2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Daejeon Convention Center
October 9-14, 2016, Daejeon, Korea

Multi-Robot Path Planning for Budgeted Active Perception with Self-Organising Maps

Graeme Best[1], Jan Faigl[2] and Robert Fitch[1]

***Abstract*— We propose a self-organising map (SOM) algorithm as a solution to a new multi-goal path planning problem for active perception and data collection tasks. We optimise paths for a multi-robot team that aims to maximally observe a set of nodes in the environment. The selected nodes are observed by visiting associated viewpoint regions defined by a sensor model. The key problem characteristics are that the viewpoint regions are overlapping polygonal continuous regions, each node has an observation reward, and the robots are constrained by travel budgets. The SOM algorithm jointly selects and allocates nodes to the robots and finds favourable sequences of sensing locations. The algorithm has polynomial-bounded runtime independent of the number of robots. We demonstrate feasibility for the active perception task of observing a set of 3D objects. The viewpoint regions consider sensing ranges and self-occlusions, and the rewards are measured as discriminability in the ensemble of shape functions feature space. Simulations were performed using a 3D point cloud dataset from a real robot in a large outdoor environment. Our results show the proposed methods enable multi-robot planning for budgeted active perception tasks with continuous sets of candidate viewpoints and long planning horizons.**

I. Introduction

Mobile robots use their sensors and perception algorithms to understand their surrounding environment. Of fundamental interest are object recognition, classification and model generation tasks, which require understanding properties such as the pose, segmentation, class and identity of a set of objects in an environment [1], [2], [3]. The informativeness of observations, and therefore the performance of perception algorithms, can be improved by judiciously selecting observation locations [4]. Performance can be significantly improved by using longer planning horizons [5], [6], [7], jointly planning for multiple robots [8], [9], [10], [11] and considering larger sets of candidate sensing locations. However, current planning algorithms with these properties are often too computationally expensive for practical use in large scale and more complex active perception tasks; we propose a self-organising map algorithm as a solution to bridge this gap.

*This work was supported by the Australian Centre for Field Robotics; the NSW Government; the Australian Research Council's Discovery Project funding scheme (No. DP140104203); and the Faculty of Engineering & Information Technologies, The University of Sydney, under the Faculty Research Cluster Program. The work of Jan Faigl was supported by the Czech Science Foundation (GAČR) under research project No. 15-09600Y.

[1]G. Best and R. Fitch are with the Australian Centre for Field Robotics (ACFR), The University of Sydney, Sydney, Australia. {g.best,rfitch}@acfr.usyd.edu.au

[2]J. Faigl is with the dept. of Computer Science, Czech Technical University in Prague, Technická 2, 166 27 Prague, Czech Republic faiglj@fel.cvut.cz

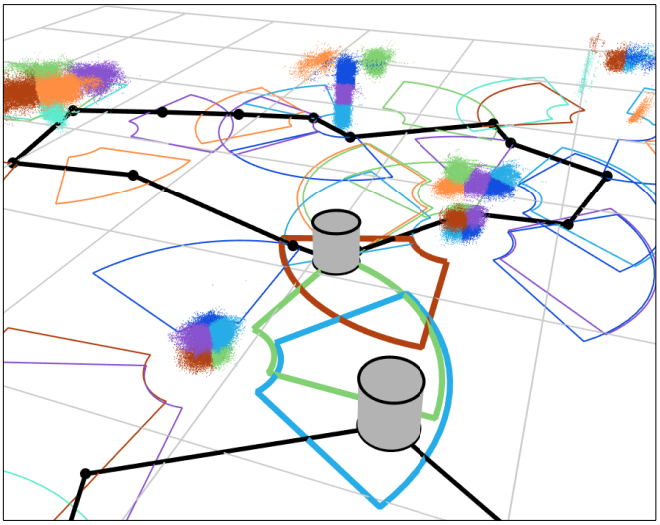


Fig. 1. Illustration of the motivating active perception problem. Each object segment (point clouds) is observed by visiting the viewpoint regions (circle segments). Grey cylinders represent positions of two robots. The currently visited viewpoint regions are drawn in bold. Black lines represent the path plans. The goal is to collectively maximise the weighted sum of viewpoint regions visited by the robots. This scene is part of the environment in Fig. 2.

Current approaches for active perception typically predict the value of expected observations as a result of moving the robot to candidate viewpoints [1], [2], [3]. For complex sensor models, these calculations can be computationally expensive, which therefore restricts the capabilities of planning algorithms. Instead, we focus on planning paths for perception tasks where informative parts of the objects in the environment have been extracted. The inverse sensor model defines a discrete set of overlapping continuous viewpoint regions, with associated rewards, where each part can be observed. Figures 1 and 2 illustrate an example outdoor environment with a collection of objects observed by a 3D laser scanner. The path planning problem is to optimise the rewards gained by visiting these desirable viewpoint regions. This new formulation enables the planner to consider a continuous space of candidate viewpoints, longer horizon planning, and multi-robot scenarios.

