Decentralised Self-Organising Maps for Multi-Robot Information Gathering

Graeme Best and Geoffrey A. Hollinger

***Abstract*— This paper presents a new coordination algorithm for decentralised multi-robot information gathering. We consider planning for an online variant of the multi-agent orienteering problem with neighbourhoods. This formulation closely aligns with a number of important tasks in robotics, including inspection, surveillance, and reconnaissance. We propose a decentralised variant of the self-organising map (SOM) learning procedure, named Dec-SOM, which efficiently plans sequences of waypoints for a team of robots. Decentralisation is achieved by performing a distributed allocation scheme jointly with a series of SOM adaptations. We also offer an efficient heuristic to select when to perform negotiations, which reduces communication resource usage. Simulation results in two settings, including an infrastructure inspection scenario with a real-world dataset of oil rigs, demonstrate that Dec-SOM outperforms baseline methods and other SOM variants, is competitive with centralised SOM, and is a viable solution for decentralised information gathering.**

I. Introduction

Mobile robots are increasingly being used to gather information about their environment. Robotic information gathering is important in a diverse range of applications, including infrastructure inspection [2], mine countermeasures [3], precision agriculture [4], and data collection from sensor networks [5]. Many information gathering scenarios, especially those that can be formulated as mapping, coverage, or search problems, involve observing a set of points of interest (POIs) from associated observation regions [2]–[12]. The problem is to plan action sequences for robots that maximise the number of observed POIs while satisfying time or energy budgets.

Performing information gathering with multiple robots simultaneously enables scaling up the number of observations in time and space. However, to achieve desirable performance, the robots are required to effectively coordinate their actions. Preferably, this coordination is decentralised, such that each robot plans primarily for itself while communicating their intent to other robots, if communication is available. This ensures that the team is more robust to unreliable communication and inconsistent beliefs, and avoids having a single point of failure. These properties are especially beneficial in domains where communication is challenging, such as marine environments [13] and underground tunnels [14]. Decentralised coordination is difficult, particularly when communication is unreliable, because the robots must plan action sequences while only having partial, and possibly incorrect, knowledge of the plans of other robots [12].

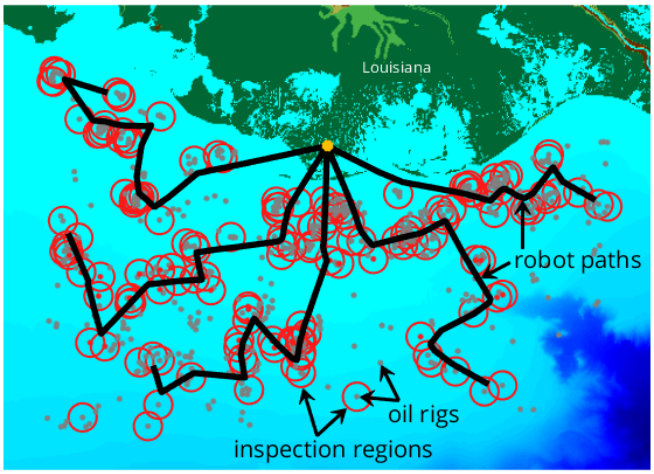

Fig. 1: **Infrastructure inspection task:** A team of 5 robots coordinate their plans (black paths) to efficiently inspect the set of oil rigs off the coast of Louisiana, USA. The robots discover which subset of the 813 oil rigs (dots) should be inspected at close range (disks) as they explore the environment. Environment size: $200 \times 100\,\text{km}^2$. Budget: 175 km paths.

Self-organising maps (SOMs) are a special class of learning procedures that aim to find a low-dimensional representation of an input space while preserving a given topology of the representation. SOMs have been widely applied to inference problems [15], but have also been adapted for path planning problems, such as variants of the travelling salesman problem (TSP) [16]. Although SOM algorithms are not competitive with state-of-the-art TSP solvers, they are particularly powerful at solving problems that require *jointly* optimising the selection of viewpoints and the path through these viewpoints. This includes the TSP with neighbourhoods and related variants, which require selecting viewpoints from continuous sets that represent observation regions [5], [7], [17], [18]. Multi-robot SOM variants have been proposed [5], [7], [19]; however, all of these methods are centralised. This paper proposes the first *decentralised* SOM algorithm for multi-robot path planning.

We present the decentralised self-organising map algorithm (Dec-SOM) for multi-robot information gathering. The task is formulated as a generalisation of the orienteering problem with neighbourhoods, where a set of continuous regions are to be maximally visited, and these regions are discovered online. The Dec-SOM algorithm consists of each robot optimising their path, defined as a sequence of

*This work was supported in part by Office of Naval Research Grant N00014-17-1-2581.

*A preliminary version of this work appeared as an extended abstract in [1].

*The authors are with the Collaborative Robotics and Intelligent Systems (CoRIS) Institute, Oregon State University, Corvallis OR, USA. {bestg,geoff.hollinger}@oregonstate.edu

waypoints in continuous space, using a series of SOM adaptations, and negotiating with other robots for the allocation of goal regions. During the negotiations, the value of goal region allocations are estimated by evaluating the effect of SOM adaptations. An efficient heuristic is also proposed to selectively decide when to perform the negotiations to reduce bandwidth usage. The algorithm is online in that it efficiently adapts to changes in the world estimates and plans of other robots by replanning while using the previous plan as a prior.

We demonstrate the performance of the algorithm in two simulated scenarios: a random partially-known world setting, and an infrastructure inspection task illustrated in Fig. 1 with a real-world dataset of oil rigs [20]. Dec-SOM is shown to (1) outperform a number of baseline comparison methods and alternative SOM approaches, (2) be competitive with centralised SOM [7], (3) plan effectively in partially-known worlds, (4) efficiently adapt to changing information, and (5) experience a gradual degradation of performance as communication becomes less reliable.

