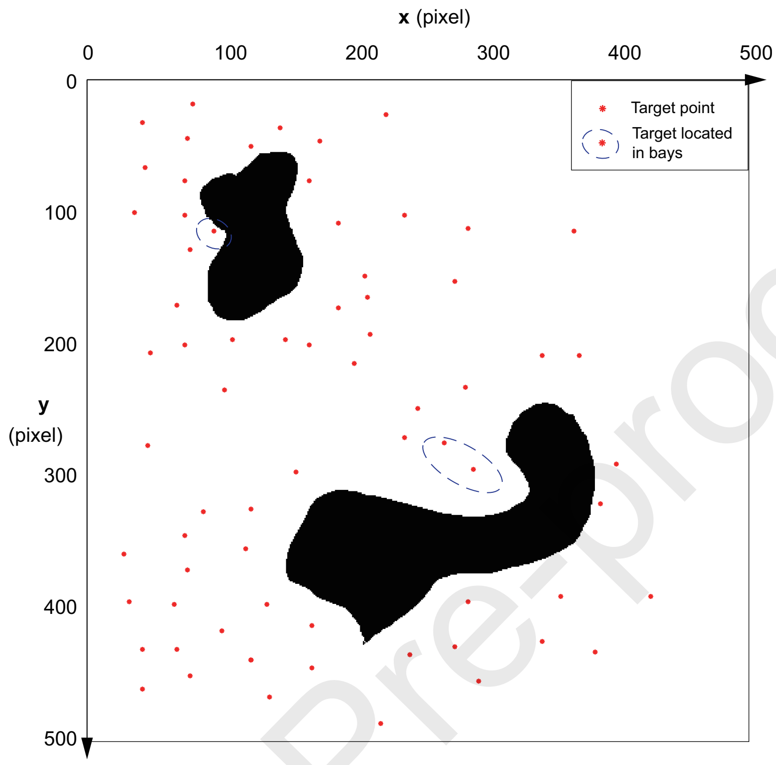
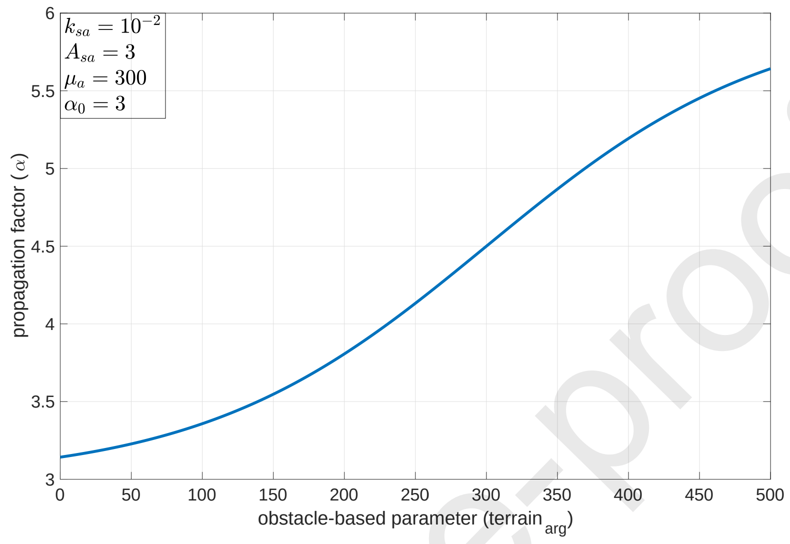


Figure 6: 500 × 500 workspace with 70 target points marked with red stars.

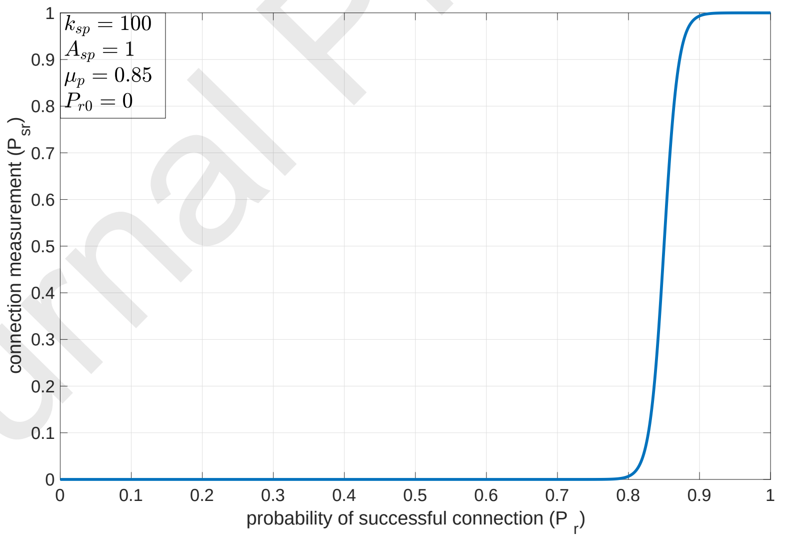
between the maximum and minimal values of $\alpha$. In addition, a gradual variation of $\alpha$ is preferred to reflect a smooth propagation loss change affected by obstacles. Therefore, $k_{sa}$ (slope) and $\mu_a$ (midpoint) are assigned with value of $10^{-2}$ and 300, respectively, with the final relationship revealing how $\alpha$ is affected by obstacles shown in Fig. 7a.

For the second sigmoid function (Eq. 8), it is important to tune a proper value for $\mu_p$ as the communication threshold so that any value below this threshold will be regarded as an unsuccessful communication and should be discarded. Improper settings of $\mu_p$ would lead to an unsatisfactory clustering result. For example, as shown in Fig. 8, where $\mu_p = 0.5$, it is clear that two mission points highlighted in blue should be allocated to the base station in red and there is only one mission point assigned to the base station in purple which is evidently unacceptable. On the contrary, if the value of $\mu_p$ is too high, it becomes difficult to identify successful communication as most of channels will be regarded as invalid. By undertaking a set of trial-and-error tuning processes of $\mu_p$, it is found that $\mu_p = 0,85$ would give the most balanced clustering results with $k_{sp}$ assigned with the value of 100 to better distinguish successful and unsuccessful communication. The final output of Eq. 8 is shown in Fig. 7b.

