Dec-MCTS: Decentralized planning for multi-robot active perception

Article *in* The International Journal of Robotics Research · March 2018

DOI: 10.1177/0278364918755924

CITATIONS
82

READS
631

5 authors, including:

Graeme Best
Oregon State University
30 PUBLICATIONS **416** CITATIONS
SEE PROFILE

Oliver Michael Cliff
The University of Sydney
30 PUBLICATIONS **810** CITATIONS
SEE PROFILE

Timothy Patten
TU Wien
54 PUBLICATIONS **631** CITATIONS
SEE PROFILE

Ramgopal R Mettu
Tulane University
46 PUBLICATIONS **642** CITATIONS
SEE PROFILE

Dec-MCTS: Decentralised planning for multi-robot active perception

Journal Title
XX(X):1–28

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Graeme Best[1], Oliver M. Cliff[1], Timothy Patten[1,2], Ramgopal R. Mettu[3] and Robert Fitch[1,4]

Abstract

We propose a decentralised variant of Monte Carlo tree search (MCTS) that is suitable for a variety of tasks in multi-robot active perception. Our algorithm allows each robot to optimise its own actions by maintaining a probability distribution over plans in the joint-action space. Robots periodically communicate a compressed form of their search trees, which are used to update the joint distribution using a distributed optimisation approach inspired by variational methods. Our method admits any objective function defined over robot action sequences, assumes intermittent communication, is anytime, and is suitable for online replanning. Our algorithm features a new MCTS tree expansion policy that is designed for our planning scenario. We extend the theoretical analysis of standard MCTS to provide guarantees for convergence rates to the optimal payoff sequence. We evaluate the performance of our method for generalised team orienteering and online active object recognition using real data, and show that it compares favourably to centralised MCTS even with severely degraded communication. These examples demonstrate the suitability of our algorithm for real-world active perception with multiple robots.

Keywords

Decentralised Monte Carlo tree search, multi-robot systems, decentralised planning, active perception, variational methods

1 Introduction

Information gathering is an important family of problems in robotics that plays a primary role in a wide variety of tasks, including scene understanding (Fäulhammer et al. 2017), manipulation (Kahn et al. 2015), environmental monitoring (Dunbabin and Marques 2012) and target localisation (Cliff et al. 2015). Although the idea of exploiting robot motion to improve the quality of information gathering has been studied for nearly three decades (Bajcsy 1988; Bajcsy et al. 2017), most real robot systems today (both single- and multi-robot) still gather information passively. The motivation for an active approach is that sensor data quality (and hence, perception quality) relies critically on an appropriate choice of viewpoints (Patten et al. 2016). One way to efficiently achieve an improved set of viewpoints is through teams of robots, where concurrency allows for scaling up the number of observations in time and space. The key challenge, however, is to coordinate the behaviour of

[1]Australian Centre for Field Robotics (ACFR), The University of Sydney, Sydney, Australia
[2]Automation and Control Institute, Vienna University of Technology, Vienna, Austria
[3]Department of Computer Science, Tulane University, New Orleans, LA, USA
[4]Centre for Autonomous Systems, University of Technology Sydney, Sydney, Australia

Corresponding author:
Graeme Best, Australian Centre for Field Robotics (ACFR), J04, The University of Sydney, NSW, 2006, Australia
Email: g.best@acfr.usyd.edu.au

robots as they actively gather information. Ideally, this coordination should be decentralised so that the system is scalable and robust to failures. This paper presents an online, decentralised planning algorithm for active perception that allows a team of robots to perform complex information gathering tasks using physically feasible sensor and motion models, and reasonable communication assumptions.

Monte Carlo tree search (MCTS) is a promising approach for online planning because it efficiently searches over long planning horizons and is anytime (Browne et al. 2012; Kocsis and Szepesvári 2006). MCTS is applicable to general problem formulations but can readily incorporate problem-specific heuristics. Recently, MCTS has been successfully applied to robotics scenarios such as active object recognition (Patten et al. 2017), wildlife monitoring (Hefferan et al. 2016), and planetary exploration (Arora et al. 2017). These studies were for single-robot problems, while in this paper we propose a new planning algorithm that is a viable solution for all of these scenarios extended for multi-robot teams.

In this paper we propose a new decentralised planning algorithm, Dec-MCTS, that is essentially a novel decentralised variant of MCTS. At a high level, our method alternates between exploring each robot's individual action space and optimising a probability distribution over the joint-action space. In any particular round, we first use a new variant of MCTS to find locally favourable sequences of actions for each robot given probabilistic estimates of other robots' actions that evolve during planning-time. The main novelty is our new tree expansion policy, motivated by discounted-UCB (Garivier and Moulines 2011), that accounts in general for changing reward distributions.

Then, robots periodically attempt to asynchronously communicate a highly compressed version of their local search trees which, together, correspond to a product distribution approximation. These communicated distributions are used to estimate the underlying joint distribution for the teams' plan. The estimates are probabilistic, unlike the deterministic representation of joint actions typically used in multi-robot coordination algorithms. Optimising a product distribution is similar in spirit to the mean-field approximation from variational inference, and also has a natural game-theoretic interpretation (Rezek et al. 2008; Wolpert and Bieniawski 2004).

Our algorithm is a powerful new method of decentralised coordination for any objective function defined over the robot action sequences. Notably, this implies that our method is suitable for complex perception tasks such as object classification, which is known to be highly viewpoint-dependent (Patten et al. 2016). Further, communication is assumed to be intermittent, and the amount of data sent over the network is small in comparison to the raw data generated by typical range sensors and cameras. Our method also inherits important properties from MCTS, such as the ability to compute anytime solutions and to incorporate prior knowledge about the environment. Moreover, our method is suitable for online replanning to adapt to changes in the objective function or team behaviour.

We provide an extensive theoretical analysis of the algorithm that leverages results from probability theory and game theory. Our main analytical result is to show convergence rates for the expected payoff at the root of the search tree towards the optimal payoff sequence. Thus, the proposed MCTS tree expansion policy balances exploration and exploitation while the reward distributions are changing. This result is proven by extending the MCTS analysis of Kocsis et al. (2006) for the context of switching bandit problems (Garivier and Moulines 2011). Our second analytical result leverages Wolpert et al. (2006) to show that the product distribution optimisation phase locally minimises the KL divergence to the optimal joint probability distribution. While, given the difficulty of the problem, these results do not directly yield guarantees for global optimality, the analysis provides strong motivation for the use of these components in our algorithm for decentralised, long-horizon planning with general objective function definitions.

We empirically evaluate our algorithm in two scenarios: generalised team orienteering and active object recognition. These experiments are run in simulation, where the robots traverse a PRM with a Dubins motion model, and the second scenario uses range sensor data collected a priori by real robots. We show that our decentralised approach performs as well as or better than centralised MCTS even with a significant rate of communication message loss. We also show the benefits

of our algorithm in performing long-horizon and online planning.

1.1 Contributions

This paper proposes a new multi-robot planning algorithm that is decentralised, admits a general class of objective functions, optimises actions over a long planning horizon, is anytime, robust to communication degradation, and is suitable for online replanning. The algorithm is a novel decentralised variant of MCTS, which features a new tree expansion policy suitable for our context, and combines MCTS with a probabilistic distributed optimisation approach inspired by variational methods. We provide a theoretical analysis and empirical results that demonstrate the suitability of our planning algorithm for coordinated information gathering tasks.

This paper is an extended and revised version of Best et al. (2016) presented at WAFR, 2016. The main new contribution is a theoretical analysis of the key MCTS component of our algorithm by relating it to a new multi-armed bandit problem. Additionally, we provide: an extended review of related literature, expanded algorithmic details, a discussion of generalisations for probabilistic objectives, and implementation details.

The remainder of this paper is organised as follows. Section 2 discusses related work in decentralised planning, MCTS, variational methods, and non-myopic planning. Section 3 defines the decentralised planning problem. Section 4 presents our proposed Dec-MCTS algorithm. Section 5 provides a theoretical analysis of Dec-MCTS, with proofs of intermediate results provided in the Appendix. Sections 6 and 7 present an empirical analysis of our algorithm for two example active perception problems. Finally, Sec. 8 concludes the paper and discusses future work.

2 Related work

2.1 Decentralised information gathering

Information gathering problems can be viewed as sequential decision processes in which actions are chosen to maximise an objective function. The computational burden of decentralised coordination is typically overcome by using myopic solvers which maximise the objective function over a limited time horizon (Xu et al. 2013; Gan et al. 2014). Unfortunately, the quality of solutions produced by myopic methods can be arbitrarily poor in the general case. Recently, however, analysis of submodularity (Nemhauser et al. 1978) has shown that myopic methods can achieve near-optimal performance (Krause et al. 2008), which has led to considerable interest in their application to information gathering with multiple robots (Singh et al. 2009; Hollinger et al. 2009; Patten et al. 2013; Garg and Ayanian 2014). While these greedy methods provide theoretical guarantees, they require a submodular objective function, which is not applicable in all cases. Additionally, while these methods often guarantee lower bounds on optimality, the solution quality can typically be improved by planning over longer horizons. Similarly, for simplified problems that only require selecting one action per robot rather than sequences of actions, decentralised task allocation approaches are often sufficient (Liu et al. 2015).

Efficient non-myopic decentralised coordination algorithms can be designed by exploiting problem-specific characteristics. Market-based methods (Dias et al. 2006) involve each robot negotiating over which tasks it will perform, and are more appropriate for coverage and exploration problems (Zlot et al. 2002). Sadeghi and Smith (2017) apply an auction method with TSP heuristics to a problem formulated as a generalisation of the TSP. Stranders et al. (2009) combine max-sum message passing with branch and bound pruning to find sequences of viewpoints that minimise the entropy of a Gaussian process. Otte and Correll (2013) propose a distributed RRT algorithm for coordinated path planning with collision avoidance. The authors demonstrated a graceful degradation of performance as communication becomes less reliable, and we observe a similar behaviour with our Dec-MCTS algorithm. Corah and Michael (2017) propose a distributed sequential greedy assignment algorithm for multi-robot exploration, and provide performance guarantees by exploiting a submodularity assumption. Atanasov et al. (2015) propose a decentralised algorithm for tracking targets that have linear Gaussian dynamics, such as for active SLAM. Our proposed algorithm is applicable to a general class of problems since it does not rely on specific assumptions about the problem; however, our approach can readily incorporate problem-specific approximate solutions, such as those above, as heuristics to guide the search.

2.2 Dec-POMDPs

Decentralised active information gathering can be viewed, in general, as a partially observable Markov decision process (POMDP) in decentralised form (Dec-POMDP) (Bernstein et al. 2002; Oliehoek and Amato 2016). Typically, Dec-POMDP formulations are solved by performing centralised, offline planning over the joint multi-agent policy space, and then these policies are executed online in a decentralised fashion (Oliehoek and Amato 2016; Amato 2015; Kumar et al. 2015; Omidshafiei et al. 2017). These centralised, offline planning approaches are impractical in scenarios with large sources of uncertainty, such as when the state of the environment is unknown ahead of time.

In contrast, Spaan et al. (2006) address a Dec-POMDP problem setting where both planning and execution are performed in a decentralised manner; we solve our problem in a similar decentralised setting such that computation is performed online and on-board the team of robots. Spaan et al. (2006) propose a general Dec-POMDP solver where each agent solves a single-agent POMDP, shares information about its own plan, then repeats. At a high-level, our algorithm is similar to this general approach, but we differ in how we share information, and we propose solving the single-agent sub-problems with an anytime, incremental planner that can account for changing information about the teams' plan in a principled manner.

While the problem definition we consider in this paper is not formulated as a Dec-POMDP in general form, extended algorithms for the Dec-POMDP case could be designed by using POMCP (Silver and Veness 2010).

2.3 Monte Carlo tree search

MCTS has recently become popular for online planning in robotics. MCTS has been proposed in many different forms (Browne et al. 2012) but by far the most common is the upper-confidence bounds applied to trees (UCT) algorithm (Kocsis and Szepesvári 2006; Kocsis et al. 2006). The UCT algorithm performs an asymmetric expansion of a search tree using a best-first policy that generalises the UCB1 policy for multi-armed bandit (MAB) problems (Auer et al. 2002). This expansion policy provides theoretical guarantees for a polynomial bound on regret and therefore is said to balance between exploration and exploitation. Several variants to UCT have been proposed, such as for exploiting smoothness of the reward function (Coquelin and Munos 2007). A key component of our proposed Dec-MCTS algorithm is a novel UCT variant, D-UCT, that accounts for a changing reward distribution by using a new expansion policy that generalises the D-UCB policy for switching bandit problems (Garivier and Moulines 2011). MCTS algorithms have also been extended for problems with partial-observability, such as the POMCP (Silver and Veness 2010) and DESPOT (Somani et al. 2013) algorithms, and Dec-MCTS could be extended in a similar way. However, MCTS has not yet been extended for decentralised multi-agent planning, which is the focus of this paper.

MCTS is parallelisable (Chaslot et al. 2008), and various techniques have been proposed that split the search tree across multiple processors and combine their results. In the multi-robot case, the joint search tree interleaves actions of individual robots and it remains a challenge to effectively partition this tree. Auger (2011) addresses the related case of multi-player games, where a separate tree is maintained for each player; however, a single simulation traverses all of the trees and therefore this approach would be difficult to decentralise. We propose a similar approach, except that each robot performs independent simulations while sampling from a locally stored probability distribution that represents the other robots' action sequences.

MCTS has been applied to a wide variety of single-robot tasks, including: active object recognition (Patten et al. 2017; Lauri et al. 2015), patrolling environments with adversarial agents (Hefferan et al. 2016; Kartal et al. 2015), information gathering by a glider in thermal wind-fields (Nguyen et al. 2015), environment exploration (Lauri and Ritala 2016; Corah and Michael 2017), autonomous science by planetary rovers (Arora et al. 2017), active parameter estimation for manipulation (Slade et al. 2017), and monitoring of a spatiotemporal process (Marchant et al. 2014). We propose a decentralised MCTS algorithm that is suitable for multi-robot generalisations of all of these problems. So far, MCTS has been less studied in multi-robot scenarios, though promising ideas have been presented by Kartal et al. (2015) for centralised planning of a team of patrolling

robots, and by Corah and Michael (2017) as a single-robot planner within a distributed multi-robot assignment algorithm for the context of exploration and mapping.