This active perception formulation describes a multi-goal path planning problem as a generalisation of the travelling salesman problem (TSP). The prize-collecting TSP with neighbourhoods (PC-TSPN) is a closely related TSP variant that has recently been solved and applied to data collection in sensor network applications [12]. In the PC-TSPN, the objective is to plan the path of a robot that maximally selects and visits a set of disks, where the objective function is defined as the sum of the path length and the rewards for

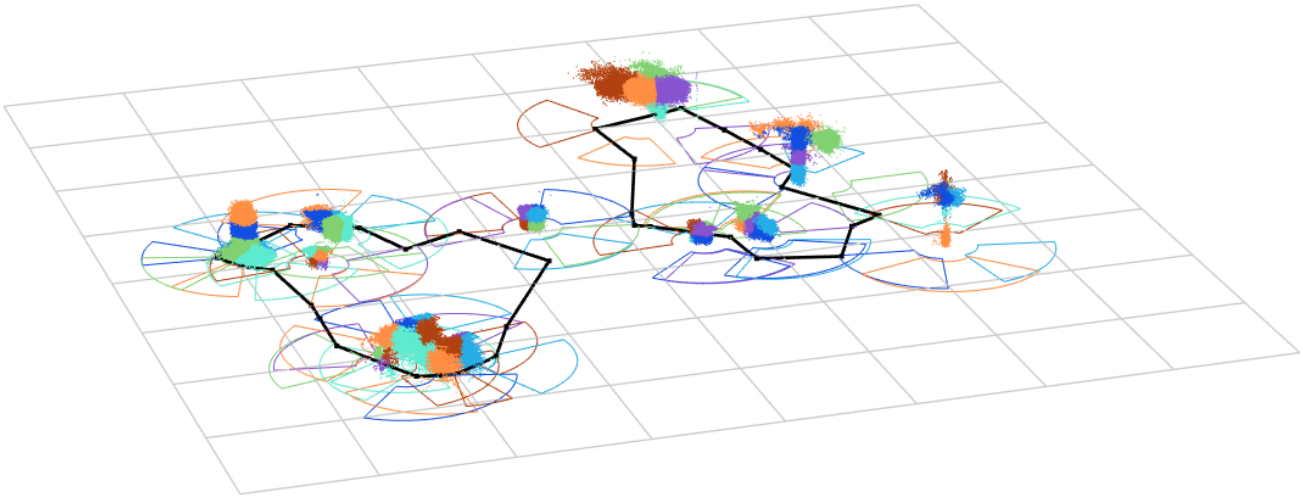


Fig. 2. An example environment, object parts, viewpoint regions and solution paths for two robots (same as Fig. 1). The 3D point cloud was generated by a real robot moving around an environment consisting of trees, tables, chairs, bins and a motorbike. Almost all object parts are observed along the planned paths, with some skipped due to the travel budget constraints. The underlying grid has 5 m spacing.

visiting each disk. This objective function has convenient algorithmic properties, however, it is unclear how to balance the trade-off between path lengths and rewards when applied to real problems. Instead, we develop a new formulation which directly optimises the observation rewards, while the path length is constrained by a maximum travel time budget defined by constraints of the application.

The considered problem is NP-hard, which can be shown by a reduction from the orienteering problem [13], and therefore we turn to heuristic solutions. In particular, we consider an extension of the self-organising map (SOM) for TSP. SOM is a two-layered neural network accompanied by an unsupervised learning procedure that has been applied to the traditional TSP by several authors, e.g., [14]. Although SOM for the TSP does not compete with the best known combinatorial heuristics for the conventional TSP [15], it provides a significant advantage in problems where it is required to determine observation locations. This is particularly important in the TSPN [16] and the orienteering problem with neighbourhoods [17] where it implicitly selects the sensing locations within the continuous neighbourhoods.

Jointly optimising the selection and sequence of nodes to observe, along with finding favourable viewpoints within sensing regions, can greatly reduce the path distance by avoiding unnecessary travel. Therefore, we consider the original idea of the SOM-based data collection planning introduced in [12] for our constrained problem with limited travel budgets. Moreover, we also generalise the approach to path planning for multi-robot teams. The algorithm jointly plans for multiple robots simultaneously by optimising the allocation of nodes to robots. Therefore, our approach does not require predefined or explicit partitioning of the environment. The algorithm is robust to the compounded complexity of optimising multiple robots since the runtime is independent of the number of robots.

In addition to theoretical analysis, we also perform simulations of several random environments and active perception tasks using a 3D point cloud dataset [18] and a realistic observation model using ensemble of shape functions [19] descriptors. The results highlight advantages of the algorithm for addressing the multi-robot, non-uniform reward, constrained budget and polygonal region characteristics of the problem. The active perception experiments show the feasibility in practice for planning long-horizon multi-robot active perception tasks.

II. Related Work

Our problem formulation and approach is motivated by the work of Faigl and Hollinger [12] for the PC-TSPN problem using an SOM algorithm. They applied the PC-TSPN to a data collection problem that requires communicating with a subset of an underwater sensor network. We generalise this problem formulation and algorithmic approach to be more suitable for our active perception formulation. In particular, we include multiple robots, polygonal goal regions [16], budget constraints and non-uniform observation rewards.

The generalised-TSP (GTSP) is a closely related TSP variant that requires visiting at least one node in every discrete set of nodes. The GTSP has been applied to robotic path planning problems for mapping [10] and mobile refuelling [20] tasks, which are solved using a transformation to the standard TSP. Related graph-based robot path planning algorithms include branch and bound [21] and sweep planes [22]. These formulations restrict the search to discrete points and the computation time increases with the number of points. In contrast, a set of continuous regions are efficiently searched by our proposed algorithm, and the runtime does not increase with the area of each region.

The m-TSP generalises the TSP to multiple agents, which requires assigning nodes to agents and finding a path for each agent. There are several variations of the m-TSP with many different approaches [23]. SOM-based approaches have been used for solving the minmax m-TSP, where the objective is to minimise the path of the longest agent. The approach

creates an individual network for each agent, and the adaptation prefers neurons from the currently shortest tours when allocating tasks to the individual agents [24]. A similar idea has been considered for multi-agent coverage of a polygonal world with obstacles [16]. However, these problems do not consider budget constraints or selecting subsets of nodes.