The discussed parameter settings are summarised in Table 2. It is noteworthy that all these parameters are tuned only to accommodate the simulations in this section, and



(a)



(b)

Figure 7: Two sigmoid functions used in the algorithm **(a)** is the sigmoid function mapping from the obstacle-based parameter to propagation factor, and **(b)** is the sigmoid function mapping from the probability of successful connection to the signal connection measurement.

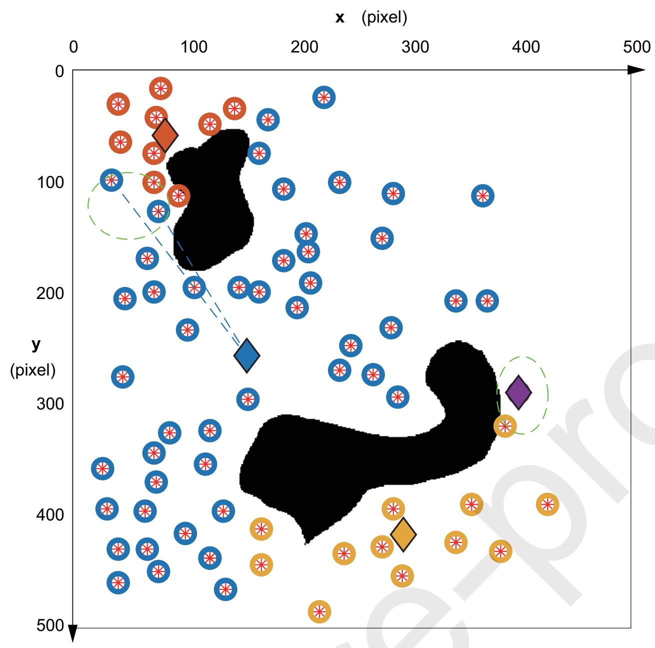


Figure 8: The result of improved K-means when $\mu_p = 0.5$

520 in the cases of practical implementations, these parameters should be set based on the specific requirements.

Table 2: The tuned parameters of the system (Logistic function)

| **Parameters** | **Value** |
|---|---|
| $A_{sa}$ | 3 |
| $k_{sa}$ | $10^{-2}$ |
| $\mu_a$ | 300 |
| $\alpha_0$ | 3 |
| $k_{sp}$ | 100 |
| $A_{sp}$ | 1 |
| $\mu_p$ | 0.85 |
| $P_{r0}$ | 0 |

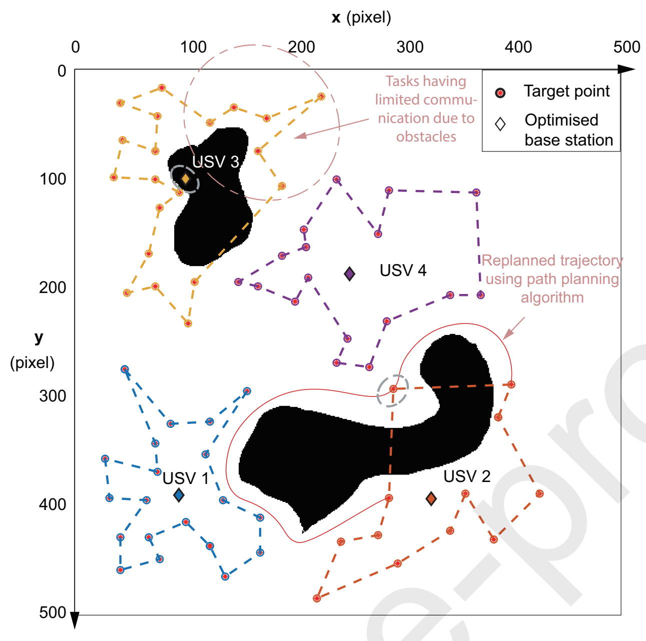
### 5.2.2. *Simulation results*

By using the tuned parameters in Table 2 the improved algorithm will be tested against the conventional K-means algorithm. The results returned by the conventional and the improved K-means algorithms are shown in Fig. 9. Note that there is no difference in USV numberings within two sub-figures. It can been seen that both algorithms are able to successfully generate four clusters of tasks (clusters are distinguished by different colours) for 4 USVs. Assigned tasks are further processed by SOM algorithm to have an optimised task execution sequence, which is connected by dash lines in different colours in Fig. 9. It also should be noted that the straight line connecting two tasks should be considered as an ideal trajectory for a USV to track if the line does not violate any obstacle area. With the existence of obstacles, the ideal path (the straight line) needs to be accordingly modified to eliminate any collision risks. For example, as highlighted for USV 2 in Fig. 9a, two dash line already cross an obstacle and a possible replanned trajectory is suggested in Fig. 9a. Note that such a trajectory can be calculated using typical path planning algorithms such as A*, rapidly-exploring random tree (RRT) and fast marching method, which will not be discussed in this paper. It is also worth emphasising that apart from successfully assigning tasks for 4 USVs, locations of 4 base stations can be provided as well, which are represented as diamond marks in Fig. 9.