2.4 Variational methods for planning

Coordination between robots is achieved in our method by combining MCTS with a framework that optimises a product distribution over the joint action space in a decentralised manner. Our approach is analogous to the classic mean-field approximation and related variational methods (Yedidia et al. 2005; Rezek et al. 2008; Koller and Friedman 2009). Other graphical model inference techniques, alternative to variational methods, have motivated related coordination algorithms (Stranders et al. 2009; Kumar et al. 2015; Regev and Indelman 2016).

Variational methods seek to approximate the underlying global likelihood with a collection of structurally simpler distributions that can be evaluated efficiently and independently. These methods characterise convergence based on the choice of product distribution, and work best when it is possible to strike a balance between the convergence properties of the product distribution and the KL divergence between the product and joint distributions. As discussed in the body of work on *probability collectives* (PC) (Wolpert and Bieniawski 2004; Wolpert et al. 2006, 2013), such variational methods can also be viewed under a game theoretic interpretation, where the goal is to optimise each agent's action selection based on examples of the global reward/utility function.

PC has been applied to robotics problems, but so far has been limited to selecting a single action from a small action space (Waldock and Nicholson 2007), rather than planning sequences of actions. An exception to this is Kulkarni and Tai (2010), who combine PC with TSP heuristics to solve the multiple-TSP in a decentralised manner. We propose a similar approach to Kulkarni and Tai (2010), but instead we leverage the long-horizon planning of MCTS to dynamically select an effective and compact sample space of action sequences.

2.5 Non-myopic, single-robot planning

Long-horizon planning is also beneficial for single-robot information gathering. As discussed in Sec. 2.3, MCTS approaches have recently gained popularity. Branch and bound tree search (Binney and Sukhatme 2012; Best and Fitch 2016) is closely related to MCTS but requires problem-specific bounds with hard guarantees; in contrast, UCT relies on generally-applicable probabilistic-bounds derived from the Chernoff-Hoeffding inequality. Active perception problems have also been formulated as travelling salesman problem variants (Best et al. 2018; Charrow 2015; Yu et al. 2016). A sampling-based planner has been proposed for exploration and inspection tasks (Bircher et al. 2016). Extensions of RRT have been designed for problems where the environment is naturally modelled as a continuous process (Hollinger and Sukhatme 2014), e.g., Gaussian processes. Hollinger (2015) extends the RRT approach to be more suitable for real-time planning. In exploration scenarios, imitation learning can be used for planning by learning from examples that have assumed full knowledge of the world (Choudhury et al. 2017). For active object recognition—a primary motivating scenario for Dec-MCTS—planning has typically been limited to greedy approaches (Wu et al. 2015; van Hoof et al. 2014; Huber et al. 2012); MCTS approaches are notable exceptions (Patten et al. 2017; Lauri et al. 2015).

3 Problem statement

We consider a team of R robots $\{1, 2, \dots, R\}$, where each robot r plans its own sequence of future actions $\boldsymbol{x}^r = (x_1^r, x_2^r, \dots)$. Each action x_j^r has an associated cost c_j^r and each robot has a cost budget B^r such that the sum of the costs must be less than the budget, i.e., $\sum_{x_j^r \in \boldsymbol{x}^r} c_j^r \leq B^r$. This cost budget may be an energy or time constraint defined by the application, or it may be used to enforce a planning horizon. The feasible set of actions and associated costs at each step j are a function of the previous actions $(x_1^r, x_2^r, \dots, x_{j-1}^r)$. Thus, there is a predefined set $\mathcal{X}^r$ of feasible action sequences $\boldsymbol{x}^r$ for each robot r. We denote $\boldsymbol{x}$ as the set of action sequences for all robots $\boldsymbol{x} \coloneqq \{\boldsymbol{x}^1, \boldsymbol{x}^2, \dots, \boldsymbol{x}^R\}$ and $\boldsymbol{x}^{(r)}$ as the set of action sequences for all robots except robot r, i.e., $\boldsymbol{x}^{(r)} \coloneqq \boldsymbol{x} \setminus \boldsymbol{x}^r$. We denote $\mathcal{X}$ as the set of all feasible $\boldsymbol{x}$ and $\mathcal{X}^{(r)}$ as the set of all feasible $\boldsymbol{x}^{(r)}$.

The aim is to maximise a global objective function $g(\boldsymbol{x})$ that is a function of the action sequences of all robots. We assume each robot r knows the global objective function g, but does not know the action sequences $\boldsymbol{x}^{(r)}$ selected by the other robots. For most

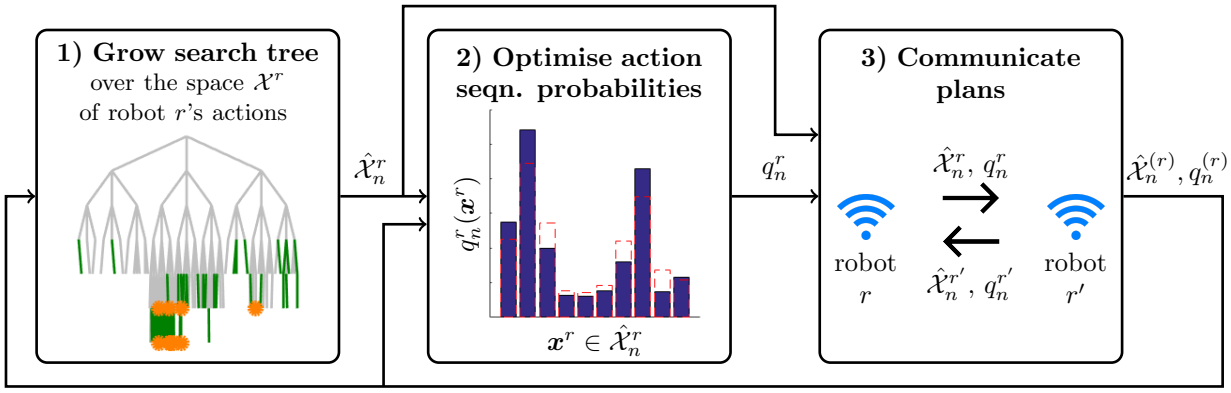

Figure 1. Overview of the algorithm running on-board robot r. 1) The search tree is expanded by adding new actions (green). Periodically, the set of best nodes (orange) is selected as the domain $\hat{\mathcal{X}}_n^r$. 2) The probability distribution q_n^r is optimised (from dotted red to solid blue). 3) If possible, the domains and distributions are communicated between robots.

of our proposed approach, we assume g is deterministic given a known set of action sequences $\boldsymbol{x}$; in Sec. 4.7 we discuss extensions for probabilistic objective functions.

The problem must be solved in a decentralised and online setting. We assume that robots can communicate during planning-time to improve coordination. The communication channel may be unpredictable and intermittent, and all communication is asynchronous. Therefore, each robot will plan based on the information it has available locally. Bandwidth may be constrained and therefore message sizes should remain small, even as the plans grow. Although we do not consider explicitly planning to maintain communication connectivity, this may be encoded in the objective function $g(\boldsymbol{x})$ if a reliable communication model is available.

4 Dec-MCTS

In this section, we present our Dec-MCTS algorithm as a decentralised solution to the general multi-robot planning problem. We first provide an overview of the algorithm followed by a detailed explanation of all components.

4.1 Algorithm overview

Dec-MCTS runs simultaneously and asynchronously on all robots; we present the algorithm from the perspective of robot r. The algorithm cycles between the three phases illustrated in Fig. 1: (1) incrementally grow a search tree using MCTS while taking into account information about the other robots' plans, (2) update the probability distribution over possible action sequences, and (3) communicate probability distributions with the other robots. These three phases continue regardless of whether or not the communication was successful, until a computation budget is met.

A key idea of Dec-MCTS is to represent and reason over plans in a probabilistic manner. In particular, robot r's current plan is represented by a probability distribution over action sequences. We define a probability mass function q_n^r, such that $q_n^r(\boldsymbol{x}^r)$ defines the probability that robot r will select the action sequence $\boldsymbol{x}^r$. In general, the domain of the distribution q_n^r is the set of all possible action sequences $\mathcal{X}^r$. However, to enable tractable computation and realistic communication, we restrict the domain of q_n^r to a dynamically selected subset $\hat{\mathcal{X}}_n^r \subset \mathcal{X}^r$, i.e., $q_n^r(\boldsymbol{x}^r) = 0, \forall \boldsymbol{x}^r \notin \hat{\mathcal{X}}_n^r$. As the Dec-MCTS algorithm progresses, both the domain $\hat{\mathcal{X}}_n^r$ and the probability distribution q_n^r are optimised. Note the subscript n for q_n^r and $\hat{\mathcal{X}}_n^r$ is used to denote the nth iteration of the main loop of our algorithm.

An illustration of the main loop is shown in Fig. 1 and pseudocode for the algorithm is provided in Alg. 1. During the MCTS phase, a search tree $\mathcal{T}^r$ is grown over the space $\mathcal{X}^r$ of robot r's action sequences using a new variant of the UCT algorithm. This tree growth is performed while considering the

Algorithm 1 Overview of Dec-MCTS for robot r.

input: global objective function g, budget B^r, feasible action sequences and costs
output: sequence of actions $\boldsymbol{x}^r$ for robot r

$\mathcal{T}^r \leftarrow$ initialise MCTS tree
while computation budget not met, at iteration n **do**
  $\hat{\mathcal{X}}_n^r \leftarrow$ SELECTSETOFSEQUENCES$(\mathcal{T}^r)$ ▷ See Sec. 4.4
  for τ_n iterations **do**
    $\mathcal{T}^r \leftarrow$ GROWTREE$(\mathcal{T}^r, \hat{\mathcal{X}}_n^{(r)}, q_n^{(r)}, B^r)$ ▷ See Alg. 2 and Sec. 4.3
    $q_n^r \leftarrow$ UPDATEDISTRIBUTION$(\hat{\mathcal{X}}_n^r, q_n^r, \hat{\mathcal{X}}_n^{(r)}, q_n^{(r)}, \beta)$ ▷ See Alg. 3 and Sec. 4.4
    COMMUNICATIONTRANSMIT$(\hat{\mathcal{X}}_n^r, q_n^r)$ ▷ See Sec. 4.5
    $(\hat{\mathcal{X}}_n^{(r)}, q_n^{(r)}) \leftarrow$ COMMUNICATIONRECEIVE ▷ See Sec. 4.5
    $\beta \leftarrow$ COOL(β) ▷ See Sec. 4.4
return $\boldsymbol{x}^r \leftarrow \arg\max_{\boldsymbol{x}^r \in \hat{\mathcal{X}}_n^r} [q_n^r(\boldsymbol{x}^r)]$

probability distributions over the other robots plans, denoted $\hat{\mathcal{X}}_n^{(r)}$, $q_n^{(r)}$. Periodically, the domain $\hat{\mathcal{X}}_n^r$ for robot r's distribution is updated by selecting the most promising action sequences identified by the tree search. In the probability distribution optimisation phase, the probabilities assigned to action sequences $q_n^r(\boldsymbol{x}^r)$ are optimised using a decentralised gradient descent algorithm while considering the distributions of the other robots. In the communication phase, robot r communicates its domain $\hat{\mathcal{X}}_n^r$ and probability distribution q_n^r to the other robots. If robot r receives a new distribution from any of the other robots, then in the next iteration $\hat{\mathcal{X}}_n^r$ and q_n^r are optimised while considering this new information. During this optimisation process, it is possible that $q_n^{(r)}$ will change such that a previously optimal leaf of the tree $\mathcal{T}^r$ becomes suboptimal; we refer to the times at which this happens as breakpoints.

When the computation budget is met, the algorithm returns the action sequence $\boldsymbol{x}^r$ that has the highest probability $q_n^r(\boldsymbol{x}^r)$. In online settings, the robot would then typically execute the first action x_1^r in the action sequence, and then perform replanning to take into account new information received by observations. If the changes to the objective function are minor, then replanning may be performed more efficiently by adapting the previous search tree.

4.2 Local utility function

The global objective function g is optimised by each robot r using a local utility function f^r. We define f^r as the difference in global utility between robot r performing action sequence $\boldsymbol{x}^r$ and a default "no reward" sequence $\boldsymbol{x}_\emptyset^r$, assuming fixed action sequences $\boldsymbol{x}^{(r)}$ for the other robots, i.e.,

$$f^r(\boldsymbol{x}) \coloneqq g(\boldsymbol{x}^r \cup \boldsymbol{x}^{(r)}) - g(\boldsymbol{x}_\emptyset^r \cup \boldsymbol{x}^{(r)}). \quad (1)$$

The default sequence $\boldsymbol{x}_\emptyset^r$ is chosen to be suitable for the application and would typically be an empty action sequence. In practice, optimising with respect to f^r rather than g improves the performance since f^r is more sensitive to robot r's plan and the variance of f^r is less affected by the uncertainty of the other robots' plans (Wolpert et al. 2013). We chose this local utility function since it is generally applicable, although further performance improvements could be achieved with problem-specific heuristics (Rahmattalabi et al. 2016). We note that this formulation assumes that all robots know the global utility function g. However, if instead each robot only has access to a local estimate of g then our proposed algorithm will optimise the action sequences with respect to this inconsistent information.

4.3 Monte Carlo tree search with discounted-UCB

The first phase of the algorithm is the MCTS update shown in Alg. 2. A single search tree $\mathcal{T}^r$ is maintained

Algorithm 2 Grow the search tree for robot r using Monte Carlo tree search.

1: **function** GROWTREE($\mathcal{T}^r, \hat{\mathcal{X}}_n^{(r)}, q_n^{(r)}, B^r$)
   input: partial tree $\mathcal{T}^r$, distributions for other robots $(\hat{\mathcal{X}}_n^{(r)}, q_n^{(r)})$, budget B^r
   output: updated partial tree $\mathcal{T}^r$
2: **for** fixed number of samples **do**
3: $i_{d-1} \leftarrow$ NODESELECTIOND-UCT($\mathcal{T}^r$) ▷ Select node to expand using D-UCT policy (Sec. 4.3.1)
4: $i_d \leftarrow$ EXPANDTREE(i_{d-1}) ▷ Add new child to node i_{d-1}
5: $\boldsymbol{x}^{(r)} \leftarrow$ SAMPLE($\hat{\mathcal{X}}_n^{(r)}, q_n^{(r)}$) ▷ Sample action sequences of other robots
6: $\boldsymbol{x}^r \leftarrow$ PERFORMROLLOUTPOLICY($i_d, \boldsymbol{x}^{(r)}, B^r$) ▷ Default policy until budget exhausted
7: $F_{i_d,t} \leftarrow f^r(\boldsymbol{x}^r \cup \boldsymbol{x}^{(r)})$ ▷ Local utility function (Sec. 4.2)
8: $\mathcal{T}^r \leftarrow$ BACKPROPAGATION($\mathcal{T}^r, i_d, F_{i_d,t}$) ▷ Update statistics in tree
9: **return** $\mathcal{T}^r$

by robot r which only contains the actions of robot r. The tree $\mathcal{T}^r$ is defined such that each edge in the tree represents an action by robot r, and a path from the root node i_0 to another node i_d at depth d represents a valid sequence of actions by robot r. The MCTS algorithm incrementally grows $\mathcal{T}^r$ from the root node using a best-first expansion policy. During the MCTS phase, coordination with other robots occurs implicitly by considering the plans of the other robots when performing the rollout policy and evaluation of the global objective function. This information about the other robots' plans comes from the second phase of the algorithm, detailed later in Sec. 4.4. In this subsection, we detail our proposed MCTS algorithm which features a novel bandit-based node selection policy designed for our planning scenario.