II. Related Work

Informative path planning is the problem of finding paths that maximise an information gain metric, subject to budget constraints [21]. Commonly, the considered information metric is an uncertainty measure for the belief of a quantity of interest. However, such metrics are often computationally demanding or not applicable for many tasks. Another approach is to formulate objectives as a set of POIs to be observed, as we do in this paper. These objectives naturally align with tasks such as area coverage [2], [6], classifying physical objects [4], [7]–[9], or observing scientifically valuable regions of oceans [10]. These objectives are typically faster to compute, which enables the efficient use of sophisticated non-myopic planners.

Despite the benefits of multi-robot systems, relatively little attention has been given to developing decentralised planners for information gathering. Dec-MDP formulations [22] are closely related to the considered problem, but are typically solved in an offline and centralised manner, in contrast to the decentralised setting we consider here. Additionally, Dec-MDPs are typically formulated with discrete action spaces, unlike our proposed approach, which plans directly over continuous goal regions without requiring discretisation. Dec-MCTS [12] is a generally-applicable algorithm that is suitable for decentralised settings similar to what we consider here, although it has been shown to be outperformed by task-specific SOM approaches [7]. Decentralised algorithms have been proposed for exploration and mapping, such as the sequential greedy assignment algorithm [23]. While [23] formulates an information-theoretic objective, most prior work involves extracting and reasoning over 'frontiers' [24]. These frontier formulations, and task planning more generally, are often solved with market-based approaches [25]–[27]. Dec-SOM has similarities to these methods, but the allocations are optimised *jointly* with path planning. Few approaches have explicitly optimised for communication requirements [28]–[30]; we propose an efficient heuristic within Dec-SOM to reduce bandwidth usage.

SOMs have recently emerged as a powerful method for path planning that involves range sensing. SOMs have been adapted for TSP generalisations that require selecting favourable viewpoints within continuous goal regions, such as the watchman route problem [17] and the orienteering problem with neighbourhoods [18]. These have been applied to problems such as data collection from sensor networks [5], perception of 3D objects [7], and area surveillance [11]. All of these SOM variants are for single-robot or *centralised* multi-robot systems. Outside of path planning, SOMs are popular for pattern analysis [15]; however, even in these other contexts, little attention has been given to distributed computation, despite being parallelisable [31], [32].

The formulation in this paper can be interpreted as a variant of the TSP or orienteering problem [33]. With the exception of SOMs, no existing TSP solvers adequately address continuous goal regions. Multi-robot TSP generalisations have been considered, but most solvers are centralised, with the exception of the Dec-MCTS and market-based approaches referenced above. Similarly, TSP variants are typically considered in offline settings, with [34] being a notable exception.

III. Online Multi-agent Orienteering Problem with Neighbourhoods

We consider the problem of planning the actions for a team of robots in a decentralised manner. We address a complex coordination task that requires the robots to collectively visit a maximal number of continuous goal regions. An online variant is formulated where an estimate of the goal regions is refined as the robots explore the world. An example problem instance is illustrated in Fig. 2.

A. Multi-robot team

The team consists of a set of R robots $\mathcal{R} = \{1, 2, ..., R\}$. The path of each robot $r \in \mathcal{R}$ is described as a sequence of waypoint locations $\boldsymbol{x}^r = (x_0^r, x_1^r, x_2^r, ...)$ with $x_i^r \in \mathbb{R}^2, \forall i$ and x_0^r is a fixed start location. Each robot r has a budget B^r which may enforce an energy constraint or time horizon. A path is feasible if the path length satisfies the budget, i.e.,

$$\sum_{(x_i^r, x_{i+1}^r) \in \boldsymbol{x}^r} \left\| x_{i+1}^r - x_i^r \right\| \leq B^r, \tag{1}$$

where $\| \cdot \|$ is the Euclidean norm. We use the following superscript convention: $\boldsymbol{x} := \{\boldsymbol{x}^1, \boldsymbol{x}^2, ..., \boldsymbol{x}^R\}$ is the set of paths of all robots, and $\boldsymbol{x}^{(r)} := \boldsymbol{x} \setminus \{\boldsymbol{x}^r\}$ is the set of paths of all robots except robot r.

B. Goal regions and estimate

The world consists of a set of goal regions $\mathcal{Z}$ to be visited. Each goal region $z_j \in \mathcal{Z}$ is a disk with centre $c_j \in \mathbb{R}^2$ and radius $\rho_j \in \mathbb{R}$. A waypoint x_i^r is said to visit goal z_j iff $\|x_i^r - c_j\| \leq \rho_j$. Each goal region can be interpreted as a 'neighbourhood' around a POI c_j. We assume the neighbourhoods are circular, although the

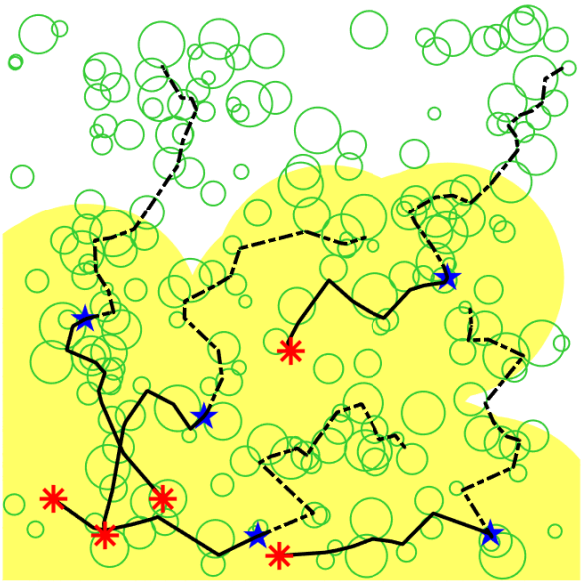

Fig. 2: Representative scenario and partial solution for the random worlds experiments. Red stars are start locations. Robots have executed paths (solid lines) up to the blue stars. Dotted lines are the planned plans. Green circles represent goal regions. Yellow shade represents explored regions assuming a circular observation model. Unshaded areas are a prediction of the unknown environment.

proposed algorithm can readily be adapted for polygonal goals, as in [7].