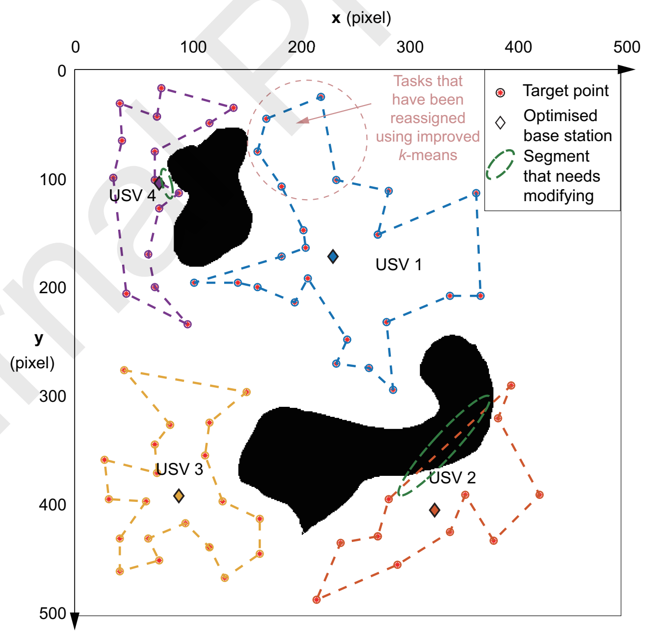
By further analysing the results in Fig. 9, superiority provided by the improved K-means is evident. First, the conventional K-means cannot prevent the base stations from falling into obstacles. For example, in Fig. 9a, it is clear that the suggested location for the base station for USV 3 is located in an obstacle area, which is unacceptable if the base station is a vessel. Even though in the cases where a UAV, which is able to fly over the obstacle area, is used as the base station, the terrain obstacle still has a high possibility to significantly influence the communication connectivity. A better solution to such an issue is to place base stations in obstacle-free areas. As presented in Fig. 9b, the improved K-means can achieve this by keeping all base stations away from obstacles.

Second, as shown in Fig. 9a, for USV 3, the communication between the circled out tasks and the base station will be largely constrained as the communication links cross most part of the obstacle. The main reason for such a disadvantage is that the competing process for the conventional K-means only considers the Euclidean distance, and tasks are grouped in terms of the shortest distance to the base station. As for the improved K-means, by integrating the weighted competing process based upon the signal propagation model, such an issue can be well addressed. As shown in Fig. 9b, by assigning the highlighted tasks in Fig. 9a to USV 1 (the same to USV 4 in Fig. 9a), tasks have been rearranged in a way that situations where communication channels violate obstacles areas are minimised.

Additionally, it is worth noticing that USV 1 in Fig. 9a and USV 3 in Fig.9b are assigned with the same number of target points, but the task planning result is different. This is mainly due to the inherent feature of SOM, i.e. SOM is prone to converge to a nearest local optimum, which may lead to a slight different result. However, the influence of such a low repeatability is minimal, especially when regarding the increase of computational efficiency.



(a)



(b)

Figure 9: Task assignment and planning results in a 500 × 500 workspace with obstacles, where (**a**) is the result returned by the system integrated with the conventional K-means algorithm and (**b**) is the result using the improved K-means algorithm.

### 5.3. Simulations in practical environments

In this section, simulaions have been carried out in two pratical environments to further validate algorithm's capability. First, a coastal patrol mission is simulated in Southampton port estuary, UK, with 6 USVs been deployed to visit a large number of afloating checkpoints along the coastal line. The distribution of checkpoints is shown in Fig. 10 and the simulation environment is a 3460 × 4800 pixels map. To facilate the running of algorithms, the simulation map is further converted into a binary map represented in Fig. 11. The underlying system configuration is still the same to previous simulations, where each USV needs to maintain a constant and reliable communication with a base station, and the location of the base station will be optimised within the task allocation process.

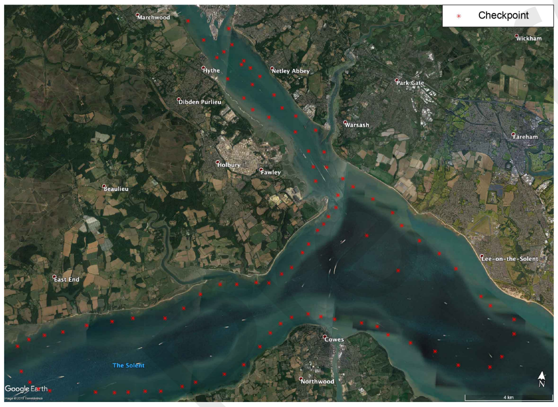


Figure 10: The 3460 × 4800 map of Southampton port estuary, with patrol checkpoints marked with red stars.

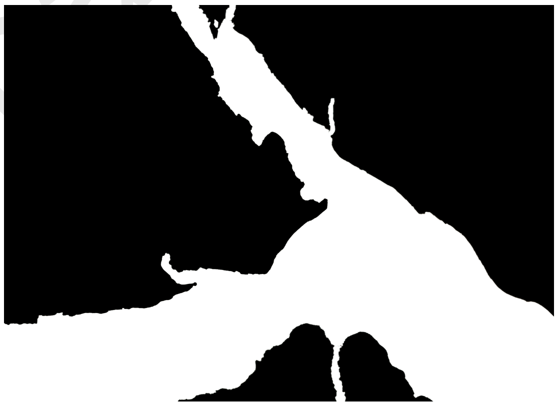


Figure 11: The binary format of the 3460 × 4800 map of Southampton port estuary.

Using the binarised geographical information, the proposed algorithms can successfully generate an optimised task allocation result for 6 USVs as shown in Fig. 12. Also, the location of each base station can be optimised, which is shown as the diamond mark. It can be see that all checkpoints are visited by 6 USVs, which forms 6 circular patrol routes for monitoring the coastal areas. It also can be observed that because the coastal patrol mission is undertaken in an area with limited obstacles, 90 tasks are allocated in a relative balanced way. As shown in Fig. 13a, about the same number of tasks are given to USV 1, 3 and 6; whereas, USV 2 and 5 are executing 11 and 10 tasks, respectively. Such an balanced allocation result can be further elaborated when considering the total distance covered by each USV. As shown in Fig. 13b, apart from USV 5 all the other USVs travel relatively the same distance to visit each assigned tasks. This computing process takes about 11.4 seconds to generate the results, which is efficient enough for a real-time operation. Note that there is a segment marked by dashed red circles in Fig. 12 indicating that proper modifications are required. This can be well addressed by integrating some path planning algorithms to generate collision-free paths. Works in [2] and [22] can be well referred to show how path planning algorithms can be well combined with the algorithms proposed in this paper. Instead of applying the path planning algorithms for each pair of nodes, a collision check algorithm should be used and routes will only be re-planned when there is a collisions risk. Such a strategy can well increase the computational efficiency of the algorithms.