Standard MCTS incrementally grows a tree by iterating through four phases: *selection*, *expansion*, *simulation* and *backprogation* (Browne et al. 2012). During each iteration t, a new leaf node is added, where each node represents a sequence of actions and contains statistics about the expected reward of all action sequences that begin with this sequence.

The selection phase (Alg. 2, line 3) selects an expandable node in the tree, where an expandable node is defined as a node that has at least one child that has not yet been visited during the search. In order to find an expandable node, the algorithm begins at the root node i_0 of the tree and recursively selects child nodes until an expandable node i_{d-1} is reached. For selecting the next child at each level of the tree, we propose an extension of the UCT policy (Kocsis and Szepesvári 2006), detailed in Sec. 4.3.1, to balance exploration and exploitation. In the expansion phase (Alg. 2, line 4), a new child node i_d is added to the selected expandable node i_{d-1}, which extends the parent's action sequence with an additional action.

In the simulation phase (Alg. 2, lines 5–7), the expected utility $\mathbb{E}[q]$ of the expanded node i_d is estimated by performing and evaluating a rollout policy that extends the action sequence represented by the node until a terminal state is reached. This rollout policy could be a random policy or a heuristic for the problem (James et al. 2017). The objective is evaluated for this sequence of actions and this result is saved.

For our problem, the objective is a function of the action sequence $\boldsymbol{x}^r$ as well as the unknown plans of the other robots $\boldsymbol{x}^{(r)}$, and thus we require an extension of the standard simulation procedure. To compute the rollout score, we first sample $\boldsymbol{x}^{(r)}$ from a probability distribution $q_n^{(r)}$ over the plans of the other robots (as defined in Sec. 4.1). A heuristic rollout policy extended from i_d defines $\boldsymbol{x}^r$, which should be a function of $\boldsymbol{x}^{(r)}$ to simulate coordination between the robots. Additionally, we optimise $\boldsymbol{x}^r$ using the local utility f^r (as defined in (1)) rather than g. The rollout score is computed as the utility of this joint sample $f^r(\boldsymbol{x}^r \cup \boldsymbol{x}^{(r)})$, which is an estimate for $\mathbb{E}_{q_n}[f^r \mid \boldsymbol{x}^r]$. We denote F_t as the rollout evaluation at sample round t.

In the backpropagation phase (Alg. 2, line 8), the rollout evaluation F_t is added to the statistics of all nodes along the path from the expanded node back to the root of

the tree. Typically, these statistics are unbiased estimators of the rollout evaluations; however, as we discuss in the following section, it is more suitable to use a weighted average in the context of Alg. 1.

4.3.1 D-UCB node selection policy The node selection policy is used in Alg. 2, line 3, and dictates the order in which the tree $\mathcal{T}^r$ is expanded. Consider an arbitrary node i_d at depth d in the tree which has an associated set of child nodes $\mathcal{C}(i_d)$. For every sample round t where node i_d is visited, the problem is to select a child $I_{i_d,t} \in \mathcal{C}(i_d)$ that balances both visiting promising subtrees and exploring uncertain ones.

An established approach for node selection is based on maintaining an *upper confidence bound (UCB)* on the value of each node. Under this paradigm, at each sample round t, a UCB U_{j,t_{i_d},t_j} is computed for all children $j \in \mathcal{C}(i_d)$ of the parent node i_d. Here, t_{i_d} is the number of times the parent node i_d has been visited and t_j is the number of times child node j has been visited. The algorithm then selects the node that maximises this quantity, i.e.,

$$I_{i_d,t} = \arg\max_{j \in \mathcal{C}(i_d)} U_{j,t_{i_d},t_j}. \tag{2}$$

This continues recursively until an expandable node is reached.

The *de facto* UCB U_{j,t_{i_d},t_j} is a combination of the empirical mean of rewards received at node j and a confidence interval derived from the Chernoff-Hoeffding inequality (Browne et al. 2012). This bound was originally used in the context of the MAB problem and called UCB1 (Auer et al. 2002); when used for tree search, it is labelled UCT (Kocsis and Szepesvári 2006). UCT was shown to yield polynomial regret when the reward distributions at the leaf nodes are stationary (Kocsis and Szepesvári 2006). However, Alg. 1 alternates between growing the tree for a number of rollouts τ_n and updating the probability distributions for other robots. As mentioned in Sec. 4.1, this introduces breakpoints as instants where the reward distribution and optimal action can change abruptly. We denote the number of breakpoints up until time t as Υ_t. Due to these breakpoints, the most recent rollouts are more relevant since they are obtained by sampling the most recent distributions. It was shown by Garivier and Moulines (2011) that UCB1 is inefficient in the bandit setting when breakpoints are expected. In this scenario a discounted variant, termed D-UCB, yields tighter bounds on regret. Due to the expected breakpoints caused by updating the distributions, we extend the approach of Garivier and Moulines (2011) for tree search, and propose a discounted variant of UCT for node selection, which we term D-UCT, described as follows.

Given some discount factor $\gamma \in (1/2, 1)$ and exploration constant $C_p > 1/\sqrt{8}$, the D-UCT bound is defined as:

$$U_{j,t_{i_d},t_j}(\gamma) \coloneqq \bar{F}_{j,t_j}(\gamma) + c_{t_{i_d},t_j}(\gamma), \tag{3}$$

where $\bar{F}_{j,t_j}(\gamma)$ is the discounted empirical reward, and $c_{t_{i_d},t_j}(\gamma)$ is a discounted exploration bonus. A lower discount factor γ enforces only the most recent rollouts to contribute towards the UCB, whereas at the upper limit $\gamma \to 1$ D-UCT becomes equivalent to UCT. These quantities are computed as follows. First, recall that the indicator function $\mathbf{1}_{\{I_{i_d,t}=j\}}$ returns 1 if node j was selected at round t, and 0 otherwise. Then, denote the discounted number of times the child node j has been visited as:

$$t_j(\gamma) \coloneqq \sum_{u=1}^{t} \gamma^{t-u} \mathbf{1}_{\{I_{i_d,u}=j\}}, \tag{4}$$

and the discounted number of times the parent node has been visited as:

$$t_{i_d}(\gamma) \coloneqq \sum_{j \in \mathcal{C}(i_d)} t_j(\gamma). \tag{5}$$

Recall that F_t is the rollout score received at sample t. Then, the discounted empirical average is given by:

$$\bar{F}_{j,t_j}(\gamma) \coloneqq \frac{1}{t_j(\gamma)} \sum_{u=1}^{t} \gamma^{t-u} F_u \mathbf{1}_{\{I_{i_d,u}=j\}}, \tag{6}$$

and the discounted exploration bonus is defined as:

$$c_{t_{i_d},t_j}(\gamma) \coloneqq 2C_p \sqrt{\frac{\log t_{i_d}(\gamma)}{t_j(\gamma)}}. \tag{7}$$

The aim of an online planner, such as Dec-MCTS, is to find the best first action, execute this action, and then replan. Thus, we are interested in the convergence of the

root node towards selection of the optimal action. Given the expected upper bound on the number of breakpoints occuring in the subtree rooted at node j, i.e., $\mathbb{E}[\Upsilon_{t_j}]$, selecting the discounted factor as

$$\gamma_{t_j} = 1 - \sqrt{\frac{\mathbb{E}[\Upsilon_{t_j}]}{16 t_j}} \tag{8}$$

allows us to minimise the time for this convergence. This is analysed further in Sec. 5. Having γ change dynamically, such as in (8), makes it difficult to efficiently recompute $\bar{F}_{j,t_j}$ and $c_{t_{i_d},t_j}$ as t grows large. Therefore, in practice, typically it is best to set γ to a fixed constant.

4.4 Decentralised product distribution optimisation

The second phase of the algorithm updates a probability distribution q_n^r over the set of possible action sequences for robot r. The distribution q_n^r serves as a way of predicting the likelihood of an action sequence being selected as the search tree continues to grow. These distributions are communicated between robots and used when performing rollouts during future iterations of MCTS. To define and optimise these distributions in a decentralised manner for improving global utility, we adapt a type of variational method originally proposed by Wolpert and Bieniawski (2004). This formulation can be viewed as a game between independent robots, where each robot selects its action sequence by sampling from a distribution. The approach to solving this formulation is essentially a decentralised gradient descent method over the space of product distributions. Pseudocode is provided in Alg. 3.

One challenge is that the set of possible action sequences $\mathcal{X}^r$ typically has a cardinality that is exponential in the time horizon. We obtain a sparse representation by periodically selecting the sample space $\hat{\mathcal{X}}_n^r \subset \mathcal{X}^r$ as the most promising action sequences $\{\boldsymbol{x}_1^r, \boldsymbol{x}_2^r, \ldots\}$ found by MCTS so far (Alg. 1, line 3). We select a fixed number of nodes in the search tree $\mathcal{T}^r$ that currently have the highest discounted empirical average $\bar{F}(\gamma)$. The set $\hat{\mathcal{X}}_n^r$ is chosen as the action sequences used during the initial rollouts when the selected nodes were first expanded.

As mentioned in Sec. 4.1, when the sample spaces $\hat{\mathcal{X}}_n^{(r)}$ are updated, this can introduce breakpoints in the reward distribution of robot r. Thus, we expect the maximum number of breakpoints $\mathbb{E}[\Upsilon_t]$ to be given by the number of changes to the sample spaces. To ensure convergence of the utility, each period τ_n is governed by a function of t such that $\{\mathbb{E}[\Upsilon_t]\}_t$ is bounded from above. An example definition for τ_n is provided in Remark 2 in Sec. 5.

The set $\hat{\mathcal{X}}_n^r$ has an associated probability distribution q_n^r such that $q_n^r(\boldsymbol{x}^r)$ defines the probability that robot r will select $\boldsymbol{x}^r \in \hat{\mathcal{X}}_n^r$. The distributions for different robots are independent and therefore they collectively define a product distribution q_n, such that the probability p_n of a joint action sequence selection $\boldsymbol{x}$ is

$$p_n(\boldsymbol{x}) = q_n(\boldsymbol{x}) \coloneqq \prod_{r \in \{1,\ldots,R\}} q_n^r(\boldsymbol{x}^r). \tag{9}$$

The advantage of defining p_n as a product distribution is so that each robot selects its action sequence independently, and therefore allows decentralised execution.

Consider the general class of joint probability distributions p_n that are not restricted to product distributions. Define the expected global objective function for a joint distribution p_n as $\mathbb{E}_{p_n}[g]$, and let Γ be a desired value for $\mathbb{E}_{p_n}[g]$. According to the maximum entropy principle, the most likely p_n that satisfies $\mathbb{E}[g] = \Gamma$ is the p_n that maximises entropy. The most likely p_n can be found by minimising the *maxent Lagrangian*, defined as

$$L(p_n) \coloneqq \lambda \left(\Gamma - \mathbb{E}_{p_n}[g]\right) - \mathrm{H}(p_n), \tag{10}$$

where

$$\mathrm{H}(p_n) \coloneqq -\sum_{\boldsymbol{x} \in \mathcal{X}} p_n(\boldsymbol{x}) \ln\left(p_n(\boldsymbol{x})\right) \tag{11}$$

is the Shannon entropy and λ is a Lagrange multiplier. The intuition is to iteratively increase Γ and optimise p_n. A descent scheme for p_n can be formulated with Newton's method.

For decentralised planning and execution, we are interested in optimising the product distribution q_n rather than a more general joint distribution p_n. We can approximate q_n by finding the q_n with the minimum pq KL divergence, where the pq KL divergence is defined as

$$D_{\mathrm{KL}}(p_n \parallel q_n) \coloneqq \sum_{\boldsymbol{x} \in \mathcal{X}} p_n(\boldsymbol{x}) \ln\left(\frac{p_n(\boldsymbol{x})}{q_n(\boldsymbol{x})}\right). \tag{12}$$

This formulates a descent scheme with the update policy for q_n^r shown in Alg. 3, line 5, where we use f^r (as

Algorithm 3 Probability distribution optimisation for robot r.

1: **function** UPDATEDISTRIBUTION($\hat{\mathcal{X}}_n^r, q_n^r, \hat{\mathcal{X}}_n^{(r)}, q_n^{(r)}, \beta$)
   input: action sequence set for each robot $\hat{\mathcal{X}}_n \coloneqq \{\hat{\mathcal{X}}_n^1, \hat{\mathcal{X}}_n^2, \ldots, \hat{\mathcal{X}}_n^R\}$
   with associated probability distributions $\{q_n^1, q_n^2, \ldots, q_n^R\}$,
   update parameter β,
   constant α (e.g., $\alpha = 0.01$),
   output: updated probability distribution q_n^r for robot r
2: **for each** $\boldsymbol{x}^r \in \hat{\mathcal{X}}_n^r$ **do**
3: $\mathbb{E}_{q_n}[f^r] \leftarrow \sum_{\boldsymbol{x} \in \hat{\mathcal{X}}_n} \left[f^r(\boldsymbol{x}) \prod_{r' \in \{1,\ldots,R\}} q_n^{r'}(\boldsymbol{x}^{r'}) \right]$
4: $\mathbb{E}_{q_n}[f^r \mid \boldsymbol{x}^r] \leftarrow \sum_{\boldsymbol{x}^{(r)} \in \hat{\mathcal{X}}_n^{(r)}} \left[f^r(\boldsymbol{x}^r \cup \boldsymbol{x}^{(r)}) \prod_{r' \in \{1,\ldots,R\} \setminus r} q_n^{r'}(\boldsymbol{x}^{r'}) \right]$
5: $q_n^r(\boldsymbol{x}^r) \leftarrow q_n^r(\boldsymbol{x}^r) - \alpha q_n^r(\boldsymbol{x}^r) \left[\frac{\mathbb{E}_{q_n}[f^r] - \mathbb{E}_{q_n}[f^r \mid \boldsymbol{x}^r]}{\beta} + \mathrm{H}(q_n^r) + \ln\left(q_n^r(\boldsymbol{x}^r)\right) \right]$
6: $q_n^r \leftarrow$ NORMALISE(q_n^r)
7: **return** q_n^r

defined in (1)) rather than g, and the expectations $\mathbb{E}_{q_n}$ are defined with respect to the product distribution q_n. Intuitively, this update rule increases the probability that robot r selects $\boldsymbol{x}^r$ if this results in an improved local utility, while also ensuring the entropy of q_n^r does not decrease too rapidly. The former behaviour is controlled by the $\left(\mathbb{E}_{q_n}[f^r] - \mathbb{E}_{q_n}[f^r \mid \boldsymbol{x}^r]\right)/\beta$ term in the update rule, while the latter behaviour is controlled by $\mathrm{H}(q_n^r) + \ln\left(q_n^r(\boldsymbol{x}^r)\right)$. Parameter β specifies the balance between these two behaviours.