The set of goal regions $\mathcal{Z}$ is not necessarily known to the robots in advance. Instead, the robots have an estimate $\tilde{\mathcal{Z}} \approx \mathcal{Z}$, such that the estimate is refined as the robots move around the environment. Although we make no explicit assumptions regarding the quality of the estimate $\tilde{\mathcal{Z}}$, the proposed algorithm is designed to work best when the estimated existence or non-existence of a potential goal z_j is correct once any robot visits a location within a distance d of z_j, with $d \geq \rho_j$. The example problem illustrated in Fig. 2 has this property, where the yellow shaded region near the robots is known while the unshaded region is a prediction. We provide two concrete definitions of prediction models $\tilde{\mathcal{Z}}$ for the experimental scenarios in Sec. V. We note that we assume that the robots do not have an explicit model for how the estimate $\tilde{\mathcal{Z}}$ will be refined, thus the information collected regarding $\tilde{\mathcal{Z}}$ will be opportunistic rather than directly considered by the planner. We also focus on the case where robots are assumed to agree on $\tilde{\mathcal{Z}}$, which may be achieved using decentralised data fusion, although the proposed algorithm is valid if these beliefs are inconsistent.

C. Decentralised planning problem

The problem we aim to address is stated as follows.

Problem 1 (Multi-agent Orienteering Problem with Neighbourhoods): Find the set of *feasible* paths $\boldsymbol{x}$ for the team of robots that collectively maximises the number of goal regions $z_j \in \mathcal{Z}$ that are visited from at least one waypoint in $\boldsymbol{x}$.

We emphasise that, in Problem 1, there is no *additional* reward for visiting a continuous goal region more than once or by multiple robots, thus requiring careful planning and coordination.

Decentralised setting: Problem 1 is to be solved in a decentralised setting. To enable decentralisation, we assume that each robot r can only modify its own planned path $\boldsymbol{x}^r$. The robots are assumed to be able to communicate to aid effective coordination. However, communication may be unreliable—such as having significant packet loss, low bandwidth, or high latency—and thus the robots should still plan reasonable joint paths even when communication is imperfect.

Online planning: As discussed above, the robots maintain an estimate $\tilde{\mathcal{Z}}$ for the environment $\mathcal{Z}$, such that the estimate is refined as the robots move around the environment. Therefore, the robots should adapt their plans in an online manner as this estimate $\tilde{\mathcal{Z}}$ changes, as well as when the plans $\boldsymbol{x}^{(r)}$ for the other robots change.

IV. Dec-SOM Algorithm

We propose Dec-SOM as a solution to Problem 1 in the decentralised and online setting defined above. The algorithm generalises the SOM learning procedure to be suitable for this decentralised setting by employing a novel distributed allocation scheme within the series of SOM adaptations. This section begins by summarising the algorithm, detailing the key algorithmic components, then providing a brief analysis.

A. Algorithm overview

A self-organising map provides a lower-dimensional representation of an input space, where the representation preserves a given topological structure. In our case, the input space is the goal regions that can be visited by robot r. The SOM aims to find a path for robot r that 'best fits' this input space. The path $\boldsymbol{x}^r$ directly defines the topological structure used by the SOM, where vertices are the waypoints of the path, and edges connect consecutive waypoints. The algorithm jointly learns both: (1) the allocation of the input space (goal regions) to individual robots and (2) the path that maximally visits the allocated goal regions. This learning is performed simultaneously and asynchronously by all robots.

We present the algorithm from the perspective of robot r at a given time instant. Pseudocode is provided in Alg. 1. The main loop (line 2) cycles between: (1) select a goal region at random (lines 4-5), (2) adapt the plan for robot r towards this goal region using an SOM adaptation procedure (line 7), (3) determine the value of this adaptation and, if appropriate, request the allocation of this goal from other robots (lines 8-13), (4) retain or discard this adaptation (lines 15-16), and (5) periodically regenerate the plan (line 18). In parallel, robots process and reply to incoming allocation requests.

The primary purpose of the allocation step is to coordinate the plans between multiple robots, whereas the other steps aim to plan efficient paths with respect to these allocations. All steps are further described and justified as follows.

B. Initialisation

The path $\boldsymbol{x}^r$ is initialised as a single waypoint at x_0^r. As the algorithm progresses, waypoints of $\boldsymbol{x}^r$ will be added,

Algorithm 1 Dec-SOM algorithm from the perspective of robot r at a given time instant.

Input: prior plan $\boldsymbol{x}^r$, budget B^r
a set of goal regions $\tilde{\mathcal{Z}}$, prior allocations $\mathcal{Z}^r$
adaptation parameters σ, δ
Output: updated plan $\boldsymbol{x}^r$, updated allocations $\mathcal{Z}^r$