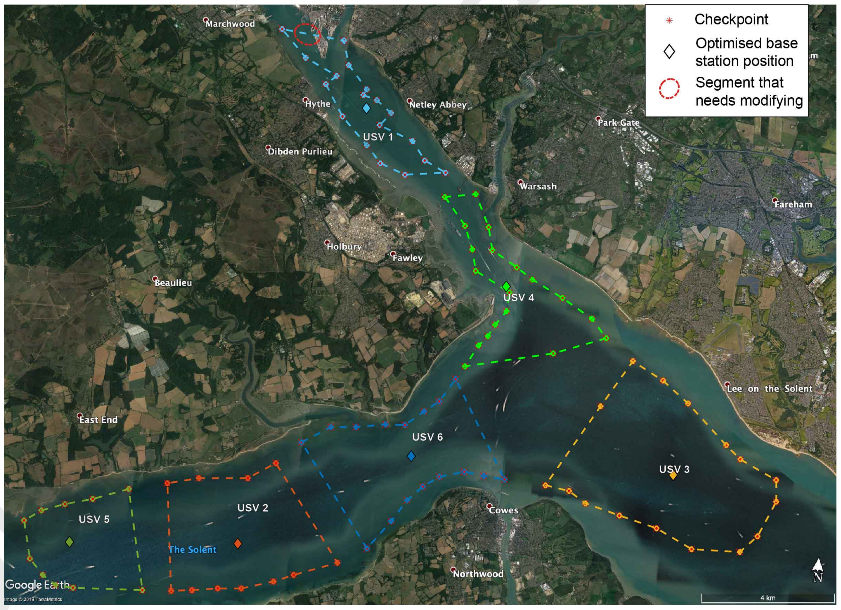
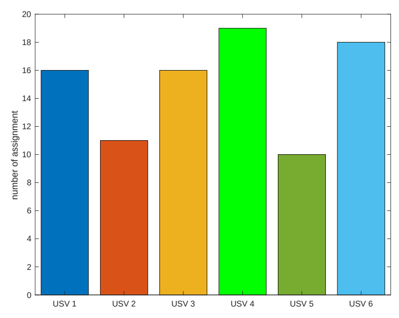
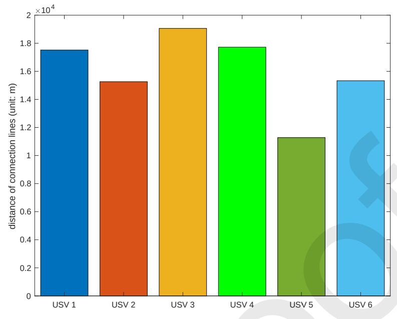


Figure 12: Result of the first practical environment simulation, at the estuary of Southampton port. 90 checkpoints are assigned to 6 USVs and the order to visit them for each USV has been provided. The optimised positions of the base stations have been mark with the diamonds, segment that needs modifying to achieve collision-free status has also been marked out with red circle.

Apart from coastal patrol mission, another important utilisation for multi-USV system is search and rescue and in the second practical simulation, such a scenario is considered for searching debris of the missing flight, Malaysia Airlines Flight 370. The



(a)



(b)

Figure 13: Statistical data of the patrol simulation result, reflecting the task amount assigned to each USV, (a) is the number of checkpoints assigned to each USV, and (b) is the distances (without modification) of each USV to travel.

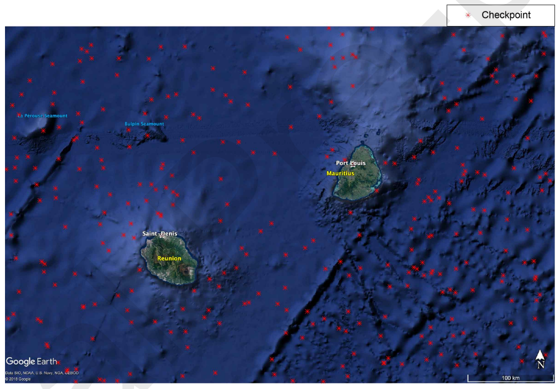


Figure 14: The workspace and checkpoints of the searching mission near Réunion, the checkpoints are marked with red stars.

simulation area is selected near French Réunion, where autonomous underwater vehicle *Bluefin-21* was used by US Navy to search for debris evidence. The simulation map is shown in Fig. 14 with three hundred checkpoints representing possible debris locations been randomly picked from the adjacent sea area. The size of the map is $3113 \times 4800$ pixels and further converted into a binary map as shown in Fig. 15.

Seven USVs are assumed be deployed with their corresponding base stations to execute this searching mission. Because of the large scale of the area, the searching mission will last for a long period and will be completely autonomous. The base station will be attached to floating chambers to ensure the stability and robustness. The whole multi-USV system is expected to be equipped with multiple power sources to support such a long endurance mission, and a possible power solution is to use a hybrid strategy consisting of electrical batteries, solar powered panels and other energy converters which can harvest energy from the environment.