Pseudocode for this approach is in Alg. 3. Each iteration of the loop beginning at line 2 updates the probability $q_n^r(\boldsymbol{x}^r)$ of performing an action sequence $\boldsymbol{x}^r$. We require computing two expectations (lines 3–4) to evaluate the update equation (line 5). In general, to compute these expectations exactly it is necessary to sum over the enumeration of all $\boldsymbol{x} \in \hat{\mathcal{X}}_n$. It is infeasible to perform this enumeration at every iteration, and therefore these expectations should instead be approximated using random sampling of $\hat{\mathcal{X}}_n$. For certain problem definitions, it may be possible to efficiently compute these expectations exactly by exploiting the structure of the problem, such as in our Sec. 6 experiments.

As the Dec-MCTS algorithm progresses, the parameter β should slowly decrease in order to slowly decrease the entropy of the probability distributions. The cooling schedule for β could be a fixed rate of descent or a more elaborate schedule (Wolpert et al. 2006). The parameter α is a fixed step size for the gradient descent. When the sample space $\hat{\mathcal{X}}_n^r$ changes (Alg. 1, line 3), theoretically it is possible to keep and update the previous distribution, i.e., $q_n^r = q_{n-1}^r$, by maintaining q_n^r over the entire space $\mathcal{X}^r$. However, in practice, this is likely to become inefficient as the number of action sequences that have ever appeared in a sample space grows, particularly when calculating the expectations and normalising, as well as when communicating these distributions. Instead, we suggest resetting q_n^r to a uniform distribution and β to its initial value whenever $\hat{\mathcal{X}}_n^r$ changes.

4.5 Communication

At each iteration of the inner-loop of Alg. 1, robot r communicates its current probability distribution $(\hat{\mathcal{X}}_n^r, q_n^r)$ to the other robots. If robot r receives an updated distribution $(\hat{\mathcal{X}}_n^{r'}, q_n^{r'})$ from another robot r', then $(\hat{\mathcal{X}}_n^{r'}, q_n^{r'})$ replaces the locally stored distribution for r'. The updated distribution is used during the next iteration, such that both the tree $\mathcal{T}^r$ and probability distribution $(\hat{\mathcal{X}}_n^r, q_n^r)$ are updated based on the new $(\hat{\mathcal{X}}_n^{(r)}, q_n^{(r)})$. If no new messages are received from a robot, then robot r continues to plan based on the most recent distribution. If robot r is yet to receive any messages then it may assume a default policy.

4.6 Online replanning

The best action is selected as the first action in the highest probability action sequence in $\hat{\mathcal{X}}_n^r$ (Alg. 1, line 10). The search tree may then be pruned by removing all children of the root except the selected action. Planning may then continue while using the sub-tree's previous results. If the objective function changes, e.g., as a result of a new observation, then the tree should be restarted. In practice, if the change is minor then it may be appropriate to continue planning with the current tree, and the discounting in D-UCT will help to quickly correct the reward estimates.

4.7 Probabilistic objective functions

So far, we have assumed the objective function g is deterministic for a given set of action sequences $\boldsymbol{x}$. This is reasonable in many scenarios since, for example, it is usually sufficient to plan based on the expectation of the reward, which is a deterministic quantity. However, sometimes it may be necessary to directly model other sources of uncertainty, such as the state of the environment, in addition to the uncertain plans of the robots. For these problems, we can define the objective function as $g(\boldsymbol{x}, \Psi)$, where Ψ is a random variable representing other sources of uncertainty. Our algorithm can readily be extended to this case by computing all expectations with respect to both q_n and Ψ. In some cases these expectations could be computed exactly (this was feasible for our Sec. 7 experiments), but in general the expectations can be efficiently approximated by sampling for Ψ, as in POMCP (Silver and Veness 2010; Patten et al. 2017). Our theoretical analysis (see Sec. 5) is valid for these cases since the standard UCT algorithm assumes the rewards obtained at leaf nodes are probabilistic (Kocsis and Szepesvári 2006), and the standard probability collectives algorithm is applicable if there is noise in the system (Wolpert et al. 2006).

5 Analysis

In this section, we provide a detailed theoretical analysis of Dec-MCTS. The algorithm is an anytime and decentralised approach to multi-robot coordination with two key algorithmic components: (1) the tree search (Sec. 4.3) is designed to perform long-horizon planning for single-robot action sequences while considering the changing plans of the other robots, and (2) the product distribution optimisation (Sec. 4.4) is designed to directly optimise the *joint* multi-robot plan while being restricted to a small subset of possible action sequences. While it is difficult to make any strong claims of global optimality in the context of decentralised, long-horizon planning with general objective functions, we focus our analysis on characterising the convergence properties of these two algorithmic components, then discuss the implications of these results.

In Sec. 5.1 we begin by presenting and analysing a special type of multi-armed bandit (MAB) problem that is related to tree search. Section 5.2 then presents our main analytical result that the D-UCT algorithm (Alg. 2) maintains an exploration-exploitation trade-off for child selection while the distributions q_n^r are changing (and converging). In Sec. 5.3 we characterise the convergence of Alg. 3 given a contracted sample space of distributions $\hat{\mathcal{X}}_n^r \subset \mathcal{X}^r$. Finally, in Sec. 5.4 we remark on the implications of these results in the context of the overall Dec-MCTS algorithm (Alg. 1).

5.1 D-UCB applied to bandits

We begin our analysis by studying D-UCB (Garivier and Moulines 2011) for a specific type of non-stationary, switching bandit problem. The classic MAB problem is that of a gambler deciding which arm to play from a row of slot machines with stationary but unknown reward distributions. As a result, bandits are the canonical model for studying the trade off of acquiring knowledge ("exploration") and maximising reward ("exploitation"), or the *exploration-exploitation dilemma*. In the context of Alg. 2, the "arm" is analogous to a node selected to expand for a given MCTS rollout. We can therefore leverage the analysis of the MAB for the tree search problem (later in Sec. 5.2). To achieve this, we modify the assumptions on the type of reward distributions for each arm to those expected at internal nodes of the tree while performing the proposed D-UCT algorithm.

Our analysis follows that of Kocsis et al. (2006); Kocsis and Szepesvári (2006) who analyse the use of UCB1 as the MCTS node selection policy. We mainly reference the technical report Kocsis et al. (2006) where the proofs for their theorems are given. Specifically, in this section

we analyse D-UCB applied to a special type of bandit problem, then in Sec. 5.2 we exploit this analysis in applying D-UCT to the root node of a tree.

In the remainder of this section, we will first provide an upper bound on the number of pulls of any arm that is suboptimal in Lemma 1. Then, Lemma 2 will bound the difference between the optimal payoff and expected total payoff up to some arbitrary time. Lemma 3 then gives concentration bounds of the actual mean about this expected value. We then give the asymptotic probability of the algorithm failing in Lemma 4. The proofs of these lemmas are provided in Appendix A.

5.1.1 Technical preliminaries We consider D-UCB applied to a particular type of switching bandit problem. Let $I_t \in \{1, \dots, K\}$ denote the arm pulled at round t, with K the number of possible arms. After selecting node $I_t = i$, the gambler receives a stochastic payoff $X_{i,t} \in [0, 1]$. The sequence of payoffs generate the stochastic process $\{X_{i,t}\}_t$, $i = 1, \dots, K$, $t \geq 1$.

The D-UCB arm selection policy uses the same bound (3) as in Sec. 4.3.1. Specifically, given a discount factor $\gamma \in (1/2, 1)$, the D-UCB algorithm chooses the arm with the best discounted UCB:

$$I_t = \underset{i \in \{1,\dots,K\}}{\arg\max} \{\bar{X}_{i,t}(\gamma) + c_{t,t_i}(\gamma)\}, \tag{13}$$

where $\bar{X}_{i,t}(\gamma)$ denotes the discounted average reward (6) and $c_{t,t_i}(\gamma)$ is the bias sequence (7) for arm i at round t. Similar to Sec. 4.3.1, $t_i(\gamma)$ denotes the discounted number of times arm i is pulled (4) and $t(\gamma)$ denotes the discounted total number of pulls (5).

As in Kocsis and Szepesvári (2006), we allow the mean value of the payoffs to drift as a function of time; however, these values can also change dramatically at *breakpoints*. These breakpoints are defined as epochs when a previously suboptimal arm becomes optimal. We denote by Υ_t the number of breakpoints *before* time t. When referring to quantities that are not discounted (i.e., $\gamma = 1$), we remove the γ argument (e.g., $t = t(1)$, $\bar{X}_{i,t} = \bar{X}_{i,t}(1)$, etc.). Further, the filtration $\mathcal{F}$ referred to in this paper is the natural filtration.

Recall we require a number of assumptions on the reward distributions of each arm so that our analysis for this bandit problem can later be exploited in our analysis for D-UCT. Our first assumption relates to the payoff sequence of each arm.

Assumption 1. *Fix $1 \leq i \leq K$. Let $\{\mathcal{F}_{i,t}\}_t$ be a filtration such that $\{X_{i,t}\}_t$ is $\{\mathcal{F}_{i,t}\}$-adapted and $X_{i,t}$ is conditionally independent of $\mathcal{F}_{i,t+1}, \mathcal{F}_{i,t+2}, \dots$ given $\mathcal{F}_{i,t-1}$. Further, there exists an integer T_p such that for $t_i \geq T_p$ and $t < t_i$, $X_{i,t}$ is independent from $\mathcal{F}_{i,t}$.*

As mentioned, we allow the expected value for each arm $\mu_{i,t}$ to drift over time, and change abruptly at a breakpoint. We assume the number of breakpoints are upper bounded as follows.

Assumption 2. *The monotone sequence giving the maximum number of breakpoints up to time t $\{\Upsilon_t\}_t$ is known and bounded, s.t., $\lim_{t\to\infty} \Upsilon_t = \sup_t \Upsilon_t < \infty$ and (by definition) $\Upsilon_{t+1} \geq \Upsilon_t$.*

The number of abrupt changes to $\mu_{i,t}$ are thus bounded by $\sup_t \Upsilon_t$. As the following assumption states, we also assume that the expected payoff converges.

Assumption 3. *The limit $\mu_i = \lim_{t\to\infty} \mu_{i,t}$ exists for all $i \in \{1, \dots, K\}$.*

The difference between the expected reward at time t and the limit is termed the drift $\delta_{i,t} = \mu_{i,t} - \mu_i$. For any arbitrary time t, denote the optimal arm as i_t^*, and define the optimal expected payoff by $\mu_{i_t^*,t} = \max_{i\in\{1,\dots,K\}} \mu_{i,t}$. Thus, we obtain the optimal expected payoff *up to time* t as

$$\mu_t^* = \frac{1}{t} \sum_{u=1}^{t} \mu_{i_u^*,u}.$$

Finally, the minimum difference between the expected reward for an optimal arm i_u^* and expected reward for arm i at all times is obtained as

$$\Delta_{i,t} = \min_{u\in\{1,\dots,t\}} \left\{\mu_{i_u^*,u} - \mu_{i,u} : i \neq i_u^*\right\}.$$

Our last assumption is that we require an index $T_0(\epsilon)$ above which the drift $\delta_{i,t}$ becomes proportional to $\Delta_{i,t}$. Let $M_i(t)$ denote the number of pulls of arm i following the most recent breakpoint.

Assumption 4. *There exists an index $T_0(\epsilon)$ such that, for any arbitrary $\epsilon > 0$ and $M_i(t) \geq T_0(\epsilon)$, $|\delta_{i,t}| \leq \epsilon\Delta_{i,t}/2$ and $|\delta_t^*| \leq \epsilon\Delta_{i,t}/2$ for all i.*

5.1.2 Theoretical analysis Given these assumptions, we now begin our analysis of D-UCB for this bandit problem. First, we bound the number of times each suboptimal arm is pulled. Denote by $\mathbb{P}_\gamma$ and $\mathbb{E}_\gamma$ the probability distribution and expectation under the policy D-UCB using the discount factor γ.

Let $\tilde{T}_i(t) = \sum_{u=1}^{t} \mathbf{1}_{\{I_u = i \neq i_u^*\}}$ be the number of times arm i was played when it was not the best arm in the first t rounds.

Lemma 1. Number of Suboptimal Pulls. *Consider D-UCB applied to a non-stationary, switching bandit problem where Assumptions 1-4 are satisfied and where the bias sequence $c_{t,t_i}(\gamma)$ used by D-UCB is given by* (7)*. Let $C_p > 1/\sqrt{8}$ and $\gamma_t = 1 - \sqrt{\mathbb{E}[\Upsilon_t]/16t}$. For any arm $i \in \{1, \ldots, K\}$ and $t > 1$:*

$$\mathbb{E}_\gamma[\tilde{T}_i(t)] \leq O\left(\sqrt{\mathbb{E}[\Upsilon_t]t}(C_p^2 \log t + T_0(\epsilon) + T_p)\right). \tag{14}$$

The value of C_p stated is more general than the common statement that $C_p = 1/\sqrt{2}$ in, e.g., Kocsis and Szepesvári (2006); Browne et al. (2012). We discuss this further in Remark 3 in Appendix A.1.

The following lemma gives convergence of the expected undiscounted payoff $\mathbb{E}_\gamma[\bar{X}_t]$ received up to time t towards the optimal payoff μ^*. The proof is a simplified version of Theorem 2 of Kocsis et al. (2006) that allows for changing "best arms". The proof uses the expected number of suboptimal pulls (Lemma 1) and the definition of drift δ_t^* to bound the payoff.