1: ▷ Repeat for a fixed number of epochs
2: **for** $i = 1$ to num_epochs **do**
3: ▷ Consider each goal region in a random order
4: $perm \leftarrow$ random permutation of $\tilde{\mathcal{Z}}$
5: **for each** $z_j \in \tilde{\mathcal{Z}}$, in order $perm$ **do**
6: ▷ Perform SOM adaptation towards z_j
7: $\boldsymbol{x}^r_+ \leftarrow \text{ADAPT}(\boldsymbol{x}^r, z_j, \sigma)$ ▷ See Sec. IV-C.1
8: **if** $L(\boldsymbol{x}^r_+) \leq B^r$ **then** ▷ Path length $\leq$ budget
9: **if** $z_j \notin \tilde{\mathcal{Z}}^r$ **then** ▷ If not allocated
10: ▷ Request allocation (see Sec. IV-C.2)
11: $s \leftarrow$ Eqn. (2)
12: **if** $\text{REQUESTALLOCATION}(z_j, s)$ **then**
13: $\tilde{\mathcal{Z}}^r \leftarrow \tilde{\mathcal{Z}}^r \cup \{z_j\}$
14: ▷ If allocated, retain the adaptation
15: **if** $z_j \in \tilde{\mathcal{Z}}^r$ **then**
16: $\boldsymbol{x}^r \leftarrow \boldsymbol{x}^r_+$
17: ▷ Remove undesirable waypoints
18: $\boldsymbol{x}^r \leftarrow \text{REGENERATE}(\boldsymbol{x}^r, \tilde{\mathcal{Z}}^r)$ ▷ See Sec. IV-C.4
19: $\sigma \leftarrow (1 - i\delta)\sigma$ ▷ Update adaptation parameter

removed or adapted. The path could also be initialised as a prior solution, and we take advantage of this during online replanning (see Sec. IV-D). SOMs are generally not overly sensitive to the initial conditions, provided that the initial value of the learning parameter σ (defined below in Sec. IV-C.1) is sufficiently high.

Robot r also maintains an allocated set of goal regions $\tilde{\mathcal{Z}}^r$ that robot r should visit, where $\tilde{\mathcal{Z}}^r \subseteq \tilde{\mathcal{Z}}$. Initially $\tilde{\mathcal{Z}}^r = \emptyset$, i.e., no goal regions are allocated to robot r, but this set will expand and shrink as the robots negotiate with each other.

C. Learning procedure

We now detail the key components of Dec-SOM. The algorithm consists of a sequence of *epochs*. During each epoch, each goal region is considered one at a time, in a random order. When a goal region z_j is presented, the path $\boldsymbol{x}^r$ is adapted towards z_j by moving and potentially adding waypoints. If goal region z_j is deemed to be desirable, then, if necessary, robot r requests the allocation of z_j. In parallel, robots respond to incoming allocation requests. This learning continues for a fixed number of epochs. At each epoch, a learning parameter is adjusted to modify the adaptation behaviour. Undesirable waypoints are removed from $\boldsymbol{x}^r$ at the end of every epoch. The start waypoint x^r_0 is never moved or removed, but may be duplicated.

1) SOM adaptation: The SOM adaptation step is essential for generating desirable paths that efficiently visit a large number of goal regions. An illustration of SOM adaptation is presented in Fig. 3. This is performed at line 7 of Alg. 1.

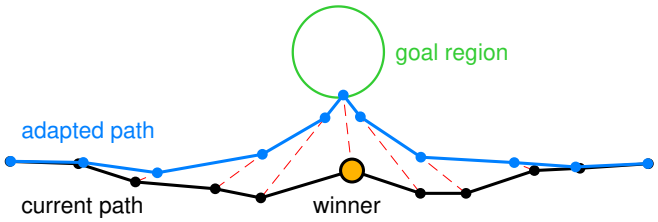

Fig. 3: Illustration of a SOM adaptation towards a goal region. The 'winner' moves to the closest point in the goal region. Topological neighbours move by a decreasing fraction towards the winner. In this example, $\sigma = 2$.

Formally, when goal region z_j is presented during an epoch, the closest waypoint x^r_i or edge (x^r_i, x^r_{i+1}) in $\boldsymbol{x}^r$ to *any* point $z^\star_j$ on the disk z_j is selected as the *winner*. If the winner is an edge, then a new waypoint is inserted between x^r_i and x^r_{i+1}. In either case, the winner waypoint is moved to $z^\star_j$. All other waypoints in x^r_i are moved towards $z^\star_j$ by a fraction $f(\sigma, l) = e^{-\frac{l^2}{\sigma^2}}$, where l is the *number* of waypoints between this waypoint and the winner, and σ is a learning parameter. Intuitively, waypoints that are topologically closer to the winner move a larger fraction towards $z^\star_j$; thus, a relatively small Euclidean distance is retained between consecutive waypoints of $\boldsymbol{x}^r$. We note that an adaptation may cause waypoints to be moved outside their associated goal region; this is desirable since it forces regenerating parts of the solution that have a high cost with low reward (see Sec. IV-C.4).

The learning parameter σ slowly decreases at the end of each epoch (line 19) as $\sigma \leftarrow (1 - i\delta)\sigma$, where δ is a fixed learning rate, and i is the epoch number. This results in global adaptations initially, then eventually only local adaptations.

If an adaptation results in the budget (1) being unsatisfied (line 8 is false), then the adaptation is discarded and the previous solution retained. If z_j is already allocated to robot r (line 15), then the new solution is immediately retained.

2) Distributed allocations: During the adaptation phase, if robot r desires to retain an adaptation towards a goal region z_j, but robot r has not been allocated z_j, then robot r requests this allocation by negotiating with other robots. This occurs at lines 11-13. Robot r broadcasts the goal region z_j and an associated score. Then robot r processes responses from other robots to determine the new allocation for z_j.