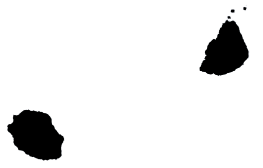


Figure 15: The binary bitmap of the workspace in Fig. 14

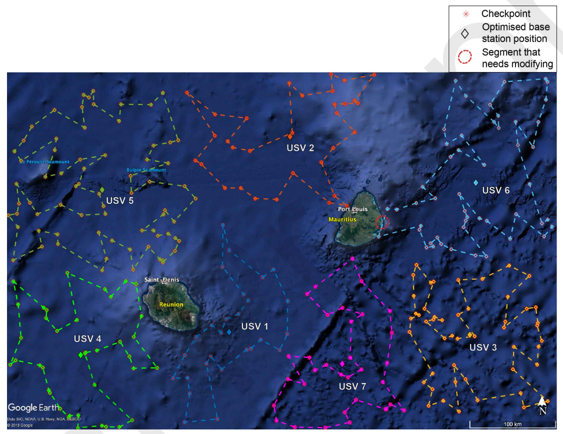
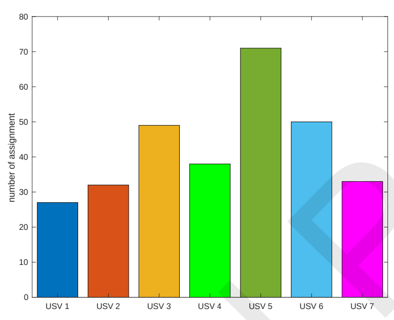
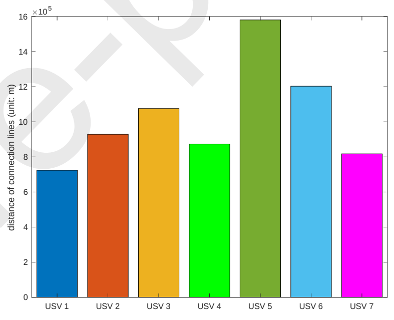


Figure 16: Result of the second practical environment simulation, at the adjacent sea area close to the French Réunion, 300 checkpoints are assigned to 7 USVs and the order to visit them for each USV has been provided. The optimised positions of the base stations have been mark with the diamonds, segment that needs modifying to achieve collision-free status has also been marked out with a red dashed circle.

The simulation result is shown in Fig. 16 with its statistical results shown in Fig. 17. In Fig. 16, it can be observed that all the 300 checkpoints can be successfully assigned and visited by the USVs. None of the 7 USVs needs to detour around an isle to reach its assigned target point. In addition, the returned base station positions are reasonable such that the communication between each USV and its base station can be largely retained. However, it also should note that there is one segment that has been marked with a red dashed circle crossing the obstacle. and such an issue can be rectified by using path planning algorithms, which are not discussed in this paper.



(a)



(b)

Figure 17: Statistical data of the searching simulation result reflecting the task amount assigned to each USV, (a) is the number of checkpoints assigned to each USV, and (b) is the distances (without modification) of each USV to travel.

### 5.4. Discussions

To further evaluate the performance of the proposed algorithms, discussions on algorithm's convergence capability as well as its computational time have been provided. The convergence capability is mainly discussed from the perspective of mitigating local minimum of K-means and the computational time is analysed by evaluating the running time for four different simulations.

#### 5.4.1. Mitigating local optimum of the improved K-means

Similar to conventional K-means, the proposed improved K-means algorithm is also possible to suffer from local minimum problem. In order to avoid such poor-quality solutions, common methods (the initialisation and restart techniques proposed in [42, 43]) have been applied in this paper including: 1) iterating the algorithm with random initialisation and 2) selecting the high-quality result from the batch of the outputs. During iterations, the quality of clustering outputs is tightly assessed by defining a scoring function to evaluate the quality of results. The scoring function is defined by following the objective function in conventional K-means as:

$$Q = \sum_{\mathbf{x} \in X} \min(Comp_{\mathbf{x}}) \tag{13}$$

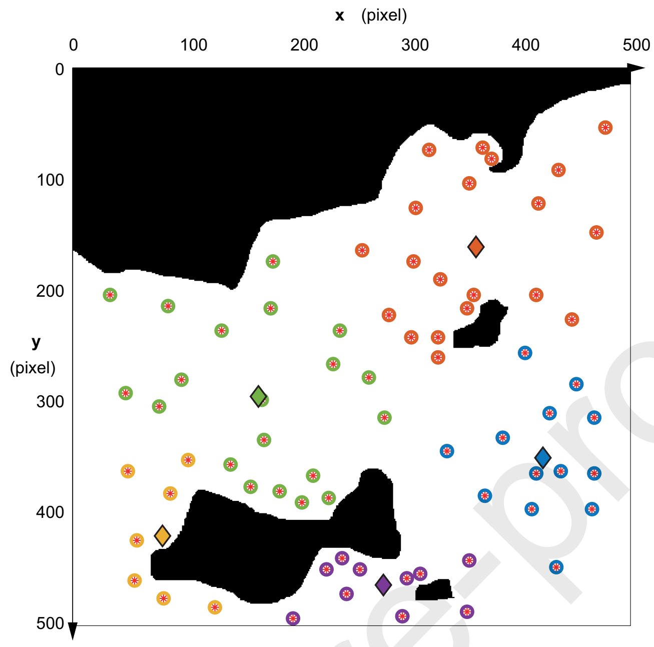
where $Comp_{\mathbf{x}} = (comp_{\mathbf{x}}^1, comp_{\mathbf{x}}^2, \dots , comp_{\mathbf{x}}^k)$ with $k$ being the number of clusters. The lower the $Q$ is, the better the clustering outputs.

Tests in a simulated map with 70 target points have been conducted. By assuming 4 USVs are deployed, 6 tests of the improved K-means algorithms were repeated 5, 10, 20, 50, 100, 400 times, respectively, with random initialisation strategy. The minimum $Q$ and its corresponding iteration where the value is first reached are recorded and shown in Table 3. It can be seen that, a minimal $Q$ value of $2.289 \times 10^{-5}$ can be found in most cases except in the first test where the algorithm has only been run 5 times. Additionally, for the worst outputs, the associated $Q = 2.0$. A comparison between the best and the worst clustering results are presented in Fig. 18. It is evident that the worst case (large value of $Q$) encounters with the local optimum (Fig. 18b), where clusters separated by obstacles (marked by dashed circles) are not acceptable. On the contrary, a small $Q$ gives a better clustering results as shown in Fig. 18a.