Lemma 2. Expected Payoff Convergence Towards Optimal Payoff. *Let*

$$\bar{X}_t := \sum_{i=1}^{K} \frac{T_i(t)}{t} \bar{X}_{i,t}.$$

Under the assumptions of Lemma 1,

$$\left|\mathbb{E}_\gamma[\bar{X}_t] - \mu^*\right| \leq |\delta_t^*| + O\left(K\sqrt{\mathbb{E}[\Upsilon_t]/t}\left(C_p^2 \log t + T_0 + T_p\right)\right), \tag{15}$$

where $T_0 = T_0(1/2)$.

From Lemma 2, we have the convergence of the *expected payoff* $\mathbb{E}_\gamma[\bar{X}_t]$ about the optimal payoff; however, we are yet to obtain results about the concentration of the actual payoff $\bar{X}_t$ about this quantity. To bound this concentration, we leverage the results of Kocsis et al. (2006), which has a non-trivial assumption related to the number of suboptimal pulls. Denote Z_t the indicator variable that a suboptimal arm was pulled. As with Kocsis et al. (2006), from Assumption 1, we have that, for $t \geq T_p$, the indicator Z_t is independent of $Z_{t+1}, Z_{t+2}, \ldots$, given $Z_1, \ldots, Z_{t-1}$. Thus, after T_p and T_0, the non-stationary bandit problem becomes equivalent to a stationary problem with high probability. This allows us to establish the concentration of $\mathbb{E}_\gamma[\bar{X}_t]$ about $\bar{X}_t$ in the following lemma.

Lemma 3. Payoff Convergence Towards Expected Payoff. *Fix an arbitrary $0 < \varepsilon \leq 1$ and let $\Gamma_t = 9C_p\mathbb{E}[\Upsilon_t]t\sqrt{2\log(2/\varepsilon)}$. Then, under the assumptions of Lemma 1, for*

$$t \geq O\left(K\sqrt{\mathbb{E}[\Upsilon_t]t}(C_p^2 \log t + T_0 + T_p)\right),$$

the following bound holds:

$$\mathbb{P}\left(t|\bar{X}_t - \mathbb{E}_\gamma[\bar{X}_t]| \geq \Gamma_t\right) \leq \varepsilon. \tag{16}$$

Next, we are interested in the probability of the algorithm failing. The proof relies on our assumption that the breakpoint sequence is known monotone and bounded, resulting in D-UCB becoming equivalent to UCB1 for large t.

Lemma 4. Convergence of Failure Probability. *Under the assumptions of Lemma 1 it holds that*

$$\lim_{t\to\infty} \mathbb{P}_\gamma(I_t \neq i_t^* = i^*) = 0.$$

5.2 D-UCB applied to trees

We now discuss the application of D-UCB as the node selection policy of MCTS. The assumptions we made about the bandit problem in the above section allows us to analyse convergence of the actual to the optimal payoff sequence at the root node after some transitory period.

Recall that the node selection problem at each node in the tree is equivalent to the bandit problem, however with different assumptions on the payoff received. From the perspective of node i_d, after selecting node $I_{i_d,t} = j$, the tree search further down the tree (e.g., $I_{j,t}$) and

subsequent MCTS rollout yield a stochastic payoff $F_{j,t} = F_t \in [0,1]$ which is adapted to $\mathcal{F}_{j,t}$ (Assumption 1). As nodes are slowly expanded in the tree search, the expected reward at any node higher up the tree slowly drifts until all nodes are explored in the subtree (Assumption 3).

The sequence of payoffs generate the stochastic process $\{F_{j,t}\}_t$, $\forall j \in \mathcal{C}(i_d)$ and $t \geq 1$. We simplify the analysis by assuming a constant branching factor K, i.e., $\mathcal{C}(i_d) = \{1, \ldots, K\}, \forall i_d$.

Applying the above lemmas to the tree $\mathcal{T}^r$, we require some extra notation. Recall that $\bar{F}_{i_d,t_{i_d}}$ is the empirical mean; it follows that $\bar{F}_{i_0,t_{i_0}}$ is the mean at the root node. Further, let $\mu^*_{i_0}$ denote the optimal expected payoff at the root node and note that $t_{i_0} = t$.

Theorem 1. *Consider algorithm D-UCT running on a tree $\mathcal{T}$ of depth D and branching factor K. The payoff distributions of the leaf nodes are independently distributed and can change at breakpoints. The sequence that gives the expected bound of breakpoints $\{\mathbb{E}[\Upsilon_{t_j}]\}$ follows Assumption 2 and $\gamma_{t_j} = 1 - \sqrt{\mathbb{E}[\Upsilon_{t_j}]/16t_j}$ for all nodes j. Then, when $M_{i_0}(t) \geq T_p$ and $M_{i_0}(t) \geq T_0$, the bias of the payoff at the root node,*

$$|\bar{F}_{i_0,t_{i_0}} - \mu^*_{i_0}| = O\left(KD\log(t)\sqrt{\mathbb{E}[\Upsilon_t]/t}\right). \quad (17)$$

Further, the probability of failure at the root node becomes zero as t grows large.

Proof. The proof is done by induction on D.

First, for $D = 1$, running the D-UCB algorithm as the node selection policy is equivalent to running D-UCB on a bandit problem. Thus, the payoffs $F_{i_0,t}$ are i.i.d. ($T_p = 0$) and, comparing Lemma 1 to Theorem 1 of Garivier and Moulines (2011), we deduce that $T_0 = 0$. The expected bias thus follows from Lemma 2, and the concentration of the actual payoff about the expected value follows from Lemma 3. By Assumption 2, the asymptotic probability of failure follows from Lemma 4.

Now, consider a tree of depth D and assume the statement holds up to a tree of depth $D-1$. First, note that, due to our assumptions, Lemmas 1-4 hold for any internal node of the tree. Regarding Assumption 1, before T_p, the payoffs $F_{i_0,t}$ are not independently distributed. Instead, since D-UCB node selection is also used further down the tree ($d > 0$), the payoff is $\{\mathcal{F}_{i_0,t}\}$-adapted. However, there is a point T_p where the payoffs become independent as the tree search problem becomes equivalent to an MAB problem. When $M_{i_0}(t) \geq T_p$ and $M_{i_0}(t) \geq T_0$, it follows by Lemma 2 that the bias at the root converges at the rate of

$$|\bar{F}_{i_0,t_{i_0}} - \mu^*| = |\delta^*_{t_{i_1}}| + O\left(K\log(t)\sqrt{\mathbb{E}[\Upsilon_t]/t}\right), \quad (18)$$

where $\delta^*_{t_{i_1}}$ is the rate of convergence of the bias for the best move. By the induction hypothesis

$$|\delta^*_{t_{i_1}}| = O\left(K(D-1)\log(t)\sqrt{\mathbb{E}[\Upsilon_t]/t}\right), \quad i_1 = 1, \ldots, K.$$

Substituting this result into (18) gives (17). By Assumption 2, the expected number of breakpoints $\lim_{t\to\infty}\mathbb{E}[\Upsilon_t]$ is bounded, and hence by Lemma 4 the probability of failure becomes zero. □

Remark 1. The results presented here are mainly concerned with the convergence of the bias *after some transitory period*. For the standard UCT case, Kocsis et al. (2006) assumed the T_p term was 0 and suggested $T_0 = O(K^D)$. However, it was recently shown that this transitory period using the UCT algorithm on a binary tree ($K = 2$) of depth D can be $\Omega(\exp(\exp(\ldots\exp(1)\ldots)))$ ($D-1$ nested exponentials) in a worst-case instance (Coquelin and Munos 2007). Gelly et al. (2012) suggest instead that the UCT (and thus D-UCT) strategy will be most successful when the leaves of large subtrees share similar rewards, i.e., a "smoothness" assumption on the reward distributions. Active perception scenarios typically exhibit some degree of "smoothness", such that similar sequences of actions yield similar rewards and thus there is a correlation amongst subtree leaves. △

Remark 2. Assumption 2 states several conditions for the breakpoint sequence $\{\mathbb{E}[\Upsilon_t]\}_t$. We can ensure these assumptions are satisfied by selecting appropriate values for the sample period τ_n (used in Sec. 4.4). Here, we provide a concrete example definition for τ_n. Recall that n is the number of times $\hat{\mathcal{X}}_n$ is changed and τ_n is the number of calls to Alg. 2 with sample space $\hat{\mathcal{X}}_n$. Let $c > 0$ denote the fixed number of iterations in Alg. 2 (line 2). For this example, we let $\tau_n = 1/\lfloor at^{-2}\rfloor$, and therefore $n = \lceil a(1-t^{-1})\rceil$ where $a > c$. Therefore, the

expected upper bound on breakpoints $\mathbb{E}[\Upsilon_t] = n/c$ and thus $\lim_{t\to\infty} \mathbb{E}[\Upsilon_t] = a/c$. This ensures Lemma 4 holds and the bias at the root node (17) is

$$|\bar{F}_{i_0,t_{i_0}} - \mu^*_{i_0}| = O\left(KD\log(t)/\sqrt{t}\right).$$

Therefore, D-UCT achieves a polynomial convergence rate, even in problems where the reward distributions are changing abruptly, such as in Dec-MCTS. △

5.3 Variational methods by importance sampling

We now consider the effect of contracting the sample space $\hat{\mathcal{X}}_n \subset \mathcal{X}$ on the convergence of Alg. 3. Recall that the pq KL divergence is the divergence from a product distribution q_n to the optimal joint distribution p_n. We then have the following proposition:

Proposition 1. *Alg. 3 asymptotically converges to a distribution that locally minimises the pq KL divergence, given an appropriate subset $\hat{\mathcal{X}}_n \subset \mathcal{X}$.*

We justify Proposition 1 as follows. Consider the situation where, at each iteration n, we randomly choose a subset $\hat{\mathcal{X}}^r_n \subset \mathcal{X}^r$ for each robot. This approach is equivalent to Monte Carlo sampling of the expected utility and thus the biased estimator is consistent (asymptotically converges to $\mathbb{E}[f^r]$). For tractable computation and faster convergence, in our algorithm we modify the random selection by choosing a sparse set of strategies $\hat{\mathcal{X}}_n$ with the highest expected utility (Sec. 4.4). Although this does not ensure we sample the entire domain $\mathcal{X}$ asymptotically, in practice $q_n(\hat{\mathcal{X}}_n)$ is a reasonably accurate representation of $q_n(\mathcal{X})$, and therefore this gives us an approximation to importance sampling (Wolpert et al. 2006). Variants of Alg. 3 have been shown to converge to a distribution that locally minimises the pq KL divergence under reasonable assumptions, such as an appropriate cooling schedule for β (Wolpert and Bieniawski 2004).

5.4 Analysis of Dec-MCTS

The analyses above show separately that the tree search of Alg. 2 balances exploration and exploitation and that, under reasonable assumptions, Alg. 3 converges to the product distribution that best optimises the joint action sequence. These results provide strong motivation for the use of these components in the algorithm. However, they do not immediately yield a characterisation of optimality for Alg. 1. To prove convergence rates and global optimality, we would need to characterise the co-dependence between the evolution of the reward distributions $\mathbb{E}_{q_n}[f^r \mid \boldsymbol{x}^r]$ used when sampling the tree and the contraction of the sample space $\hat{\mathcal{X}}_n$ used for optimising q_n. This co-dependence is complex due to the cyclic nature of the algorithm and communication of information between robots, and thus it is unlikely that any strong claims for global optimality can be made. However, this is generally not achievable in the context of decentralised, long-horizon planning with general objective functions, as addressed in this paper. Despite this, the following experiments show that the Dec-MCTS algorithm converges rapidly to high-quality solutions in multi-robot active perception scenarios.

6 Experiments: Generalised team orienteering

In this section, we evaluate the performance of our algorithm in an abstract multi-robot information gathering problem. An illustration of the problem and an example solution is shown in Fig. 2. We empirically show convergence, robustness to intermittent communication and a comparison to a centralised variant of MCTS.

6.1 Problem statement

The problem is motivated by tasks where a team of Dubins robots maximally observes a set of features of interest in an environment, given a travel budget (Best et al. 2018). Each feature can be viewed from multiple viewpoints and each viewpoint may be within observation range of multiple features. This formulation generalises the orienteering problem (Vansteenwegen et al. 2011; Gunawan et al. 2016) by combining the set structure of the generalised travelling salesman problem (Noon and Bean 1989) with the budget constraints of the orienteering problem with neighbourhoods (Faigl et al. 2016) extended for multi-agent scenarios (Best et al. 2018).

Robots navigate within a graph representation of an environment with vertices $v_i \in \mathcal{V}$, edges $e_{ij} \coloneqq \langle v_i, v_j \rangle \in \mathcal{E}$ and edge traversal costs c_{ij}. Each vertex v_i represents a location and orientation (x, y, θ) within a square

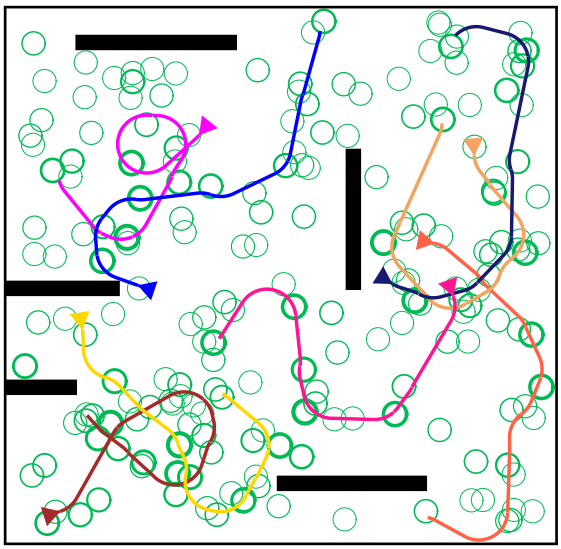

Figure 2. The generalised team orienteering problem. The 8 robots (coloured paths) aim to collectively visit a maximal number of goal regions (green circles, weighted by importance). The robots follow Dubins paths, are constrained by distance budgets and must avoid obstacles (black).

workspace with randomly placed obstacles. The action sequences of each robot are defined as paths through the graph beginning at a start vertex unique to each robot. The edge costs are defined as the distance length of the Dubins path between the two configurations. All edges are connected within a fixed distance.