In multi-robot planning more generally, there are a variety of related problems. Coverage tasks are most similar to our problem, which require a team of robots to collectively observe every location in an environment [25]. Target tracking and search problems require using the sensing capabilities of multiple robots to locate and maintain contact with targets [8], [10], [11]. Mapping tasks require adaptively exploring unobserved regions [10]. Persistent monitoring tasks require sensing an environment to maintain a belief of a spatio-temporal process [9]. Although our problem is similar to these, in that we require dividing the workload and finding paths for multiple robots, our objectives are different; therefore, we require a new algorithmic approach.

Traditional active perception for object recognition type problems use vision sensing [4], although there has been a recent push towards modalities with depth information, such as laser 3D range sensing [18], RGB-D sensing [1], [3], and thermal depth sensing [26]. In most work, the planning is single-step [1], [2], [26], which is reasonable in small environments where it is assumed that previous actions do not affect the cost of future actions. Scaling the problem up to larger environments results in location-dependent action costs, and therefore performance is significantly improved by planning sequences of viewpoints over longer planning horizons. Some approaches have been proposed for planning sequences of locations [5], [6], [7], but the formulations have been limited to restricted cases, such as a single object or a constrained action space. Little attention has been given to continuous candidate viewpoint regions or multi-robot planning for active perception in object recognition tasks.

III. Problem Formulation

This section formally defines the budgeted multi-robot active perception problem. The objective is to plan the paths for a team of robots such that they maximally observe a set of nodes in the environment with varying rewards. Each robot has an associated travel speed and maximum travel budget. Each node may be observed by visiting any point in its associated viewpoint region represented as a polygon.

A. Multi-Robot Team

The problem involves a team of R robots $\mathcal{R} = \{r^1, r^2, ..., r^R\}$. The trajectory of each robot r^i is defined as a sequence of waypoints $X^i = (x^i_1, x^i_2, x^i_3, ...)$, where each waypoint is a position within a free space environment $x^i_j \in \mathbb{R}^2$. Each robot r^i moves along a straight line between waypoints at a constant speed s^i, which may be different for each robot. The cost of each robot's path $c^i \geq 0$ is the time taken to travel through the sequence of waypoints X^i. Each robot has a cost budget $b^i > 0$, and a set of robot paths $\{X^i\}$ is deemed to be feasible if every robot meets its budget constraint, i.e., $c^i \leq b^i, \forall r^i \in \mathcal{R}$. Although we consider the problem where each robot's path is a tour with an unconstrained start location, the algorithm can readily be extended for open loop or constrained paths.

B. Viewpoint Regions and Rewards

The robots aim to observe a set of N nodes $\mathcal{N} = \{n^1, n^2, ..., n^N\}$ at different locations within the environment. Every node has a weight $w^k > 0$ that defines the reward for observing the node. Each node n^k has a continuous set of viewpoints $\mathcal{Z}^k$ defined as all points on and within a simple polygon. The robot observes a node if any waypoint of the robot's path is within the viewpoint region, i.e., $\exists x^i_j \in X^i : x^i_j \in \mathcal{Z}^k$. The binary indicator variable $o^k \in \{0, 1\}$ for each node n^k is 1 if the node is observed by any robot and 0 otherwise. All robots sense continuously along their paths, which can be taken into account in the above definition by adding additional waypoints along a path at no extra cost. Although we assume the regions are the same for each robot, the algorithm can easily be extended to robot-dependent observations.

C. Problem Statement

The optimisation problem is to plan the locations of waypoints for each robot and the sequence the waypoints are visited X^i, such that all budget constraints are met and the sum of the observation rewards for the nodes is maximised. More formally, we aim to:

$$\text{maximise} \sum_{n^k \in \mathcal{N}} o^k w^k,$$
$$\text{s.t. } c^i \leq b^i, \ \forall r^i \in \mathcal{R}.$$

D. NP-hardness

The problem is NP-hard and a reduction from the orienteering problem [13] with Euclidean costs can readily be shown by setting the number of robots R to 1 and the viewpoint sets $\{Z^k\}$ as singleton. This result motivates the development of a heuristic algorithm to approximately solve the problem in polynomial time.

IV. Self-Organising Map Algorithm

Self-organising map algorithms aim to give a lower-dimensional representation of an input space, while preserving a given topological graph-based structure of the representation. For our problem the input space is the set of viewpoint regions in the environment, and the algorithm aims to find a set of rings representing robot paths that best fits this input space. The learning procedure is competitive in that each viewpoint region is presented one at a time, and each waypoint competes to be the *winner* for representing that region. A winner waypoint moves towards that region, and neighbours of the winner in the graph topology will also move towards the region by a decreasing distance. This process is repeated for a fixed number of learning epochs, when convergence of the paths to a stable state is guaranteed.

This section details the proposed SOM learning procedure for our problem formulation, that includes addressing

Algorithm 1 Self-organising map algorithm.

Input: robot speeds $\{s^i\}$ and budgets $\{b^i\}$,
a set of nodes $\{n^k\}$ with associated
viewpoint regions $\{\mathcal{Z}^k\}$ and rewards $\{w^k\}$
Output: planned path for each robot $\{X^i\}^*$

1: $X^i \leftarrow$ circle around arbitrary node n^i, $\forall r^i \in \mathcal{R}$
2: $\mathcal{N}' \leftarrow$ duplicate $n^k \in \mathcal{N}$ by factor $w^k/\text{GCD}(\{w^k\})$
3: $\sigma \leftarrow 1$; $i \leftarrow 1$ ▷ Adaptation parameters
4: **while** not converged **do**
5: $perm \leftarrow$ random permutation of $\{n^k\}$
6: **for each** $n^k \in \mathcal{N}'$, in order $perm$ **do**
7: **for each** $r^i \in \mathcal{R}$ **do**
8: $X^{i\prime} \leftarrow \text{ADAPTATION}(X^i, \mathcal{Z}^k, \sigma)$
9: $c^{i\prime} \leftarrow$ travel time of path $X^{i\prime}$ at speed s^i
10: $r^i \leftarrow \underset{r^i \in \mathcal{R}, c^{i\prime} \leq b^i}{\text{argmin}} \left(\frac{c^{i\prime}}{b^i}\right)$ ▷ Robot selection
11: $X^i \leftarrow X^{i\prime}$ ▷ Update selected robot
12: $\{x^i\} \leftarrow$ regeneration of $\{x^i\}$
13: $F \leftarrow \sum_{n^k \in \mathcal{N}} o^k w^k$ ▷ Evaluate objective
14: **if** $F > F^*$ **then** $\{X^i\}^* \leftarrow \{X^i\}$ ▷ Save best plan
15: $\sigma \leftarrow (1 - 0.001i)\sigma$; $i \leftarrow i + 1$