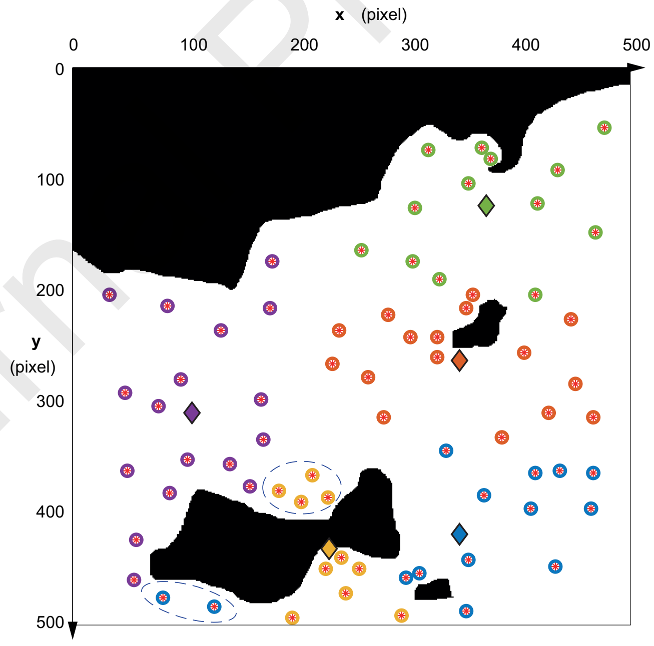
Note that to better address the local minimum, especially when the complexity of problem is high, a perturbation strategy has been used. A random disturbance is given to the location of centroid so that the algorithm can 'move out' the local minimum point and continues to iterate until the whole iteration number reaches a pre-set number or no new local minimum can be found. It also can be seen that, by executing the algorithms repetitively, there is a high possibility that the algorithm can reach the best results (or converge to the best results) within 30 times. Therefore, in this paper, to largely avoid the local minimum problem, the proposed improved K-means will be run 50 times and returns the result with the smallest $Q$.

#### 5.4.2. Computational cost

The improved K-means algorithm and the conventional K-means have the same time complexity which is $O\left(t_k \times n \times k \times d\right)$, where $t_k$ is the iteration times; $n$ is the input



(a) Best result with $Q = 2.289 \times 10^{-5}$



(b) Worst result woth $Q = 2.0$

Figure 18: The best and worst solutions evaluated with the quality score $Q$ (lower value indicates a better result) in a simulated environment with 70 targets. Targets assigned to the same cluster are circled with the same colour and their centroid is marked with a diamond marker. (**a**) The best solution, $Q = 2.289 \times 10^{-5}$. (**b**) The worst solution, $Q = 2.0$.

Table 3: A series of 6 tests over 70 target points in a same map, $Q_{min}$ is the minimum value of the quality score $Q$ throughout a number of repeats of the improved $k$-mean algorithm, and the first occurrence of $Q_{min}$ happens at the $i$th try.

| **Test** | **Repeats** | $Q_{min}(\times 10^{-5})$ | $i$ |
|---|---|---|---|
| 1 | 5 | 30.97 | 4 |
| 2 | 10 | 2.289 | 7 |
| 3 | 20 | 2.289 | 17 |
| 4 | 50 | 2.289 | 24 |
| 5 | 100 | 2.289 | 20 |
| 6 | 400 | 2.289 | 27 |

size and each input is $d$-dimensional; $k$ is the number of clusters. In this work, as the workspace is two-dimensional, $d$ equals to 2. $t_k$ is configured to be 200 to ensure the convergence rate. Therefore, the time complexity of the improved K-means is $O(kn)$, where $k$ and $n$ are the number of base stations and targets, respectively. The time complexity of SOM is $O\left(t_s\left(pn + qn^2\right)\right)$ [22], where $t_s$ is the iteration steps; $p$ and $q$ are the dimensions of inputs and outputs, respectively. In the scenario of this paper, $p$ and $q$ equal to 2, and $t_s$ is set to 5000 for a good convergence rate. Hence the overall time complexity of the SOM-based task execution algorithm is $O\left(n^2\right)$. Because the task execution algorithm is run once for each base station (in total there are $k$ base stations in the system), the time complexity of the entire algorithm is $O\left(kn^2\right)$.

In this section, the computational cost of running four simulations (the conventional K-means in the simulated map (KM), improved K-means in the simulated map (IK), simulation in Southampton estuary (S), and simulation in Réunion adjacent sea area (R)) has been analysed. The computational time records the time of running both the task assignment part (improved K-means) and the task execution part (SOM) with the results shown in Table 4.

It can be observed that when the simulation environments has a small dimension (in a map with 500*500 pixels in both KM and IK simulations), the computational time is relatively fast with 0.97s for KM simulation and 2.61s for IK simulation, which are suitable for on-line decision making. Although the proposed improved K-means share the same inherent structure with conventional K-means, the added computational time for the improved K-means is mainly caused by the inclusion of obstacle obstruction algorithm (Algorithm 2).

However, when the algorithms are implemented for large-scale environments on high-resolution maps (3460*4800 pixels in simulation S and 3113*4800 pixels in simulation R), more time will be taken with 11.40s for simulation S and 50.46s for simulation R. Such a computational time is suitable for an off-line decision-making in refined maps, and proper improvements can be made to reduce the time cost. For example, high-resolution maps can be compressed to a smaller size on which the computation will take less time. Also, when the algorithm is implemented for the on-line decision making, more computational efficient languages such as C++ should be used. In addition, current algorithms are coded in single-threaded execution structure, some subroutines could be parallelised so that the multi-core processor can be used to

Table 4: The computational time of the simulations of 1) the conventional K-means in the simulated map (KM), 2) improved K-means in the simulated map (IK), 3) simulation in Southampton estuary (S), and 4) simulation in Réunion adjacent sea area (R).

| Simulation | USVs | Targets | Map size (pixel) | Computational time (s) |
|---|---|---|---|---|
| KM | 4 | 70 | 500*500 | 0.97 |
| IK | 4 | 70 | 500*500 | 2.61 |
| S | 6 | 90 | 3460*4800 | 11.40 |
| R | 7 | 300 | 3113*4800 | 50.46 |

improve the computational efficiency.