For the objective function, we have a collection of sets $\mathcal{S} = (S_1, S_2, \ldots)$, where each $S_k \subseteq \mathcal{V}$. These sets may represent a set of features of interest, where a vertex is an element of a set only if the associated feature can be observed from the vertex location. We assume each set is a disc, however the formulation could extend to more complex models (Best et al. 2018). The vertices $v_j \in \mathcal{V}$ are randomly placed within the sets. A set S_k is visited if $\exists v_j \in \boldsymbol{x}, v_j \in S_k$ and each visited set yields an associated reward w_k. There is no additional reward for revisiting a set. The objective is defined as the sum of the rewards of all visited sets.

6.2 Calculating expectations

The Dec-MCTS algorithm requires computing several expectations that, in general, should be approximated using sampling. However, for this problem definition it is possible to exploit the structure of the objective function to efficiently compute exact expectations. We compute expectations in a similar way to (3) by Best and Fitch (2016):

$$\mathbb{E}_{q_n}[g] = \sum_{S_k \in \mathcal{S}} w_k \times \mathbb{P}_{q_n}(\exists v_j \in \boldsymbol{x}, v_j \in S_k)$$

$$= \sum_{S_k \in \mathcal{S}} w_k \left(1 - \prod_{v_j \in S_k} \prod_{\boldsymbol{x}^r \in \hat{\mathcal{X}}_n} \left(1 - q_n^r(\boldsymbol{x}^r)\mathbf{1}_{\{v_j \in \boldsymbol{x}^r\}}\right)\right)$$

where $\mathbb{P}_{q_n}(\exists v_j \in \boldsymbol{x}, v_j \in S_k)$ is the probability that at least one $v_j \in S_k$ is visited by at least one robot. This can be computed much more efficiently than the general equation (linear rather than exponential time in the number of robots) since it only requires iterating over the possible paths for individual robots ($\boldsymbol{x}^r \in \hat{\mathcal{X}}_n^r, \forall r$) rather than iterating over all *joint* paths ($\boldsymbol{x} \in \hat{\mathcal{X}}_n$).

6.3 Experiment setup

We compare our algorithm (Dec-MCTS) to centralised MCTS (Cen-MCTS), which consists of a single tree where robot r's actions appear at tree depths $(r, r+R, r+2R, \ldots)$. Intermittent communication is modelled by randomly dropping messages. Messages are broadcast by each robot at 4 Hz and a message has a probability of being received by each individual robot.

Experiments were performed with 8 simulated robots running in separate ROS nodes on a 4-core computer with hyperthreading (8 virtual cores). Each random problem instance (Fig. 2) consisted of 200 discs with rewards between 1 and 10, 5 obstacles, 4000 graph vertices and random start vertices for each robot. Each iteration of Alg. 1 performs 10 MCTS rollouts and 1 communication broadcast. The set $\hat{\mathcal{X}}_n^r$ consists of 10 paths that are resampled every 10 iterations. The MCTS rollout policy recursively selects the next edge that does not exceed the travel budget and maximises the ratio of the increase of the weighted set cover to the edge cost.

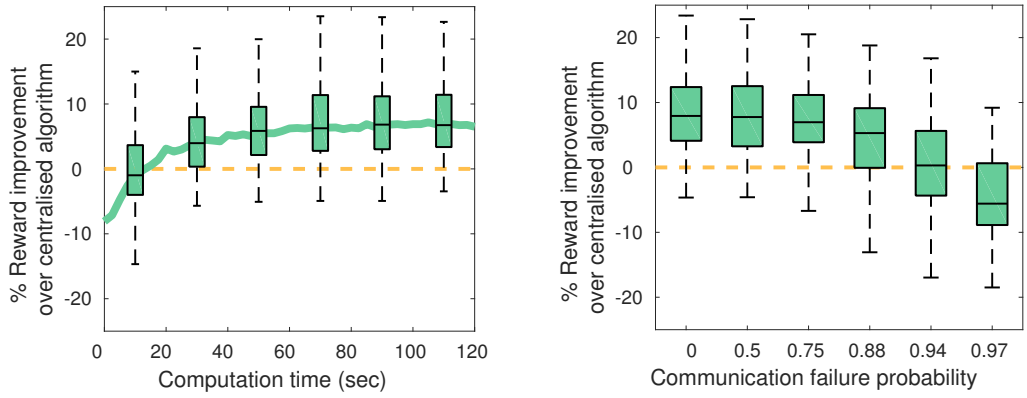

Figure 3. (a) Comparison of Dec-MCTS with varying computation time to Cen-MCTS (120 s). (b) Performance of Dec-MCTS with intermittent communication (60 s computation time). (a,b) Vertical axes show percentage additional reward achieved by Dec-MCTS compared to Cen-MCTS. Error bars show 0, 25, 50, 75 and 100 percentiles (excluding outliers) of 100 random problem instances.

6.4 Results

The first experiments (Fig. 3(a)) show that Dec-MCTS outperforms Cen-MCTS despite the increased difficulty of planning in a decentralised setting. Dec-MCTS achieved a median 7 % reward improvement over Cen-MCTS after 120 s, and a higher reward in 91 % of the environments. Dec-MCTS typically converged after $\sim$60 s. A paired single-tailed t-test supports the hypothesis ($p < 0.01$) that Dec-MCTS achieves a higher reward than Cen-MCTS for time > 7 s. Cen-MCTS performs well initially since it performs a centralised greedy rollout that finds reasonable solutions quickly. Dec-MCTS soon reaches deeper levels of the search trees, though, which allows it to outperform Cen-MCTS. This is because Dec-MCTS uses a collection of search trees with smaller branching factors than Cen-MCTS, but still successfully optimises over the joint space. We note that in this implementation Dec-MCTS is performing parallel computation while Cen-MCTS is mostly sequential. While it is difficult to measure the difference in computation resources used (due to the use of virtual cores, less than 100% processor utilisation, and overheads of using ROS message passing), the results indicate that Dec-MCTS would outperform Cen-MCTS after adjusting for this difference in computation resources.

The second experiments analysed the effect of communication degradation. When the robots did not communicate, the algorithm achieved a median 31 % worse than Cen-MCTS, but with full communication achieved 7 % better than centralised, which shows the robots can successfully cooperate by exploiting the communication channel. Fig. 3(b) shows the results for partial communication degradation. When half of the packets are lost, there is no significant degradation of performance. When 97 % of packets are lost the performance is degraded but the algorithm still performs significantly better than with no communication. These results demonstrate the algorithm is robust to unpredictable communication loss.

7 Experiments: Active object recognition

This section describes experiments for online active object recognition, using point cloud data collected from an outdoor mobile robot in an urban scene (Fig. 4). We first outline the problem and experiment setup, and then present results that analyse the value of online replanning and compare Dec-MCTS to a greedy planner.

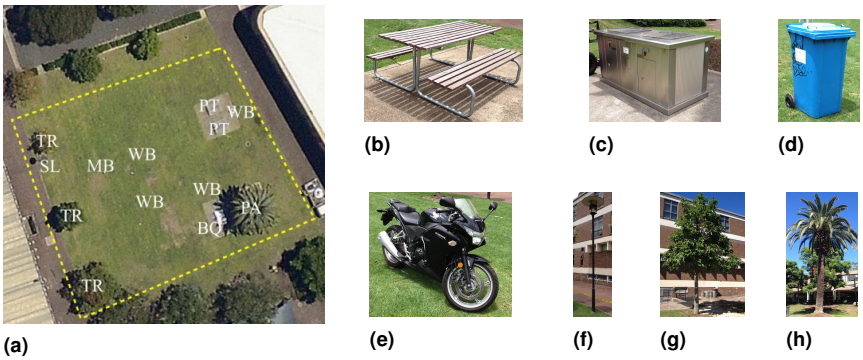

Figure 4. Experiment setup for the point cloud dataset. (a) Environment with labelled locations, (b) picnic table (PT), (c) barbecue (BQ), (d) wheelie bin (WB), (e) motorbike (MB), (f) street light (ST), (g) tree (TR), (h) palm tree (PT).

7.1 Problem setup

A team of robots aim to determine the identity of a set of static objects in an unknown environment. Each robot asynchronously executes the following cycle: (1) plan a path that is expected to improve the perception quality, (2) execute the first planned action, (3) make a point cloud observation using onboard sensors, and then (4) update the belief of the object identities and their poses. Each robot asynchronously performs this cycle until their travel budget is exhausted. The robots have the same Dubins motion model as in Sec. 6. Each graph edge has an additional constant cost that represents the time required to process an observation and perform replanning. Thus, each robot's budget is a constraint on the sum of its travel distance and processing time.

We use a perception model for object recognition similar to that proposed in Ch. 7.2 of Patten (2017). The robots maintain a belief of the identity of each observed object, represented as the probability that each object is an instance of a particular object from a given database. The aim is to improve this belief, which is achieved by maximising the mutual information objective proposed by Patten et al. (2015). The posterior probability distribution for each object after a set of observations is computed recursively using Bayes' rule. The observation likelihood is calculated by measuring the similarity between the shape of the point cloud with each model instance in the database. Similarity is computed by first aligning the point clouds of a pair of objects using the Iterative Closest Point (ICP) algorithm (Besl and McKay 1992) and then calculating the symmetric residual error (Douillard et al. 2012). Objects may merge or split after each observation if the segmentation changes. Observations are fused using decentralised data fusion or a central processor and shared between all robots, and thus all robots are assumed to have the same belief of the environment. While planning, the value of future observations are estimated by simulating observations of objects in the database for all possible object identities, weighted by the belief probabilities, and using maximum likelihood estimates for poses.

7.2 Experiment setup

The experiments use a point cloud dataset (Patten et al. 2015) of Velodyne scans of outdoor objects in a $30\times30\,\mathrm{m}^2$ park shown in Fig. 4(a). The environment consisted of 13 objects from 7 different model types as shown in Figs. 4(b)–(h). The dataset consists of single scans from 50 locations and each scan was split into 8 overlapping observations with different orientations. Each observation had a 180° field of view and 8 m range. These locations and orientations form the roadmap vertices with associated observations. Each object was analysed from separate data to generate the model database. The robots are given a long-range observation from the start location to create an initial belief of most object locations. The

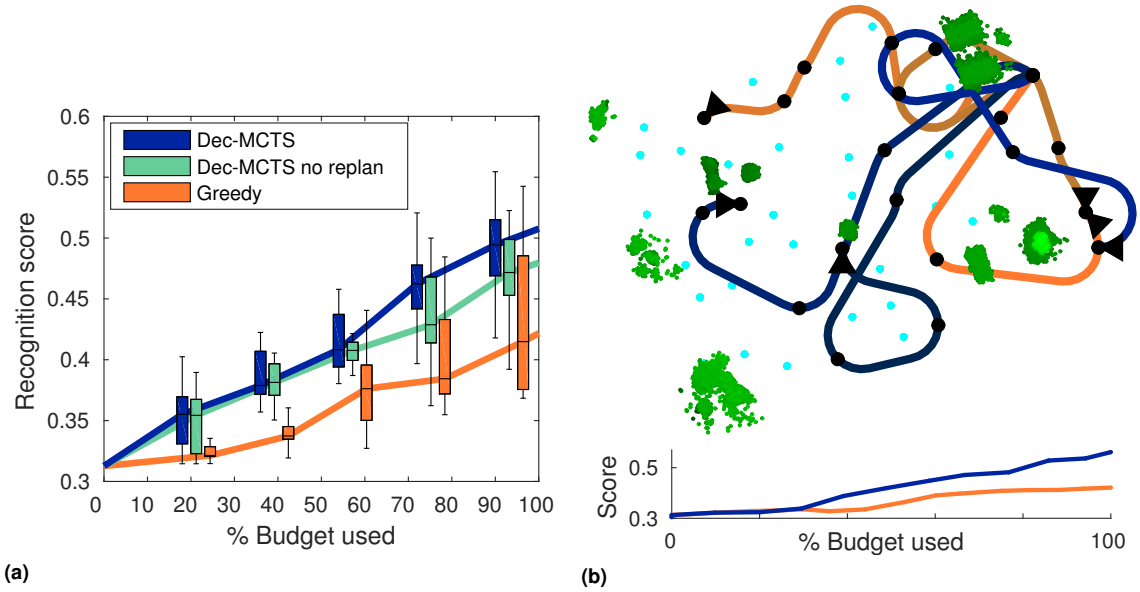

Figure 5. (a) Task performance over mission duration for 10 trials (maximum possible score is 0.62). (b) Overlay of 2 example missions with 3 robots. Blue paths denote online Dec-MCTS (score 0.53). Orange paths denote greedy policy (score 0.42). Objects are green point clouds where shading indicates height. Robots observe at black dots in direction of travel. Start location top right.

team consists of 3 robots, who share a fixed start location with different orientations.

The experiments simulate an online mission where each robot asynchronously alternates between planning and perceiving. Three planners were trialled: our Dec-MCTS algorithm with 120 s replanning after each action, Dec-MCTS without replanning, and a decentralised greedy planner that selects the next action that maximises the mutual information divided by the edge cost. The *recognition score* of an executed path was calculated as the belief probability that each object matched the ground-truth object type, averaged over all objects. The planners cannot directly optimise the paths with respect to the recognition score since the ground-truth is not known in advance; however, planning with respect to the mutual information objective function is intended to indirectly optimise the recognition score.

7.3 Results

Overall, the results validate the coordination performance of Dec-MCTS. Fig. 5(a) shows the recognition score (task performance) over the duration of the mission for 10 trials with 3 robots. The maximum possible recognition score subject to the perception algorithm and dataset was 0.62, which was achieved by visiting every location in the dataset. Dec-MCTS outperformed greedy halfway through the missions since some early greedy decisions and poor coordination reduced the possibility of making subsequent valuable observations. By the end of the missions some greedy plans successfully made valuable observations, but less often than Dec-MCTS. The no-replanning scenario achieved a similar score as the online planner in the first half, showing that the initial plans are robust to changes in the belief. For the second half, replanning improved the recognition score since the belief had changed considerably since the start. This shows that while the generated plans are reasonable for many steps

into the future, there is also value in replanning as new information becomes available.

Fig. 5(b) shows two example missions using online Dec-MCTS (blue) and greedy (orange) planners, and their score over the mission duration. Greedy stayed closer to the start location to improve the recognition of nearby objects, and consequently observed objects on the left less often; reaching this part of the environment would require making high cost/low immediate value actions. On the other hand, Dec-MCTS achieved a higher score since the longer planning horizon enabled finding the high value observations on the left, and was better able to coordinate to jointly observe most of the environment.