The score s associated with an allocation request for goal region z_j is designed to favour allocations that fit in well with the existing plan $\boldsymbol{x}^r$. Define $L(\text{ADAPT}(\boldsymbol{x}^r, z_j, \sigma = 0))$ as the path length of $\boldsymbol{x}^r$ after performing the SOM adaptation procedure towards z_j. Define $L(\text{REMOVE}(\boldsymbol{x}^r, z_j))$ as the path length of $\boldsymbol{x}^r$ after removing any waypoints (possibly none) that are within z_j and not within any other goal region. A favourable allocation is likely to have a small difference between these two path lengths, thus the score s is a normalised difference between these lengths. Specifically,

$$s = \frac{L(\text{ADAPT}(\boldsymbol{x}^r, z_j, 0)) - L(\text{REMOVE}(\boldsymbol{x}^r, z_j))}{\hat{B}^r - L(\text{REMOVE}(\boldsymbol{x}^r, z_j))}, \quad (2)$$

where $\hat{B}^r$ is the remaining budget after subtracting the *executed* path length from B^r. The denominator in (2) is designed to favour allocating goal regions to robots that currently use a smaller fraction of their budget. Smaller s values are favourable. For the adaptation in (2) we set $\sigma = 0$ so that the comparisons are fair between robots.

Robot r waits for a response from other robots. Another robot r' will reply to this request if r' believes z_j has been allocated to r'. Robot r' also computes (2) as the utility of the z_j allocation. If a replying robot has a score higher than the initiating robot, then locally the replying robot removes the allocation, i.e., $\tilde{\mathcal{Z}}^{r'} \leftarrow \tilde{\mathcal{Z}}^{r'} \setminus \{z_j\}$. Multiple, one, or no robots may reply, since partitioning is not strictly enforced by $\tilde{\mathcal{Z}}^r$, particularly if communication is unreliable. After a fixed time, robot r processes the replies. If robot r has the lowest s then it receives the allocation of z_j, adds it to $\tilde{\mathcal{Z}}^r$ and $\boldsymbol{x}^r$ retains the adaptation.

3) Selective communication: The distributed allocation step above can result in unnecessary communication for requesting allocations that are quickly reversed in later epochs. If communication is particularly restricted, we offer a simple and efficient heuristic to reduce the number of requests.

In this case, an allocation request is initiated if it meets the above criteria, and additionally meets the following more restrictive criteria. Define $L(\boldsymbol{x}^r)$ as the path length of $\boldsymbol{x}^r$, and $L(\textsc{adapt}(\boldsymbol{x}^r, z_j, \sigma))$ as the path length after the intended adaptation. The adaptation uses the current σ value. An allocation request is only initiated if the increase in path length as a result of the adaptation is not more than κ times the average edge length in $\boldsymbol{x}^r$, where κ is a constant, i.e.,

$$L(\textsc{adapt}(\boldsymbol{x}^r, z_j, \sigma)) - L(\boldsymbol{x}^r) \leq \kappa \underset{(x_i^r, x_{i+1}^r) \in \boldsymbol{x}^r}{\text{average}} \|x_{i+1}^r - x_i^r\|.$$

The intuition behind this heuristic is to avoid requesting allocations that cause relatively large changes to the path. Although this selection approach may not be appropriate in all situations, we show empirically that it works well in the two problem domains considered in Sec. V.

4) Regeneration: At the end of each epoch (line 18), $\boldsymbol{x}^r$ is regenerated to ensure the paths are efficient. This is performed by removing all waypoints x_i^r from $\boldsymbol{x}^r$ that are not visiting a goal region allocated to robot r. A removal may occur for several reasons: x_i^r may not have been selected as a winner in this epoch, x_i^r may have moved out of a goal region due to other adaptations, or the allocations may have changed during this epoch.

D. Online replanning

In an online setting, defined in Sec. III, the information available to the robots changes as they execute their paths. This changing information could be in regards to the environment estimate $\tilde{\mathcal{Z}}$ or the allocations and plans $\boldsymbol{x}^{(r)}$. We suggest the use of replanning to efficiently adapt to changing information as the robots move around the environment. This is in contrast to offline policy planning that would require planning for all possible changes to the environment estimate. SOMs are well suited for online replanning as previous plans can be used as a prior to initialise the optimisation. Furthermore, if the changes to the information are small, then it is not necessary to do global SOM adaptation, and therefore σ can be initialised to be relatively small [7, Sec. 7.2.4]. If $\tilde{\mathcal{Z}}$ changes within replanning rounds, then each robot removes all allocations $z_j : z_j \notin \tilde{\mathcal{Z}}$ from $\tilde{\mathcal{Z}}^r$.

E. Analysis

Here, we remark on several practically significant analytical properties of Dec-SOM. The computational complexity is $\mathcal{O}(|\tilde{\mathcal{Z}}|^2 N + |\tilde{\mathcal{Z}}|A)$, where N is the number of epochs, and A is the number of allocation requests that require a reply. In [7, Thm. 1] and elsewhere, it is noted that N is constant for a given σ cooling schedule, as eventually no further adaptations occur. The number of allocation requests that require replying is $A = \mathcal{O}(|\tilde{\mathcal{Z}}|NR)$ in the worst case; however, A will be significantly fewer in practice as it is highly unlikely that all goal regions would be allocated to robot r and that all robots request all possible allocations in every epoch. Nevertheless, the selection approach in Sec. IV-C.3 is proposed to reduce A in practice. The number of communication messages sent is $\mathcal{O}(A)$, with messages of size $\mathcal{O}(1)$. We note that an allocation request involves waiting for a response from other robots; however, this could be implemented as a non-blocking wait, and thus have only a small effect on computation time. No strong theoretical claims are made for the optimality of SOMs, but below we show empirically that Dec-SOM works well in practice.

V. Experiments

We present a set of experiments to demonstrate the behaviour and performance of the proposed Dec-SOM algorithm. We consider two scenarios: randomised worlds where goal regions are discovered as the robots explore the environment, and an infrastructure inspection scenario where a partially-known subset of assets need to be observed at close range. The results aim to show that Dec-SOM outperforms several baseline methods and SOM variants, and is an effective decentralised planner for scenarios with partially-known goals and unreliable communication.

A. Random worlds

In this set of experiments, we demonstrate the performance of the algorithm in comparison to various baseline comparison methods. Results are shown for the case where communication is perfect and the case where communication is lossy. We formalise these methods and scenarios as follows.