Algorithm 2 Adaptation step of the SOM algorithm

function ADAPTATION
Input: path X^i of robot r^i,
viewpoint region $\mathcal{Z}^k$ of node n^k,
adaptation parameter σ
Output: an adapted path $X^{i\prime}$ for robot r^i

1: $x_\text{w} \leftarrow$ closest waypoint in X^i to $\mathcal{Z}^k$
2: $z_\text{w} \leftarrow$ closest point in $\mathcal{Z}^k$ to x_w
3: $d_\text{w} \leftarrow \|z_\text{w} - x_\text{w}\|$
4: $x_\text{e} \leftarrow$ closest point on edges of X^i to $\mathcal{Z}^k$
5: $z_\text{e} \leftarrow$ closest point in $\mathcal{Z}^k$ to x_e
6: $d_\text{e} \leftarrow \|z_\text{e} - x_\text{e}\|$
7: **if** $x_\text{w} \in \mathcal{Z}^k \vee d_\text{w} \leq d_\text{e}$ **then** ▷ Winner selection
8: $x^\star \leftarrow x_\text{w}$; $z^\star \leftarrow z_\text{w}$ ▷ Select waypoint
9: **else**
10: $X^i \leftarrow$ insert x_e into X^i along edge
11: $x^\star \leftarrow x_\text{e}$; $z^\star \leftarrow z_\text{e}$ ▷ Select edge
12: **for each** $x^i_j \in X^i$ **do**
13: ▷ Adapt waypoints in neighbourhood of $x^\star$
14: $l \leftarrow$ cardinal distance from $x^\star$ to x^i_j
15: $x^{i\prime}_j \leftarrow$ move x^i_j towards $z^\star$ by factor e^{-l^2/σ^2}

non-uniform observation rewards, node selection satisfying budget constraints and multi-robot task allocation to nodes. Pseudocode for the algorithm is shown in Alg. 1.

A. Graph Topology

The graph topology for the SOM is a set of R rings that directly represent the robot paths. Each of these rings will transform over time according to the following learning procedure. Each ring is initialised as a small circle (consisting of N/R waypoints) around the centre of a unique arbitrary node. This initialisation is reasonable since the paths quickly spread over the input space during the first learning epochs.

B. Viewpoint Rewards

Each node has an associated reward for being visited. To ensure that the learning procedure favours visiting the higher reward viewpoint regions, each node is duplicated according to its reward. The node n^k is duplicated by a factor of w^k divided by the greatest common divisor of the set of rewards $\text{GCD}(\{w^k\})$. Computation time is dependent on these factors, and therefore it may be beneficial to round the rewards to the nearest multiple of a number greater than $\text{GCD}(\{w^k\})$.

The motivation for this approach is that high-reward regions will be trialled more often in each learning epoch. This increases the likelihood of a robot transforming towards the higher weighted nodes and decreases the likelihood of the node not being selected due to budget constraints.

C. Learning Epochs

In each learning epoch (iteration of line 4 loop), each node is considered one at a time, and one robot is selected to transform its path towards each viewpoint region, if it meets its budget constraint. At the end of each learning epoch, any unnecessary waypoints are removed before starting the next learning epoch.

Permute the nodes: At the start of each epoch, the nodes are permuted in a random order which will determine the order that they are considered (lines 5-6). This ensures the algorithm is less sensitive to the ordering and the initial conditions, and more likely to escape from local optima.

Winner selection and adaptation: The key steps in the SOM algorithm are the winner waypoint selection and the adaptation of the rings (Alg. 2). For every node, this is performed for each robot, but then in the following step a robot allocation policy ensures only one of the robots gets updated for each node. The winner waypoint selection is performed by considering all waypoints and edges in the current robot path X^i. The existing waypoint or a point along one of the existing edges that is closest to any point within the viewpoint region $\mathcal{Z}^k$ is considered as the winner (line 8 or 11). If the winner is a point along an edge, then a new waypoint is inserted into the path at this point (line 10).

The winner waypoint $x^\star$ is then moved to the closest point $z^\star$ in $\mathcal{Z}^k$. If $z^\star$ is on the edge of the polygon it is moved slightly towards the centre. The cardinal distance (number of hops) in the ring from $x^\star$ to every other existing waypoint is denoted l. Each waypoint in X^i is moved some fraction towards $z^\star$ (line 15). This fraction is determined by the neighbourhood function e^{-l^2/σ^2} and set to zero if below a threshold. Waypoints with low cardinal distance to $x^\star$ will move further towards $z^\star$ than other waypoints.

Robot-node allocation: After adapting each path towards the viewpoint region $\mathcal{Z}^k$, the algorithm then only allocates one (or none) of the robots to the node and only this robot keeps their adapted path. The selection is performed by greedily selecting the robot that has used the least fraction of

its budget after performing the adaptation (Alg. 1, line 10). Note that this allocation of robots to nodes are often re-modified in later learning epochs. If no robot meets their travel budget then no paths are adapted. This allocation approach is motivated by the observation that in most cases an optimal solution should have each robot using approximately all of its travel budget. We wish to divide the work evenly between the robots as the learning progresses towards the final solution, such that a natural partitioning is found between the robots.