8 Conclusion and future work

We have presented a new algorithm for decentralised coordination that is suitable for a general class of problems. Our results demonstrate that the performance (i.e., solution quality) of our approach is as good as or better than its centralised counterpart in real-world applications, and that it effectively optimises over sequences in the joint-action space even with intermittent communication. A key conceptual feature of our approach is its generality in representing joint action sequences probabilistically rather than deterministically. Dec-MCTS has the ability to efficiently plan over long planning horizons, computes anytime solutions, allows incorporating prior knowledge, and provides convergence rate guarantees.

The problem formulation considered in this paper is general in that we are interested in planning sequences of actions to optimise a joint objective function, without requiring assumptions such as submodularity. A straightforward extension to our approach would be to adapt the algorithm to address the Dec-POMDP formulation. This could be achieved by generalising the MCTS component of our algorithm to POMCP (Silver and Veness 2010) while still using our proposed D-UCT tree expansion policy. A difficulty would be to efficiently find good-quality solutions while also considering probabilistic transition models and having the search tree branch for both actions and observations.

Our experiments demonstrate that Dec-MCTS achieves reasonable performance even when the communication becomes less reliable. While these results show a robustness to communication loss, they also indicate that some of the communication messages are not entirely necessary. An interesting avenue for future work would be to develop a communication-planning algorithm that selects when to communicate and who to communicate to while running Dec-MCTS in scenarios with limited communication bandwidth. A possible approach could be a dynamic programming formulation for planning communication to maintain coordination in a similar way to how Ondruska et al. (2015) plans the use of navigation hardware to maintain localisation accuracy. Other communication-planning formulations that may be useful here include Kassir et al. (2015); Lindhé and Johansson (2013). A key difficulty is to develop a measure of information value of a communication message in the context of improving planning performance. Along this line of inquiry, Cliff et al. (2017) introduced measures for quantifying the dynamics of inter-agent dependencies in a team that is optimising a collective goal.

Another interesting line of inquiry is to incorporate coalition forming into our approach. As formulated, static coalitions of agents can be formed by generalising the product distributions in our framework to be partial joint distributions. The product distribution described in Sec. 4.4 would be defined over *groups* of robots rather than individuals. Each group acts jointly, with a single distribution modelling the joint actions of its members, and coordination between groups is conducted as in our algorithm. Just as our approach corresponds to mean-field methods, this approach maps nicely to generalised mean field inference (Xing et al. 2004) or region-based variational methods (Yedidia et al. 2005), and guarantees from these approaches may be applicable. It would also be interesting to study *dynamic* coalition forming, where the mapping between agents and robots is allowed to change, and to develop convergence guarantees for this case. A key challenge would be to determine which robots' plans are more tightly coupled and therefore would benefit from planning within a coalition.

It would be interesting to apply the same general framework to multi-agent scenarios where standard algorithms already exist for associated single-agent scenarios. Problem-specific single-agent planning algorithms could replace the MCTS component of Dec-MCTS, while still

performing the distributed product distribution optimisation phase, in order to provide stronger theoretical guarantees or algorithmic efficiency for special cases. Scenarios where this could be applicable include multi-robot mission monitoring (Best et al. 2017), persistent monitoring (Alamdari et al. 2014), cooperative wildlife localisation (Cliff et al. 2015), travelling salesman problem variants (Best et al. 2018), collision avoidance (Otte and Correll 2013), and dynamic coverage problems (Hönig and Ayanian 2016). It would also be worth investigating other MCTS variants, e.g., BRUE (Feldman and Domshlak 2014), as an alternative to UCT.

Funding

This work was supported in part by the Australian Research Council's Discovery Project funding scheme (No. DP140104203); the Australian Centre for Field Robotics; the New South Wales State Government; the Faculty of Engineering & Information Technologies, The University of Sydney, under the Faculty Research Cluster Program; and The University of Sydney's International Research Collaboration Award.

References

Alamdari S, Fata E and Smith SL (2014) Persistent monitoring in discrete environments: Minimizing the maximum weighted latency between observations. *The International Journal of Robotics Research* 33(1): 138–154.

Amato C (2015) *Decision Making Under Uncertainty: Theory and Application*, chapter Cooperative Decision Making. MIT Press, Cambridge and London.

Arora A, Fitch R and Sukkarieh S (2017) An approach to autonomous science by modeling geological knowledge in a Bayesian framework. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Atanasov N, Ny JL, Daniilidis K and Pappas GJ (2015) Decentralized active information acquisition: Theory and application to multi-robot SLAM. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. pp. 4775–4782.

Auer P, Cesa-Bianchi N and Fischer P (2002) Finite-time analysis of the multiarmed bandit problem. *Machine Learning* 47(2): 235–256.

Auger D (2011) Multiple tree for partially observable Monte-Carlo tree search. In: *Proceedings of European Conference on the Applications of Evolutionary Computation (EvoApplications)*. pp. 53–62.

Bajcsy R (1988) Active perception. *Proceedings of the IEEE* 76(8): 966–1005.

Bajcsy R, Aloimonos Y and Tsotsos JK (2017) Revisiting active perception. *Autonomous Robots* doi:10.1007/s10514-017-9615-3.

Bernstein DS, Givan R, Immerman N and Zilberstein S (2002) The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research* 27(4): 819–840.

Besl PJ and McKay HD (1992) A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14(2): 239–256.

Best G, Cliff O, Patten T, Mettu R and Fitch R (2016) Decentralised Monte Carlo tree search for active perception. In: *Proceedings of International Workshop on the Algorithmic Foundations of Robotics (WAFR)*.

Best G, Faigl J and Fitch R (2018) Online planning for multi-robot active perception with self-organising maps. *Autonomous Robots* doi:10.1007/s10514-017-9691-4.

Best G and Fitch R (2016) Probabilistic maximum set cover with path constraints for informative path planning. In: *Proceedings of ARAA Australasian Conference on Robotics and Automation (ACRA)*.

Best G, Martens W and Fitch R (2017) Path planning with spatiotemporal optimal stopping for stochastic mission monitoring. *IEEE Transactions on Robotics* 33(3).

Binney J and Sukhatme G (2012) Branch and bound for informative path planning. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. pp. 2147–2154.

Bircher A, Kamel M, Alexis K, Oleynikova H and Siegwart R (2016) Receding horizon path planning for 3D exploration and surface inspection. *Autonomous Robots* doi:10.1007/s10514-016-9610-0.

Browne C, Powley E, Whitehouse D, Lucas S, Cowling P, Rohlfshagen P, Tavener S, Perez D, Samothrakis S and Colton S (2012) A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games* 4(1): 1–43.

Charrow B (2015) *Information-theoretic active perception for multi-robot teams*. PhD Thesis, University of Pennsylvania.

Chaslot GMJB, Winands MHM and van den Herik HJ (2008) Parallel Monte-Carlo tree search. In: *Proceedings of International Conference on Computers and Games (CG)*. pp. 60–71.

Choudhury S, Kapoor A, Ranade G, Scherer S and Dey D (2017) Adaptive information gathering via imitation learning. In: *Proceedings of Robotics: Science and Systems*.

Cliff OM, Fitch R, Sukkarieh S, Saunders DL, and Heinsohn R (2015) Online localization of radio-tagged wildlife with an autonomous aerial robot system. In: *Proceedings of Robotics: Science and Systems*.

Cliff OM, Lizier JT, Wang XR, Wang P, Obst O and Prokopenko M (2017) Quantifying long-range interactions and coherent structure in multi-agent dynamics. *Artificial Life* 23(1): 34–57.

Coquelin P and Munos R (2007) Bandit algorithms for tree search. In: *Proceedings of Conference on Uncertainty in Artificial Intelligence (UAI)*. pp. 67–74.

Corah M and Michael N (2017) Efficient online multi-robot exploration via distributed sequential greedy assignment. In: *Proceedings of Robotics: Science and Systems*.

Dias MB, Zlot R, Kalra N and Stentz A (2006) Market-based multirobot coordination: A survey and analysis. *Proceedings of the IEEE* 94(7): 1257–1270.

Douillard B, Quadros A, Morton P, Underwood JP and Deuge MD (2012) A 3D classifier trained without field samples. In: *Proceedings of IEEE International Conference on Control, Automation, Robotics and Vision (ICARCV)*. pp. 805–810.

Dunbabin M and Marques L (2012) Robots for environmental monitoring: Significant advancements and applications. *IEEE Robotics and Automation Magazine* 19(1): 24–39.

Faigl J, Pěnička R and Best G (2016) Self-organizing map-based solution for the orienteering problem with neighborhoods. In: *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. pp. 1315–1321.

Fäulhammer T, Ambruş R, Burbridge C, Zillich M, Folkesson J, Hawes N, Jensfelt P and Vincze M (2017) Autonomous learning of object models on a mobile robot. *IEEE Robotics and Automation Letters* 2(1): 26–33.

Feldman Z and Domshlak C (2014) Simple regret optimization in online planning for Markov decision processes. *Journal of Artificial Intelligence Research* 51: 165–205.

Gan SK, Fitch R and Sukkarieh S (2014) Online decentralized information gathering with spatial–temporal constraints. *Autonomous Robots* 37(1): 1–25.

Garg S and Ayanian N (2014) Persistent monitoring of stochastic spatio-temporal phenomena with a small team of robots. In: *Proceedings of Robotics: Science and Systems*.

Garivier A and Moulines E (2011) On upper-confidence bound policies for switching bandit problems. In: *Proceedings of International Conference on Algorithmic Learning Theory*. pp. 174–188.

Gelly S, Kocsis L, Schoenauer M, Sebag M, Silver D, Szepesvári C and Teytaud O (2012) The grand challenge of computer Go: Monte Carlo tree search and extensions. *Communications of the ACM* 55(3): 106–113.

Gunawan A, Lau HC and Vansteenwegen P (2016) Orienteering problem: A survey of recent variants, solution approaches and applications. *European Journal of Operational Research* 255(2): 315–332.

Hefferan B, Cliff OM and Fitch R (2016) Adversarial patrolling with reactive point processes. In: *Proceedings of ARAA Australasian Conference on Robotics and Automation (ACRA)*.

Hollinger G (2015) Long-horizon robotic search and classification using sampling-based motion planning. In: *Proceedings of Robotics: Science and Systems*.

Hollinger G, Singh S, Djugash J and Kehagias A (2009) Efficient multi-robot search for a moving target. *The International Journal of Robotics Research* 28(2): 201–219.

Hollinger GA and Sukhatme GS (2014) Sampling-based robotic information gathering algorithms. *The International Journal of Robotics Research* 33(9): 1271–1287.

Hönig W and Ayanian N (2016) Dynamic multi-target coverage with robotic cameras. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 1871–1878.

Huber MF, Dencker T, Roschani M and Beyerer J (2012) Bayesian active object recognition via gaussian process regression. In: *Proceedings of IEEE International Conference on Information Fusion (FUSION)*. pp. 1718–1725.

James S, Konidaris G and Rosman B (2017) An analysis of Monte Carlo tree search. In: *Proceedings of AAAI Conference on Artificial Intelligence*.

Kahn G, Sujan P, Patil S, Bopardikar S, Ryde J, Goldberg K and Abbeel P (2015) Active exploration using trajectory optimization for robotic grasping in the presence of occlusions. In: *Proceedings of IEEE International*

Conference on Robotics and Automation (ICRA). pp. 4783–4790.

Kartal B, Godoy J, Karamouzas I and Guy SJ (2015) Stochastic tree search with useful cycles for patrolling problems. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. pp. 1289–1294.

Kassir A, Fitch R and Sukkarieh S (2015) Communication-aware information gathering with dynamic information flow. *The International Journal of Robotics Research* 34(2): 173–200.

Kocsis L and Szepesvári C (2006) Bandit based Monte-Carlo planning. In: *Proceedings of Eurpoean Conference on Machine Learning (ECML)*. pp. 282–293.

Kocsis L, Szepesvári C and Willemson J (2006) Improved Monte-Carlo search. Technical report, University of Tartu.

Koller D and Friedman N (2009) *Probabilistic graphical models: Principles and techniques*. MIT press.

Krause A, Singh A and Guestrin C (2008) Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research* 9: 235–284.

Kulkarni AJ and Tai K (2010) Probability collectives: A multi-agent approach for solving combinatorial optimization problems. *Applied Soft Computing* 10(3): 759–771.

Kumar A, Zilberstein S and Toussaint M (2015) Probabilistic inference techniques for scalable multiagent decision making. *Journal of Artificial Intelligence Research* 53: 223–270.

Lauri M, Atanasov N, Pappas G and Ritala R (2015) Active object recognition via Monte Carlo tree search. In: *Proceedings of IEEE ICRA Workshop on Beyond Geometric Constraints*.

Lauri M and Ritala R (2016) Planning for robotic exploration based on forward simulation. *Robotics and Autonomous Systems* 83: 15–31.

Lindhé M and Johansson KH (2013) Exploiting multipath fading with a mobile robot. *The International Journal of Robotics Research* 32(12): 1363–1380.

Liu L, Michael N and Shell DA (2015) Communication constrained task allocation with optimized local task swaps. *Autonomous Robots* 39(3): 429–444.

Marchant R, Ramos F and Sanner S (2014) Sequential bayesian optimisation for spatial-temporal monitoring. In: *Proceedings of Conference on Uncertainty in Artificial Intelligence (UAI)*.

Nemhauser GL, Wolsey LA and Fisher ML (1978) An analysis of approximations for maximizing submodular set functions–I. *Mathematical Programming* 14(1): 265–294.

Nguyen J, Lawrance N, Fitch R and Sukkarieh S (2015) Real-time path planning for long-term information gathering with an aerial glider. *Autonomous Robots* 40(6): 1017–1039.

Noon CE and Bean JC (1989) An efficient transformation of the generalized traveling salesman problem. Technical Report 89–36, Department of Industrial and Operations Engineering, University of Michigan.

Oliehoek FA and Amato C (2016) *A Concise Introduction to Decentralized POMDPs*. Springer.

Omidshafiei S, Agha-Mohammadi AA, Amato C, Liu SY, How JP and Vian J (2017) Decentralized control of multi-robot partially observable Markov decision processes using belief space macro-actions. *The International Journal of Robotics Research* 36(2): 231–258.

Ondruska P, Gurau C, Marchegiani L, Tong CH and Posner I (2015) Scheduled perception for energy-efficient path following. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. pp. 4799–4806.

Otte M and Correll N (2013) Any-com multi-robot path-planning: Maximizing collaboration for variable bandwidth. In: *Distributed Autonomous Robotic Systems: The 10th International Symposium*. Springer Berlin Heidelberg, pp. 161–173.