## 6. Conclusions and future work

In this paper, an unsupervised learning based multi-task allocation algorithm has been developed for multi-USV systems. When using the unsupervised learning algorithms, the conventional K-means method has been improved by introducing a weighted competing strategy to better accommodate the requirements in communications. The underlying communication is established upon the case that a USV needs to maintain a stable and robust communication with its base station, and by integrating a practical signal propagation model into the weighted competing strategy, tasks can be assigned by minimising the obstruction to communication by obstacles and the locations of base stations can also be optimised. By taking the assigned tasks, another unsupervised learning algorithm, self-organising map (SOM), is adopted to plan an efficient task execution sequence for USVs to follow. The proposed algorithms have been validated in computer-based simulations, especially by using two practical simulation environments, algorithm's implementation to support vital maritime missions such as coastal patrol and search and rescue has been verified.

In terms of future work, the low repeatability of the algorithm needs to be improved. Because both K-means and SOM algorithms belong to heuristic algorithm, which is prone to local minima problem, a new guidance strategy can be integrated into the algorithm to assist with searching for global rather than local optimum. Although proper local minimum mitigation methods have been applied in this work, further global optimum searching strategies such as the convex optimisation [44], random swap [45] and variable neighbourhood search [46] should be explored especially when the proposed multi-task allocation algorithm is used with a high dimensional input space. Also, in this work, one of the assumption has been made that each USVs have sufficient energy onboard. An adaptive energy strategy that considers the variations in available energy should be incorporated in the algorithm to better address practical issues. Finally, the tuning of parameters should be improved and by using some real-life data, supervised learning algorithm can be used to find a better tuning strategy, which will potentially improve the algorithm's effectiveness.

## 7. Acknowledgements

This work was made possible through a collaboration between University College London (UCL) and Rutgers University, which was partially supported by the International Collaborative Research Grant from Rutgers Global at Rutgers, The State University of New Jersey.

## References

[1] Y. Liu, R. Bucknall, Path planning algorithm for unmanned surface vehicle formations in a practical maritime environment, Ocean Engineering 97 (6) (2015) 126–144.

[2] Y. Liu, R. Bucknall, Efficient multi-task allocation and path planning for unmanned surface vehicle in support of ocean operations, Neurocomputing 275 (2018) 1550–1566.

[3] C. T. Cunningham, R. S. Roberts, An adaptive path planning algorithm for cooperating unmanned air vehicles, in: Proceedings 2001 IEEE International Conference on Robotics and Automation (ICRA), Vol. 4, IEEE, 2001, pp. 3981–3986.

[4] V. K. Shetty, M. Sudit, R. Nagi, Priority-based assignment and routing of a fleet of unmanned combat aerial vehicles, Computers & Operations Research 35 (6) (2008) 1813–1828.

[5] H. A. Kurdi, E. Aloboud, M. Alalwan, S. Alhassan, E. Alotaibi, G. Bautista, J. P. How, Autonomous task allocation for multi-uav systems based on the locust elastic behavior, Applied Soft Computing 71 (2018) 110–126.

[6] Y. Deng, P.-P. Beaujean, E. An, E. Carlson, Task allocation and path planning for collaborative auvs operating through an underwater acoustic network, in: OCEANS 2010 MTS/IEEE SEATTLE, IEEE, 2010, pp. 1–9.

[7] D. Zhu, Y. Qu, S. X. Yang, Multi-auv som task allocation algorithm considering initial orientation and ocean current environment, Frontiers of Information Technology & Electronic Engineering 20 (3) (2019) 330–341.

[8] E. Raboin, P. Švec, D. S. Nau, S. K. Gupta, Model-predictive asset guarding by team of autonomous surface vehicles in environment with civilian boats, Autonomous Robots 38 (3) (2015) 261–282.

[9] P. Scerri, B. Kannan, P. Velagapudi, K. Macarthur, P. Stone, M. Taylor, J. Dolan, A. Farinelli, A. Chapman, B. Dias, et al., Flood disaster mitigation: A real-world challenge problem for multi-agent unmanned surface vehicles, in: International Conference on Autonomous Agents and Multiagent Systems, Springer, 2011, pp. 252–269.

[10] L. Wang, J. Zhang, H. Li, An improved genetic algorithm for tsp, in: 2007 International Conference on Machine Learning and Cybernetics, Vol. 2, IEEE, 2007, pp. 925–928.

[11] M. X. Goemans, Worst-case comparison of valid inequalities for the tsp, Mathematical Programming 69 (1-3) (1995) 335–349.

[12] A. Scholz, S. Henn, M. Stuhlmann, G. Wäscher, A new mathematical programming formulation for the single-picker routing problem, European Journal of Operational Research 253 (1) (2016) 68–84.

[13] K. Helsgaun, General k-opt submoves for the lin–kernighan tsp heuristic, Mathematical Programming Computation 1 (2-3) (2009) 119–163.

[14] F. Glover, G. Gutin, A. Yeo, A. Zverovich, Construction heuristics for the asymmetric tsp, European Journal of Operational Research 129 (3) (2001) 555–568.

[15] A. A. Hosseinabadi, M. Kardgar, M. Shojafar, S. Shamshirband, A. Abraham, Gels-ga: hybrid metaheuristic algorithm for solving multiple travelling salesman problem, in: 2014 14th International Conference on Intelligent Systems Design and Applications, IEEE, 2014, pp. 76–81.

[16] L. Bianchi, M. Birattari, M. Chiarandini, M. Manfrin, M. Mastrolilli, L. Paquete, O. Rossi-Doria, T. Schiavinotto, Hybrid metaheuristics for the vehicle routing problem with stochastic demands, Journal of Mathematical Modelling and Algorithms 5 (1) (2006) 91–110.

[17] T. Kohonen, Self-organized formation of topologically correct feature maps, Biological cybernetics 43 (1) (1982) 59–69.