1) Experimental setup: These experiments involve planning for a team of 5 robots in random worlds, as illustrated earlier in Fig. 2. The world is $100 \times 100\,\text{m}^2$ and consists of 200 goal regions, that have uniform-random centre locations, and uniform-random radii between 1 and 4 m. The start locations are random near the bottom left, the budgets are $B^r = 80\,\text{m}$ and replanning occurs every 20 m of distance travelled. Goals within 25 m of the executed path are known precisely to the robots. The example world estimate $\tilde{\mathcal{Z}}$ model

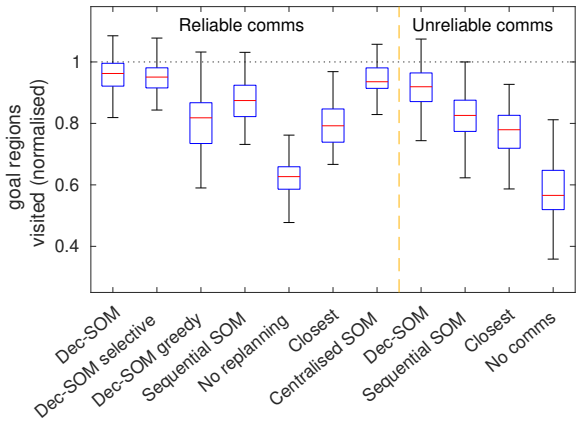

Fig. 4: Performance of Dec-SOM and various comparison methods in 100 random environments. The *unreliable comms* scenarios simulated random packet loss following the model described in Sec. V-A.1. Scores are normalised by dividing by the performance of a Dec-SOM planner that has access to an *oracle* in each environment. Box plot shows the bounds and quartiles.

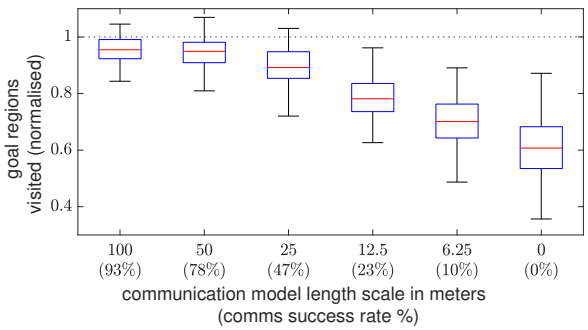

Fig. 5: Performance of Dec-SOM as the communication reliability degrades. Scores are normalised by dividing by the performance of *oracle*.

used for these experiments generates random goal regions in the unobserved areas using the same model as for the ground-truth worlds.

We compare Dec-SOM to several alternative methods. We refer to the proposed algorithm as 'Dec-SOM', and the proposed algorithm with the communication selection described in Sec. IV-C.3 (with $\kappa = 1$) as *selective*. The *greedy* method is the same as Dec-SOM but with a planning horizon of 20 m. The *sequential* method plans for one robot at a time using a single-robot SOM while keeping the other robots' plans $\boldsymbol{x}^{(r)}$ fixed. The *no replanning* method is the same as Dec-SOM but executes the initial plans without any replanning to take into account changing information. The *closest* method iteratively selects the closest unvisited goal region. The *centralised* method is a centralised implementation of our method, similar to [7] except instead uses our proposed scoring function (2) to determine the winning robot at each iteration. Results are normalised with respect to Dec-SOM with an *oracle* where $\mathcal{Z}$ is known fully in advance. For the SOM methods, learning parameters were set as $\sigma = 2, \delta = 0.001$; prior work with related algorithms indicate that SOMs are not particularly sensitive to the choice of these parameters [7]. Computation times for all methods were on the order of several seconds or less for the initial planning round, then 1 s or less for subsequent replanning rounds.

We consider two communication scenarios. The first assumes perfect communication. The second models unreliable communication such that robots may not receive or reply to allocation requests, and may not receive intended plans for *sequential*. The probability of a message being successful is modelled as $e^{\frac{-d^2}{2l^2}}$ where d is the distance between robots and l is a length scale. In the Fig. 4(right) experiments, $l =$ 33 m; in the Fig. 5, l is varied from 0 to 100 m.

2) Results: The partial solution presented earlier in Fig. 2 illustrates an example plan output from Dec-SOM. The 5 robots have naturally split up to explore different parts of the world, while focussing particularly on regions with higher goal densities. The individual plans are efficient paths through the allocated goal regions. The executed and planned paths balance between visiting known goal regions and the predicted goals.

The results are shown in Fig. 4, with the reliable communication case shown on the left. We see that Dec-SOM is competitive with *oracle*, meaning it is able to effectively address the challenge of having only an estimate of $\mathcal{Z}$. In 21% of trials, Dec-SOM outperformed *oracle* due to the probabilistic nature of SOM learning and *oracle* not necessarily being optimal. *Selective* performed similarly to Dec-SOM while sending 50% less communication, indicating this is an effective heuristic for reducing communication. *Sequential* performed relatively poorly, which demonstrates the benefits of the iterative negotiations of Dec-SOM. *Greedy* and *closest* performed remarkably poorly since they lack the foresight that Dec-SOM has to move towards high-reward regions. The *no replanning* case performed the worst, which demonstrates the need to adapt to changing information. Dec-SOM was competitive with *centralised*, showing no significant difference in performance, despite the challenges of planning in a decentralised setting.

When the communication was unreliable (Fig. 4 right), similar trends were observed. All methods exhibited a small reduction in performance; however, they were still able to perform reasonably well despite this difficulty. When no communication was allowed, the results were very poor, which demonstrates the need for the robots to communicate and coordinate their plans to successfully complete the task.