Ring regeneration: At the end of each learning epoch, waypoints of the paths that are no longer useful are removed. A waypoint is useful if it is still within one of the viewpoint regions. If multiple waypoints are within a viewpoint region then only one random waypoint is selected to remain.

Adaptation parameter: The σ parameter of the neighbourhood function decreases after each epoch, meaning the neighbours get less attracted to each other as the learning progresses. This ensures convergence in a constant time.

V. Analysis of Algorithm

This section gives a theoretical analysis of the algorithm's runtime complexity and convergence, and then empirical analysis of the behaviour of the algorithm for various random environments. Further experiments are shown later in Sec. VI that focus on active perception of 3D point clouds.

A. Theoretical Analysis

The time complexity for each learning epoch is bounded by $\mathcal{O}(NN') = \mathcal{O}(N^2 \frac{\text{MAX}(\{w^k\})}{\text{GCD}(\{w^k\})})$, where N is the number of viewpoint regions and N' is the number of viewpoint regions after duplication to take into account the rewards $\{w^k\}$. This is because the runtime for each iteration of the line 6 loop is linear in the current total number of waypoints for all robots, which is bounded by $\mathcal{O}(N)$. The number of duplications per node N'/N is bounded by the ratio of the maximum reward $\text{MAX}(\{w^k\})$ to the greatest common divisor $\text{GCD}(\{w^k\})$.

The algorithm continues until convergence is reached, which is guaranteed to occur in a constant number of learning epochs. This is since the decreasing neighbourhood function will eventually cause there to be no adaptations occurring. Therefore, the time complexity of the algorithm is polynomial, bounded by $\mathcal{O}(N^2 \frac{\text{MAX}(\{w^k\})}{\text{GCD}(\{w^k\})})$. Interestingly, this bound does not depend on the number of robots R, since each viewpoint region is allocated to a maximum of one robot during each learning epoch, and therefore the maximum total number of waypoints is independent of R. We also note the computation time can be reduced by rounding the rewards to multiples of a divisor greater than $\text{GCD}(\{w^k\})$.

Self-organising map algorithms, including ours, are stochastic learning procedures that can guarantee convergence in polynomial time, but unfortunately cannot guarantee optimality in finite time. These algorithms therefore are heuristic algorithms for giving approximate solutions to NP-hard problems in polynomial time. We also note that the algorithm is anytime, such that the algorithm can be halted early, since all intermediate solutions are feasible solutions.

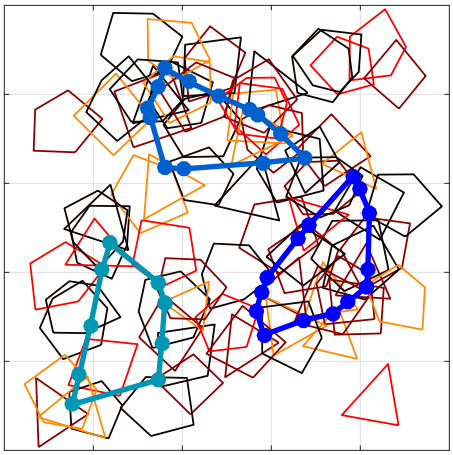


Fig. 3. Example path plans for three robots (blue) through a set of random viewpoint regions weighted from 1 (black) to 4 (orange). The robots visit a weighted sum of 155 viewpoint regions out of a maximum 170. Each robot has a budget of 1000 and speed 1.

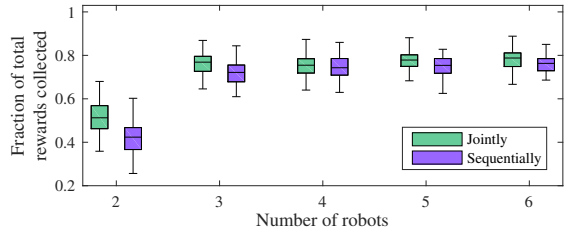
B. Empirical Analysis

Simulated experiments were performed to analyse the behaviour of the algorithm under various conditions. Since the problem is new, we do not have algorithms for direct comparison[1]. Therefore, we compare to restricted versions of our algorithm with some aspects removed. We compare i) planning using the joint multi-robot optimisation compared to sequential optimisation, ii) planning with and without the viewpoint rewards, and iii) planning with the viewpoint polygons compared to single points. The algorithm plans paths through 100 random environments consisting of random sets of polygons. An example environment is illustrated in Fig. 3.

The parameters are as follows, except where varied for specific experiments. The environments are a continuous 1000×1000 space. There are 80 polygons with random centre points and from 3 to 6 vertices spaced at equal angles around the centre. The distance from the centre to each vertex is random between 40 and 120. Rewards are exponentially distributed between 1 and 4 and rounded to the nearest integer, such that few regions have high rewards. There are 3 robots with budgets 800 and speeds 1. In all cases, convergence was reached in 70 epochs. The same sample environments are used for each pair of methods and a single-tailed paired t-test was performed for each comparison.

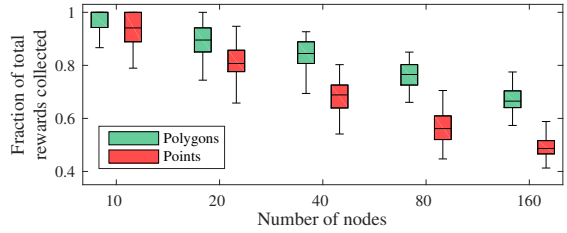
1) Multiple robots: Fig. 4a shows the rewards collected by planning using the proposed method, which jointly optimises multiple robots, compared to planning for the robots sequentially. The sequential method performs the SOM algorithm for each robot one at a time, with each robot ignoring the nodes selected by the previous robots. The two methods were compared for 2 to 6 robots, where the budgets were uniform and summed to 2400. The simulations show the proposed approach has the best performance in all cases, and these results were statistically significant ($p < 0.01$) in all cases except $R = 4$. The largest improvements were for planning

[1]In [17] we empirically compare a related SOM algorithm to state-of-the-art solvers for the single-agent orienteering problem.