[18] T. Kohonen, P. Somervuo, Self-organising maps of symbol strings, Neurocomputing 21 (1) (1998) 19 – 30. `doi:https://doi.org/10.1016/S0925-2312(98)00031-9`.

[19] H. Sasamura, R. Ohta, T. Saito, A simple learning algorithm for growing ring som and its application to *tsp*, in: Proceedings of the 9th International Conference on Neural Information Processing, 2002. ICONIP ’02, Vol. 3, IEEE, 2002, pp. 1287–1290 vol.3.

[20] M. L. Kleinhans, D. V. Martín, Using Self-Organizing Maps to solve Travelling Salesman Problem, Tech. rep., Norges teknisk-naturvitenskapelige universitet, Høgskoleringen, [Accessed on 18th March 2019] (2016).
URL `https://github.com/DiegoVicen/ntnu-som`

[21] J. Faigl, An application of self-organizing map for multirobot multigoal path planning with minmax objective, Computational intelligence and neuroscience 2016.

[22] Y. Liu, R. Song, R. Bucknall, X. Zhang, Intelligent multi-task allocation and planning for multiple unmanned surface vehicles (usvs) using self-organising maps and fast marching method, Information Sciences 496 (2019) 180–197.

[23] R. Song, Y. Liu, J. Balbuena, F. Cuellar, R. Bucknall, Developing an energy effective autonomous usv for undertaking missions at the highlands of peru, in: 2018 OCEANS-MTS/IEEE Kobe Techno-Oceans (OTO), IEEE, 2018, pp. 1–7.

[24] J. MacQueen, Some methods for classification and analysis of multivariate observations, in: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, Vol. 1, Oakland, CA, USA, 1967, pp. 281–297.

[25] A. A. Abbasi, M. Younis, A survey on clustering algorithms for wireless sensor networks, Computer communications 30 (14-15) (2007) 2826–2841.

[26] A. Fahad, N. Alshatri, Z. Tari, A. Alamri, I. Khalil, A. Y. Zomaya, S. Foufou, A. Bouras, A survey of clustering algorithms for big data: Taxonomy and empirical analysis, IEEE transactions on emerging topics in computing 2 (3) (2014) 267–279.

[27] M. Elango, S. Nachiappan, M. K. Tiwari, Balancing task allocation in multi-robot systems using k-means clustering and auction based mechanisms, Expert Systems with Applications 38 (6) (2011) 6486–6491.

[28] J. Waterston, J. Rhea, S. Peterson, L. Bolick, J. Ayers, J. Ellen, Ocean of things: Affordable maritime sensors with scalable analysis, in: OCEANS 2019-Marseille, IEEE, 2019, pp. 1–6.

[29] A. Coates, A. Y. Ng, Learning feature representations with k-means, in: Neural networks: Tricks of the trade, Springer, 2012, pp. 561–580.

[30] G. Cleuziou, An extended version of the k-means method for overlapping clustering, in: 2008 19th International Conference on Pattern Recognition, IEEE, 2008, pp. 1–4.

[31] S. J. Redmond, C. Heneghan, A method for initialising the k-means clustering algorithm using kd-trees, Pattern recognition letters 28 (8) (2007) 965–973.

[32] G. Hamerly, C. Elkan, Alternatives to the k-means algorithm that find better clusterings, in: Proceedings of the eleventh international conference on Information and knowledge management, ACM, 2002, pp. 600–607.

[33] H.-H. Bock, Clustering methods: a history of k-means algorithms, in: Selected contributions in data analysis and classification, Springer, 2007, pp. 161–172.

[34] S. Kim, H. Oh, J. Suk, A. Tsourdos, Coordinated trajectory planning for efficient communication relay using multiple uavs, Control Engineering Practice 29 (2014) 42–49.

[35] Z. Han, A. L. Swindlehurst, K. R. Liu, Optimization of manet connectivity via smart deployment/movement of unmanned air vehicles, IEEE Transactions on Vehicular Technology 58 (7) (2009) 3533–3546.

[36] J. E. Bresenham, Algorithm for computer control of a digital plotter, IBM Systems journal 4 (1) (1965) 25–30.

[37] T. Martinetz, K. Schulten, A "neural gas" network learns topologies, artificial neural networks, in: Proceedings of the 1991 International Conference on Artificial Neural Networks, ICANN-91, Vol. 1, 1991, pp. 397–402.

[38] B. Fritzke, A growing neural gas network learns topologies, in: Advances in neural information processing systems, 1995, pp. 625–632.

[39] J. Hollmën, Process modeling using the self-organizing map.

[40] B. Angeniol, G. D. L. C. Vaubois, J.-Y. Le Texier, Self-organizing feature maps and the travelling salesman problem, Neural Networks 1 (4) (1988) 289–293.

[41] Ł. Brocki, D. Koržinek, Kohonen self-organizing map for the travelling salesperson problem, in: R. Jabłoński, M. Turkowski, R. Szewczyk (Eds.), Recent Advances in Mechatronics, Springer, Berlin, Heidelberg, 2007, pp. 116–119.

[42] D. Aloise, P. Hansen, L. Liberti, An improved column generation algorithm for minimum sum-of-squares clustering, Mathematical Programming 131 (1-2) (2012) 195–220.

[43] D. J. Hand, W. J. Krzanowski, Optimising k-means clustering results with standard software packages, Computational Statistics & Data Analysis 49 (4) (2005) 969 – 973. `doi:https://doi.org/10.1016/j.csda.2004.06.017`.

[44] A. M. Bagirov, S. Taheri, J. Ugon, Nonsmooth dc programming approach to the minimum sum-of-squares clustering problems, Pattern Recognition 53 (2016) 12–24.

[45] P. Fränti, Efficiency of random swap clustering, Journal of Big Data 5 (1) (2018) 13.

[46] P. Hansen, N. Mladenović, J. A. M. Pérez, Variable neighbourhood search: methods and applications, Annals of Operations Research 175 (1) (2010) 367–407.