Fig. 5 shows a gradual degradation of performance of Dec-SOM as the communication becomes more unreliable. The length scale affects the communication packet loss rate, which is quantified in the Fig. 5 axis labels. With a length scale of 100 m, the communication is near perfect and Dec-SOM is competitive with *oracle*. Even as the communication begins to degrade, the performance remains reasonably high. When the length scale is 6.25 m, there is a clear degradation of performance as the 90% communication loss impairs the robots' ability to negotiate. However, even with only 10% communication, Dec-SOM is able to use this restricted resource to outperform the no communication case.

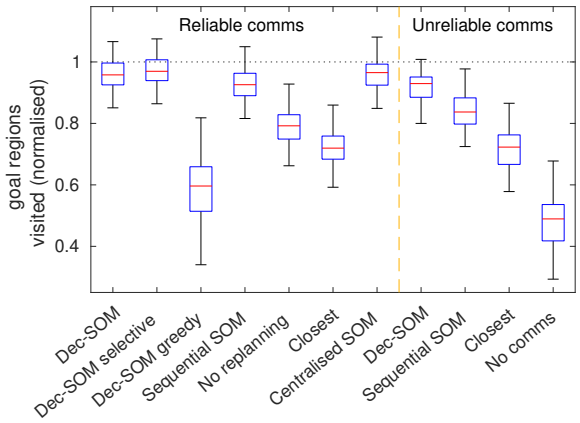

Fig. 6: Results for the infrastructure inspection scenario. 100 trials with different subsets of Gulf of Mexico oil rigs to be inspected (Fig. 1 scenario). The *unreliable comms* scenarios simulated random packet loss following the model described in Sec. V-A.1. Same format as Fig. 4.

B. Infrastructure inspection scenario

We now test our proposed Dec-SOM algorithm in an infrastructure inspection scenario using a real dataset of oil rig locations in the Gulf of Mexico. An illustration of this scenario is provided earlier in Fig. 1. This scenario involves observing a partially-known subset of assets at close range. We define these assets as a set of oil rigs located off the coast of Louisiana, USA [20]. Each oil rig is to be inspected if it meets a criteria that is only discovered online, such as the presence of biofouling or storm damage. Our results here make similar findings to the previous experiments, but in this case we demonstrate the performance of Dec-SOM for a realistic non-uniform distribution of goal regions, a different world estimation model, and a larger team of robots.

1) Experimental setup: As illustrated in Fig. 1, this scenario involves a set of oil rigs to be inspected. The ground-truth location of oil rigs were obtained from [20], for a $200\times100\,\text{km}^2$ region off the coast of Louisiana, USA. This region contains 813 oil rigs, and their locations are known to the robots. A subset of these oil rigs are to be visited by the robots at a range of 5 km. This subset is not initially known to the robots, but rather the need to inspect an oil rig is discovered once a robot is within 25 km of an oil rig. Approximately 25% of the oil rigs are to be inspected, with a different random subset chosen during each trial. A team of 8 robots is simulated, each with a budget of 100 km. During online planning, each world estimate $\tilde{\mathcal{Z}}$ samples oil rigs in unexplored regions with a 0.25 inclusion rate.

2) Results: The results for these experiments are shown in Fig. 6. Similar trends were observed in this scenario as for the random scenario (Fig. 4). As previously, Dec-SOM and *selective* were competitive with *oracle* and *centralised*, and the comparison methods performed significantly worse. *Greedy* performed relatively poorly in this scenario, due to robots getting stuck in areas with sparse unvisited goals, which were avoided by the long-horizon planning of Dec-SOM. Overall, these results demonstrate that Dec-SOM achieves strong performance in a realistic inspection task.

VI. Future Work

We have presented Dec-SOM as an efficient planning algorithm for decentralised multi-robot information gathering. The results demonstrate the usefulness of the algorithm for practical tasks, and motivate several avenues of future work. Straightforward extensions would be to incorporate polygonal goals, non-uniform rewards, heterogeneous sensing, obstacles, and 3D environments, as in other SOM variants [7], [17]. It would be interesting to investigate alternative approaches for reducing the communication requirements, such as by incorporating planning-aware communication [29]. It would also be interesting to exploit probabilistic online estimates for $\tilde{\mathcal{Z}}$ [35] that may explicitly account for future updates to $\tilde{\mathcal{Z}}$, and address cases where the robots have inconsistent beliefs.

References

[1] G. Best and G. A. Hollinger, "Decentralised self-organising maps for the online orienteering problem with neighbourhoods [Extended Abstract]," in *Proc. of Int. Sym. Multi-Robot and Multi-Agent Systems*, 2019, pp. 139–141.

[2] M. Hassan and D. Liu, "Simultaneous area partitioning and allocation for complete coverage by multiple autonomous industrial robots," *Autonomous Robots*, vol. 41, no. 8, pp. 1609–1628, 2017.

[3] S. Sariel, T. Balch, and N. Erdogan, "Naval mine countermeasure missions," *IEEE Robotics Automation Mag.*, vol. 15, no. 1, pp. 45–52, 2008.

[4] F. Sukkar, G. Best, C. Yoo, and R. Fitch, "Multi-robot region-of-interest reconstruction with Dec-MCTS," in *Proc. of IEEE ICRA*, 2019.

[5] J. Faigl and G. A. Hollinger, "Autonomous data collection using a self-organizing map," *IEEE Trans. Neural Networks and Learing Systems*, vol. 29, no. 5, pp. 1703–1715, 2018.

[6] C. Dornhege, A. Kleiner, A. Hertle, and A. Kolling, "Multirobot coverage search in three dimensions," *J. Field Robotics*, vol. 33, no. 4, pp. 537–558, 2016.

[7] G. Best, J. Faigl, and R. Fitch, "Online planning for multi-robot active perception with self-organising maps," *Autonomous Robots*, vol. 42, no. 4, pp. 715–738, 2018.