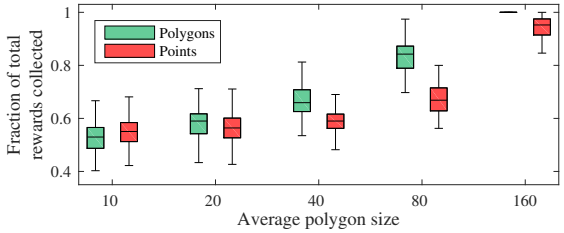


(a) Jointly planning for all robots in the team following the proposed method compared to sequentially planning each robot.

(b) Planning taking into account the rewards following the proposed method compared to planning assuming uniform rewards.



(c) Planning taking into account the viewpoint regions compared to planning for only the centroid of the polygons.



(d) Planning taking into account the viewpoint regions compared to planning for only the centroid of the polygons.

Fig. 4. Simulation results for random environments under various scenarios and comparison methods. Vertical axes shows performance as the ratio of the achieved weighted sum of nodes visited to the weighted sum of all nodes in the environment. Box plots show lower bound, lower quartile, median, upper quartile and upper bound for 100 sample environments.

with a smaller number of robots, because in these cases the performance is greatly influenced by effective partitioning of the workspace between the robots, which can be more effectively optimised when planning for all robots jointly.

2) Observation rewards: Fig. 4b shows the simulation results for planning using the proposed duplication approach compared to assuming all rewards are uniform. The rewards are exponentially distributed between 1 and $\bar{w}$ with lower rewards more likely, and $\bar{w}$ varied from 2 to 32. In all cases, planning using the proposed approach improved the performance, and these results were statistically significant ($p < 0.01$). Greater improvements were achieved when the maximum reward was large since the proposed approach is more likely to select the nodes with a large reward.

3) Viewpoint regions: We analyse the value of the proposed planning with continuous polygonal viewpoint regions compared to planning with single points at the region centres. Fig. 4c compares these two methods with a varying number of nodes and Fig. 4d has a varying average polygon size. The proposed planner outperformed the single point planner for all number of nodes and when the polygon size ≥ 20, and these results were statistically significant ($p < 0.01$). When the polygon size was very small (10) it was sufficient to plan by approximating the polygons as single points. The proposed approach achieved greater improvements when the number of nodes and the size of the polygons were large. In these cases, the algorithm can more effectively take advantage of being able to optimise the waypoint locations

Computation time: The algorithm was implemented in MATLAB and the simulations were performed on a standard desktop computer with an Intel i7 processor on a single core. The runtime varied from 0.5 s to 30 s depending on the scenario. The trends agreed with the theoretical analysis such that runtime increased with the number of nodes and maximum weight. Runtime increased with the number of robots, possibly due to the multi-robot planning achieving greater performance and therefore a larger number of nodes, however this increase was sublinear. Runtime was dominated ($\approx 70\%$) by the winner selection and the waypoint usefulness evaluation, since these geometric computations are relatively expensive. Our implementation has not been thoroughly optimised, since our primary focus was on validating the feasibility of the approach. Therefore, the runtime can be significantly improved by the implementation, as well as by using approximations, such as decreasing the number of polygon vertices or approximating as discs.

VI. Active Perception of 3D Point Clouds

Our primary motivation for the proposed problem formulation and SOM algorithm is active perception tasks that aim to observe a set of object parts in a large environment. These problems rely on prior observations or a predefined belief of the environment, that may have come from a coarse scan with noisy sensors. The aim is now to perform a more informative or complete scan of the environment, and this process may be repeated. In this section, we demonstrate how the algorithm can be applied to this class of active perception tasks.

We consider example scenarios using three variations of an outdoor scene from a 3D point cloud dataset. The data was recorded with a Velodyne laser scanner mounted on a robot. Observations were made from several locations and fused together. The dataset was initially recorded for testing object classification algorithms [18]. The three scenes consist of 12, 15 and 18 objects spread around a 40 m$\times$40 m environment, including trees, tables, chairs, bins and a motorbike.

The environment is represented by a set of parts in a 3D point cloud with associated viewpoints and rewards. Examples of the segmentation and viewpoint regions are

shown earlier in Figs. 1 and 2. The point cloud processing is summarised as follows, with further details below: i) oversegment the environment into parts, ii) estimate self-occlusion free viewpoint regions for each part, and iii) defining the rewards as the discriminability between parts. We emphasise that the proposed SOM algorithm is not limited to this point cloud processing method, but rather the processing can be adapted to suit the requirements of a perception task.

A. 3D Point Cloud Processing

The point cloud of the environment is segmented into parts by first removing the ground plane and then segmented into objects using region growing. Each object is then oversegmented into 5 parts using k-means clustering.

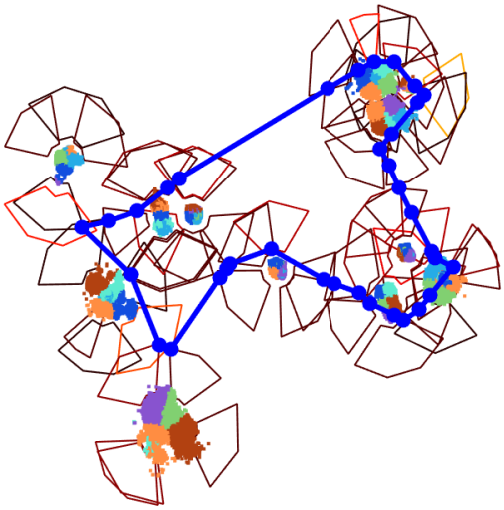
The viewpoint region for each part considers the sensing range and object self-occlusions. This is achieved by comparing the set of vectors from all points within a part to all points in other parts of the object. A vector is considered an occlusion if the vertical angle is within $-\pi/8$ and $\pi/8$. The viewing angle range is defined as the middle third of the largest window of horizontal angles that contains less than 10% of the occlusion vectors. The useful sensing range is defined as 1 to $4\,\mathrm{m}$. The viewpoint region is defined as the intersection of the horizontal viewing angle range and the sensing range, measured relative to the part's centroid. This region is approximated by a polygon with 6 to 8 vertices.

We define the rewards as the discriminability of each part in a feature space. To measure the discriminability, we perform feature extraction for each part, calculate the distance to all other parts in feature space, and normalise for each object. We compare each part to all other parts in the environment, although alternatively each part could be compared to an object library. For the feature extraction, we use the ensemble of shape functions [19], which is commonly used for object classification tasks [3]. Discriminability is measured as the exponential of the sum of Mahalanobis distances in the feature space between each part and every other part. Each object is equally important, and therefore the sum of rewards for each object is normalised to 10. Each reward is rounded to the nearest integer. The rewards for the datasets ranged from 1 to 10 with 1 or 2 more likely.

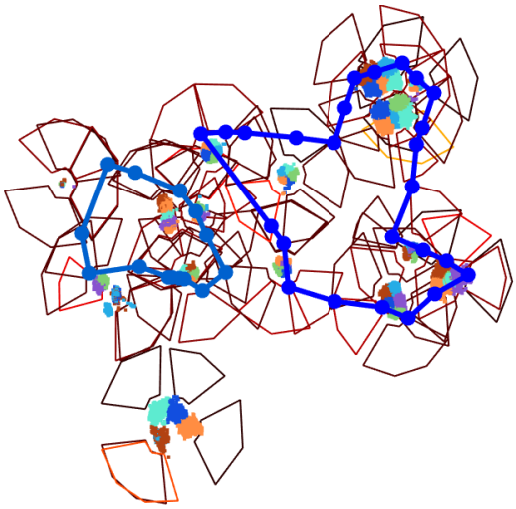
B. Experiments

We analyse three example scenarios illustrated in Fig. 5, for three environments with varying clutter. In the scenarios we plan for: i) a single robot in the low clutter environment, ii) two robots in medium clutter, where one robot has double the budget, and iii) three robots in high clutter, where the robots have speeds 2, 1.5 and 1. Planning was repeated 100 times each to measure the planning consistency.

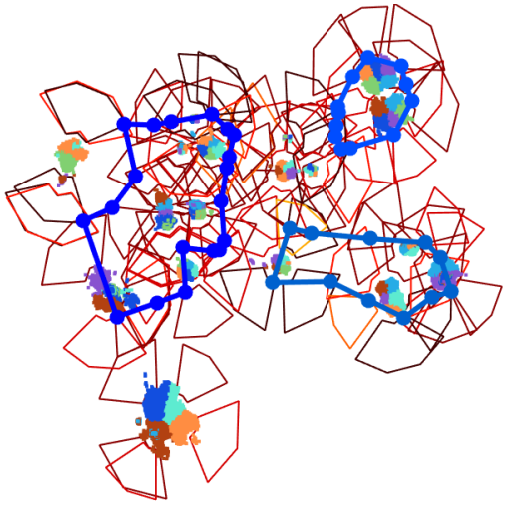
In the first scenario, the robot observed a weighted sum of 132 nodes, averaged over 100 trials, out of the maximum possible 151 nodes. The performance was consistent over the 100 trials, with a standard deviation of 2.13 weighted nodes. The worst plan had 124 and the best had 135. The average runtime was $6\,\mathrm{s}$ with standard deviation $0.1\,\mathrm{s}$. An example solution is shown in Fig. 5a. All objects have at least one of



(a) Low clutter (12 object) environment with a single robot.



(b) Medium clutter (15 object) environment with two robots. Robot on right has $2\times$ budget.



(c) High clutter (18 object) environment with three robots. Robot on left has $2\times$ speed; bottom right has $1.5\times$ speed.

Fig. 5. Three active perception tasks and solution paths. Object parts are shown in the coloured point clouds. Viewpoint regions are coloured black (low reward), orange (medium) and yellow (high).

its parts observed. The parts not selected were in the bottom left and top left, which is expected since the time to travel to these regions is relatively high. The waypoints within the selected regions naturally found locations near the edges of the regions and closer to the other regions, which implicitly minimises the travel time. All of the parts in the top right were selected, even though they are further from the other objects, since there is a significant reward to be gained by visiting two objects in close proximity.

The second scenario was planned for two robots with different budgets. Fig. 5b shows that the algorithm finds a natural partitioning between the robots in the same ratio of the travel budgets. The implicit partitioning naturally shared some of the objects between the two robots where the object parts were closer to a different robot. The 100 trials had a weighted sum of 174 nodes on average, with a standard deviation of 3.4, out of the maximum 189. The worst plan had 156 while the best had 177, showing the distribution of plans was skewed towards the best performing plans. The average runtime was 11.7 s with the standard deviation 0.2 s.