[8] P. Tokekar, J. V. Hook, D. Mulla, and V. Isler, "Sensor planning for a symbiotic UAV and UGV system for precision agriculture," *IEEE Trans. Robotics*, vol. 32, no. 6, pp. 1498–1511, 2016.

[9] P. A. Plonski and V. Isler, "Approximation algorithms for tours of height-varying view cones," *Int. J. Robotics Research*, vol. 38, no. 2-3, pp. 224–235, 2019.

[10] S. McCammon and G. A. Hollinger, "Topological hotspot identification for informative path planning with a marine robot," in *Proc. of IEEE ICRA*, 2018.

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

[12] G. Best, O. Cliff, T. Patten, R. R. Mettu, and R. Fitch, "Dec-MCTS: Decentralized planning for multi-robot active perception," *Int. J. Robotics Research*, vol. 38, no. 2-3, pp. 316–337, 2019.

[13] G. A. Hollinger, S. Yerramalli, S. Singh, U. Mitra, and G. Sukhatme, "Distributed coordination and data fusion for underwater search," in *Proc. of IEEE ICRA*, 2011, pp. 349–355.

[14] C. Rizzo, D. Tardioli, D. Sicignano, L. Riazuelo, J. L. Villarroel, and L. Montano, "Signal-based deployment planning for robot teams in tunnel-like fading environments," *Int. J. Robotics Research*, vol. 32, no. 12, pp. 1381–1397, 2013.

[15] T. Kohonen, "Essentials of the self-organizing map," *Neural Networks*, vol. 37, pp. 52–65, 2013.

[16] B. Angéniol, G. D. L. C. Vaubois, and J.-Y. Le Texier, "Self-organizing feature maps and the travelling salesman problem," *Neural Networks*, vol. 1, no. 4, pp. 289–293, 1988.

[17] J. Faigl, "Approximate solution of the multiple watchman routes problem with restricted visibility range," *IEEE Trans. Neural Networks*, vol. 21, no. 10, pp. 1668–1679, 2010.

[18] J. Faigl, R. Pěnička, and G. Best, "Self-organizing map-based solution for the orienteering problem with neighborhoods," in *Proc. of IEEE SMC*, 2016, pp. 1315–1321.

[19] S. Somhom, A. Modares, and T. Enkawa, "Competition-based neural network for the multiple travelling salesmen problem with minmax objective," *Computers & Operations Research*, vol. 26, no. 4, pp. 395–407, 1999.

[20] Bureau of Ocean Energy Management. BOEM data center: Platform structures. [Online]. Available: http://www.data.boem.gov [Accessed: 1-Feb-2019]

[21] G. A. Hollinger and G. S. Sukhatme, "Sampling-based robotic information gathering algorithms," *Int. J. Robotics Research*, vol. 33, no. 9, pp. 1271–1287, 2014.

[22] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein, "The complexity of decentralized control of Markov decision processes," *Mathematics of Operations Research*, vol. 27, no. 4, pp. 819–840, 2002.

[23] M. Corah and N. Michael, "Distributed matroid-constrained submodular maximization for multi-robot exploration: Theory and practice," *Autonomous Robots*, vol. 43, no. 2, pp. 485–501, 2019.

[24] W. Burgard, M. Moors, C. Stachniss, and F. E. Schneider, "Coordinated multi-robot exploration," *IEEE Trans. Robotics*, vol. 21, no. 3, pp. 376–386, 2005.

[25] R. Zlot, A. Stentz, M. B. Dias, and S. Thayer, "Multi-robot exploration controlled by a market economy," in *Proc. of IEEE ICRA*, vol. 3, 2002, pp. 3016–3023.

[26] H.-L. Choi, L. Brunet, and J. P. How, "Consensus-based decentralized auctions for robust task allocation," *IEEE Trans. Robotics*, vol. 25, no. 4, pp. 912–926, 2009.

[27] L. Liu and D. A. Shell, "Large-scale multi-robot task allocation via dynamic partitioning and distribution," *Autonomous Robots*, vol. 33, no. 3, pp. 291–307, 2012.

[28] M. Otte, M. Kuhlman, and D. Sofge, "Multi-robot task allocation with auctions in harsh communication environments," in *Proc. of International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, 2017.

[29] G. Best, M. Forrai, R. R. Mettu, and R. Fitch, "Planning-aware communication for decentralised multi-robot coordination," in *Proc. of IEEE ICRA*, 2018, pp. 1050–1057.

[30] A. Kassir, R. Fitch, and S. Sukkarieh, "Communication-efficient motion coordination and data fusion in information gathering teams," in *Proc. of IEEE/RSJ IROS*, 2016, pp. 5258–5265.

[31] T. D. Hämäläinen, "Parallel implementations of self-organizing maps," in *Self-Organizing Neural Networks*. Springer, 2002, pp. 245–278.

[32] M. Jayaratne, D. de Silva, and D. Alahakoon, "Unsupervised machine learning based scalable fusion for active perception," *IEEE Trans. Automation Science Engineering*, vol. 16, no. 4, pp. 1653–1663, 2019.

[33] A. Gunawan, H. C. Lau, and P. Vansteenwegen, "Orienteering problem: A survey of recent variants, solution approaches and applications," *Eur. J. Operations Research*, vol. 255, no. 2, pp. 315–332, 2016.

[34] S. D. Bopardikar, S. L. Smith, and F. Bullo, "On dynamic vehicle routing with time constraints," *IEEE Trans. Robotics*, vol. 30, no. 6, pp. 1524–1532, 2014.

[35] M. Saroya, G. Best, and G. A. Hollinger, "Online exploration of tunnel networks leveraging topological CNN-based world predictions," in *Proc. of IEEE/RSJ IROS*, 2020.