A similar partitioning was achieved in the third scenario, shown in Fig. 5c, for three robots with varying speeds in the most cluttered environment. The size of the implicit partitions are proportional to the speeds of the robots. The centre was well covered since several parts are observed at once from these locations and therefore have high reward. The average sum of weighted nodes was 199.2 out of the maximum 221, with standard deviation 9.1, worst case 175 and best case 211. The average runtime was 17.6 s with a standard deviation of 0.6 s. The performance was almost as consistent in this more complex scenario, and the solution paths have credible partitioning between multiple robots, selected lower cost locations within regions and favoured high-reward locations with overlapping viewpoint regions.

VII. Discussion and Future Work

Overall, our results show that the proposed method enables multi-robot planning for budgeted active perception tasks with continuous sets of candidate viewpoints and multi-step planning horizons. The results motivate several avenues of future work. In particular, we are interested in extending the algorithm for non-euclidean distances such as for obstacles [16] or non-holonomic constraints [27]. We are also interested in having different sensors within a heterogeneous team, which should be a straight-forward extension. The observation model may also include rewards for exploring unobserved space, and different robots in the team may be better equipped to achieve this. Online replanning would also be useful and achievable with our approach since the iterative learning process should be able to adapt to changes in the environment model. Other extensions include time-varying and probabilistic sensing models, constrained start or end locations [17], and decentralised planning.

References

[1] H. van Hoof, O. Kroemer, and J. Peters, "Probabilistic segmentation and targeted exploration of objects in cluttered environments," *IEEE T. Robot.*, vol. 30, no. 5, pp. 1198–1209, 2014.

[2] K. Wu, R. Ranasigne, and G. Dissanayake, "Active recognition and pose estimation of household objects in clutter," in *Proc. of IEEE ICRA*, 2015, pp. 4230–4237.

[3] T. Patten, M. Zillich, R. Fitch, M. Vincze, and S. Sukkarieh, "Viewpoint evaluation for online 3-D active object classification," *IEEE Robot. Autom. Lett.*, vol. 1, no. 1, pp. 73–81, 2016.

[4] S. Chen, Y. Li, and N. M. Kwok, "Active vision in robotic systems: A survey of recent developments," *Int. J. Robot. Res.*, vol. 30, no. 11, pp. 1343–1377, 2011.

[5] I. Becerra, L. M. Valentín-Coronado, R. Murrieta-Cid, and J.-C. Latombe, "Reliable confirmation of an object identity by a mobile robot: A mixed appearance/localization-driven motion approach," *Int. J. Robot. Res.*, 2016, doi 10.1177/0278364915620848.

[6] N. Atanasov, B. Sankaran, J. Le Ny, G. Pappas, and K. Daniilidis, "Nonmyopic view planning for active object classification and pose estimation," *IEEE T. Robot.*, vol. 30, no. 5, pp. 1078–1090, 2014.

[7] G. A. Hollinger, U. Mitra, and G. S. Sukhatme, "Active classification: Theory and application to underwater inspection," in *Proc. of ISRR*, 2011.

[8] Z. Xu, R. Fitch, J. Underwood, and S. Sukkarieh, "Decentralized coordinated tracking with mixed discrete-continuous decisions," *J. Field Robot.*, vol. 30, no. 5, pp. 717–740, 2013.

[9] S. Garg and N. Ayanian, "Persistent monitoring of stochastic spatio-temporal phenomena with a small team of robots," in *Proc. of Robotics: Science and Systems*, 2014.

[10] B. Charrow, "Information-theoretic active perception for multi-robot teams," Ph.D. dissertation, University of Pennsylvania, 2015.

[11] P. M. Dames, M. Schwager, D. Rus, and V. Kumar, "Active magnetic anomaly detection using multiple micro aerial vehicles," *IEEE Robot. Autom. Lett.*, vol. 1, no. 1, pp. 153–160, 2016.

[12] J. Faigl and G. Hollinger, "Unifying multi-goal path planning for autonomous data collection," in *Proc. of IEEE/RSJ IROS*, 2014, pp. 2937–2942.

[13] P. Vansteenwegen, W. Souffriau, and D. V. Oudheusden, "The orienteering problem: A survey," *Eur. J. Oper. Res.*, vol. 209, no. 1, pp. 1–10, 2011.

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

[15] K. Helsgaun, "An effective implementation of the Lin-Kernighan traveling salesman heuristic," *Eur. J. Oper. Res.*, vol. 126, no. 1, 2000.

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

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

[18] T. Patten, A. Kassir, W. Martens, B. Douillard, R. Fitch, and S. Sukkarieh, "A Bayesian approach for time-constrained 3D outdoor object recognition," in *Proc. of IEEE ICRA Workshop on Scaling Up Active Perception*, 2015.

[19] W. Wohlkinger and M. Vincze, "Ensemble of shape functions for 3D object classification," in *Proc. of IEEE ROBIO*, 2011, pp. 2987–2992.

[20] N. Mathew, S. Smith, and S. Waslander, "A graph-based approach to multi-robot rendezvous for recharging in persistent tasks," in *Proc. of IEEE ICRA*, 2013, pp. 3497–3502.

[21] J. Binney and G. Sukhatme, "Branch and bound for informative path planning," in *Proc. of IEEE ICRA*, 2012, pp. 2147–2154.

[22] G. Best, W. Martens, and R. Fitch, "A spatiotemporal optimal stopping problem for mission monitoring with stationary viewpoints," in *Proc. of Robotics: Science and Systems*, 2015.

[23] T. Bektas, "The multiple traveling salesman problem: An overview of formulations and solution procedures," *Omega*, vol. 34, no. 3, pp. 209–219, 2006.

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

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

[26] S. Cunningham-Nelson, P. Moghadam, J. Roberts, and A. Elfes, "Coverage-based next best view selection," in *Proc. of ACRA*, 2015.

[27] J. Faigl, "On self-organizing map and rapidly-exploring random graph in multi-goal planning," in *Advances in Self-Organizing Maps and Learning Vector Quantization*. Springer, 2016, pp. 143–153.