Patten T (2017) *Active Object Classification from 3D Range Data with Mobile Robots*. PhD Thesis, The University of Sydney.

Patten T, Fitch R and Sukkarieh S (2013) Large-scale near-optimal decentralised information gathering with multiple mobile robots. In: *Proceedings of ARAA Australasian Conference on Robotics and Automation (ACRA)*.

Patten T, Kassir A, Martens W, Douillard B, Fitch R and Sukkarieh S (2015) A Bayesian approach for time-constrained 3D outdoor object recognition. In: *Proceedings of ICRA 2015 Workshop on Scaling Up Active Perception*.

Patten T, Martens W and Fitch R (2017) Monte Carlo planning for active object classification. *Autonomous Robots* doi:10.1007/s10514-017-9626-0.

Patten T, Zillich M, Fitch R, Vincze M and Sukkarieh S (2016) Viewpoint evaluation for online 3-D active object classification. *IEEE Robotics and Automation Letters* 1(1):

73–81.

Rahmattalabi A, Chung JJ, Colby M and Tumer K (2016) D++: Structural credit assignment in tightly coupled multiagent domains. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 4424–4429.

Regev T and Indelman V (2016) Multi-robot decentralized belief space planning in unknown environments via efficient re-evaluation of impacted paths. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Rezek I, Leslie DS, Reece S, Roberts SJ, Rogers A, Dash RK and Jennings NR (2008) On similarities between inference in game theory and machine learning. *Journal of Artificial Intelligence Research* 33: 259–283.

Sadeghi A and Smith SL (2017) Heterogeneous task allocation and sequencing via decentralized large neighborhood search. *Unmanned Systems* .

Silver D and Veness J (2010) Monte-Carlo planning in large POMDPs. In: Lafferty JD, Williams CKI, Shawe-Taylor J, Zemel RS and Culotta A (eds.) *Advances in Neural Information Processing Systems 23*. Curran Associates, Inc., pp. 2164–2172.

Singh A, Krause A, Guestrin C and Kaiser WJ (2009) Efficient informative sensing using multiple robots. *Journal of Artificial Intelligence Research* 34(1): 707–755.

Slade P, Culbertson P, Sunberg Z and Kochenderfer MJ (2017) Simultaneous active parameter estimation and control using sampling-based Bayesian reinforcement learning. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Somani A, Ye N, Hsu D and Lee WS (2013) DESPOT: Online POMDP planning with regularization. In: Burges CJC, Bottou L, Welling M, Ghahramani Z and Weinberger KQ (eds.) *Advances in Neural Information Processing Systems 26*. Curran Associates, Inc., pp. 1772–1780.

Spaan MTJ, Gordon GJ and Vlassis N (2006) Decentralized planning under uncertainty for teams of communicating agents. In: *Proceedings of International Joint Conference on Autonomous Agents and Multiagent Systems*.

Stranders R, Farinelli A, Rogers A and Jennings N (2009) Decentralised coordination of mobile sensors using the max-sum algorithm. In: *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*. pp. 299–304.

van Hoof H, Kroemer O and Peters J (2014) Probabilistic segmentation and targeted exploration of objects in cluttered environments. *IEEE Transactions on Robotics* 30(5): 1198–1209.

Vansteenwegen P, Souffriau W and Oudheusden DV (2011) The orienteering problem: A survey. *European Journal of Operational Research* 209(1): 1–10.

Waldock A and Nicholson D (2007) Cooperative decentralised data fusion using probability collectives. In: *Proceedings of AAMAS Workshop on Agent Technology for Sensor Networks*.

Wolpert DH and Bieniawski S (2004) Distributed control by Lagrangian steepest descent. In: *Proceedings of IEEE Conference on Decision and Control (CDC)*. pp. 1562–1567.

Wolpert DH, Bieniawski SR and Rajnarayan DG (2013) *Handbook of Statistics 31: Machine Learning: Theory and Applications*, chapter Probability collectives in optimization. Elsevier, pp. 61–99.

Wolpert DH, Strauss CEM and Rajnarayan D (2006) Advances in distributed optimization using probability collectives. *Advances in Complex Systems* 09(04): 383–436.

Wu K, Ranasigne R and Dissanayake G (2015) Active recognition and pose estimation of household objects in clutter. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. pp. 4230–4237.

Xing EP, Jordan MI and Russell S (2004) Graph partition strategies for generalized mean field inference. In: *Proceedings of Conference on Uncertainty in Artificial Intelligence (UAI)*. pp. 602–610.

Xu Z, Fitch R, Underwood J and Sukkarieh S (2013) Decentralized coordinated tracking with mixed discrete-continuous decisions. *Journal of Field Robotics* 30(5): 717–740.

Yedidia JS, Freeman WT and Weiss Y (2005) Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory* 51(7): 2282–2312.

Yu J, Schwager M and Rus D (2016) Correlated orienteering problem and its application to persistent monitoring tasks. *IEEE Transactions on Robotics* 32(5): 1106–1118.

Zlot R, Stentz A, Dias M and Thayer S (2002) Multi-robot exploration controlled by a market economy. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. pp. 3016–3023.

A Appendices

A.1 Proofs

A.1.1 Proof of Lemma 1

Proof. We follow the proof of Theorem 1 of Garivier and Moulines (2011), with minor modifications to account for the transitory periods T_0, T_p discussed in Theorem 2 of Kocsis et al. (2006). Note that in order to simplify notation, we substitute temporal functions (e.g., $u_i(\gamma)$ for $t_i(\gamma)$) when the index u is used instead of t.

Fix the index i of a suboptimal arm. Let

$$A_0(t,\epsilon,\gamma) = \min\{t_i(\gamma) \mid c_{t,t_i}(\gamma) \leq (1-\epsilon)\Delta_{i,t}/2\}.$$

Thus, by the definition of $c_{t,t_i}(\gamma)$,

$$A_0(t,\epsilon,\gamma) = \frac{16C_p^2 \log t(\gamma)}{(1-\epsilon)^2\Delta_{i,t}^2}.$$

We let $A(t,\epsilon,\gamma) = \max(A_0(t,\epsilon,\gamma), T_0(\epsilon), T_p)$. Then the number of times a suboptimal arm i is played is:

$$\begin{aligned}\tilde{T}_i(t) = 1 &+ \sum_{u=K+1}^{t} \mathbf{1}_{\{u:(I_u=i\neq i_u^*)\wedge(u_i(\gamma)<A(u,\epsilon,\gamma))\}} \\ &+ \sum_{u=K+1}^{t} \mathbf{1}_{\{u:(I_u=i\neq i_u^*)\wedge(u_i(\gamma)\geq A(u,\epsilon,\gamma))\}}.\end{aligned}$$

Further, let

$$D(\gamma) = \frac{\log\left((1-\gamma)C_p^2 \log\left(K(\gamma)\right)\right)}{\log(\gamma)}.$$

From Garivier and Moulines (2011), we have

$$\begin{aligned}&\tilde{T}_i(t) \leq 1 + \lceil (1-\gamma)t \rceil A(t,\epsilon,\gamma)\gamma^{-1/(1-\gamma)} \\ &+ \Upsilon_t D(\gamma) + \sum_{u=K+1}^{t} \mathbf{1}_{\{u:(I_u=i\neq i_u^*)\wedge(u_i(\gamma)\geq A(u,\epsilon,\gamma))\}},\end{aligned} \tag{19}$$

for any positive $A(t,\epsilon,\gamma)$ and $D(\gamma)$. As in Garivier and Moulines (2011), there are three conditions under which a suboptimal arm will be played when $t_i(\gamma) \geq A(t,\epsilon,\gamma)$ (following a breakpoint):

$$\begin{aligned}&\{t : (I_t = i \neq i_t^*) \wedge (t_i(\gamma) \geq A(t,\epsilon,\gamma))\} \\ &\subseteq \begin{cases}\{t : (\mu_t^* - \mu_{i,t} < 2c_{t,t_i}(\gamma)) \wedge (t_i(\gamma) \geq A(t,\epsilon,\gamma))\} \\ \cup\{t : \bar{X}_t^*(\gamma) \leq \mu_t^* - c_{t,t_{i^*}}(\gamma)\} \\ \cup\{t : \bar{X}_{i,t}(\gamma) \geq \mu_{i,t} + c_{t,t_i}(\gamma)\}.\end{cases}\end{aligned}$$

We will start with the first case, following the logic of Theorem 2 in Kocsis et al. (2006). Since $c_{t,t_i}(\gamma)$ decreases in t_i and $t_i(\gamma) \geq A(t,\epsilon,\gamma) \geq A_0(t,\epsilon,\gamma)$, we have that $c_{t,t_i}(\gamma) \leq c_{t,A_0(t,\epsilon,\gamma)}(\gamma)$ and thus for the choice of $A_0(t,\epsilon,\gamma)$,

$$c_{t,t_i}(\gamma) \leq 2\sqrt{C_p^2 \log t(\gamma)/A_0(t,\epsilon,\gamma)} \leq \Delta_{i,t}/2.$$

Thus, the first case can not occur when $t_i(\gamma) \geq A_0(t,\epsilon,\gamma)$. Now, when $t_i(\gamma) \geq T_0(\epsilon)$, we have that $|\delta_{i,t}| \leq \epsilon\Delta_{i,t}/2$. Since $\mu^* - \mu_i \geq \Delta_{i,t}, t = 1,2,\ldots$, we have

$$\begin{aligned}\mu_t^* - \mu_{i,t} - 2c_{t,t_i}(\gamma) &\geq \Delta_{i,t} - |\delta_t^*| - \delta_{i,t} - 2c_{t,t_i}(\gamma) \\ &\geq \Delta_{i,t} - \epsilon\Delta_{i,t} - (1-\epsilon)\Delta_{i,t} \\ &= 0.\end{aligned}$$

Thus, the set is empty (i.e., event $(\mu_t^* - \mu_{i,t} < 2c_{t,t_i}(\gamma)) \wedge (t_i(\gamma) \geq A(t,\epsilon,\gamma))$ never occurs).

We now examine the probability of the second and third cases occurring. Recall that $M_i(t)$ denotes the number of pulls of arm i after the most recent breakpoint. Then, under Assumption 1, when $M_i(t) \geq T_p \leq A(t,\epsilon,\gamma)$ we can exploit Theorem 18 of Garivier and Moulines (2011) to complete the proof. The probability of poorly estimating the mean payoffs is now upper bounded as (Garivier and Moulines 2011)

$$\begin{aligned}&\mathbb{P}_\gamma\left(\bar{X}_{i,t}(\gamma) \geq \mu_{i,t} + c_{t,t_i}(\gamma)\right) \\ &\quad\leq (1-\gamma)^{-1} - K + \left\lceil \frac{\log\frac{1}{1-\gamma}}{\log(1+\eta)} \right\rceil \frac{(1-\gamma)t}{1-\gamma^{1/(1-\gamma)}}\end{aligned}$$

for all positive η. Substituting this result into (19) and taking expectations of both sides gives (Garivier and Moulines 2011)

$$\mathbb{E}_\gamma[\tilde{T}_i(t)] \leq C_1(1-\gamma)t + C_2\frac{\mathbb{E}[\Upsilon_t]}{1-\gamma}\log\frac{1}{1-\gamma}, \tag{20}$$

where

$$C_1 = \frac{32\sqrt{2}C_p^2 \log \frac{1}{1-\gamma}}{(1-\epsilon)^2 \Delta_{i,t}^2 \gamma^{1/(1-\gamma)}} + \frac{T_0(\epsilon)}{2\sqrt{2}} + \frac{4}{(1-\frac{1}{e})\log\left(1+4\sqrt{1-1/2C_p^2}\right)}$$

and

$$C_2 = \frac{\gamma - 1}{\log(1-\gamma)\log\gamma} \times \log(1-\gamma) C_p^2 \log K(\gamma).$$

When γ goes to 1, $C_2 \to 1$ and

$$C_1 \to \frac{16eC_p^2 \log \frac{1}{1-\gamma}}{(1-\epsilon)^2 \Delta_{i,t}^2} + T_0(\epsilon) + T_p + \frac{2}{(1-\frac{1}{e})\log\left(1+4\sqrt{1-1/2C_p^2}\right)}.$$

Finally, we can minimise the expected number of times a suboptimal action is taken by setting the discount factor to $\gamma_t = 1 - \sqrt{\mathbb{E}[\Upsilon_t]/16t}$. Selecting this discount factor gives $\mathbb{E}_\gamma[\tilde{T}_i(t)] = O\left(\sqrt{\mathbb{E}[\Upsilon_t]t}(C_p^2 \log t + T_0(\epsilon) + T_p)\right)$ and thus we obtain the bound (14) for $t > 1$. □

Remark 3. A common misconception is that the parameter C_p should be set to $1/\sqrt{2}$ in order to satisfy the Chernoff-Hoeffditg bound (Kocsis and Szepesvári 2006; Browne et al. 2012). However, in the analysis by Auer et al. (2002) and Kocsis et al. (2006), setting C_p to $1/\sqrt{2}$ simply allows the tail inequality to be bounded by t^{-4} and thus converge (Auer et al. 2002). Alternatively, we can select any positive C_p to ensure that the tail inequality is bounded by a negative exponent on t. As a result, we leave the value $C_p > 1/\sqrt{8}$. △

A.1.2 Proof of Lemma 2

Proof. The proof is a slightly generalised version of Theorem 3 of Kocsis et al. (2006) to allow for switching optimal arms.

Without loss of generality we assume that there is a unique "best arm" at any given time t. We denote the index of this arm by i_t^*. By the triangle inequality, $\left|\mu^* - \mathbb{E}_\gamma[\bar{X}_t]\right| \leq |\mu^* - \mu_t^*| + \left|\mu_t^* - \mathbb{E}_\gamma[\bar{X}_t]\right| = |\delta_t^*| + \left|\bar{\mu}_t^* - \mathbb{E}_\gamma[\bar{X}_t]\right|$. We bound the last term as follows:

$$\begin{aligned} t\left|\mu_t^* - \mathbb{E}_\gamma[\bar{X}_t]\right| &= \left|\sum_{u=1}^t \mathbb{E}_\gamma[X_u^*] - \mathbb{E}_\gamma\left[\sum_{i=1}^K T_i(t)\bar{X}_{i,t}\right]\right| \\ &= \left|\sum_{u=1}^t \mathbb{E}_\gamma[X_u^*] - \mathbb{E}_\gamma\left[T^*(t)\bar{X}_t^*\right]\right| \\ &\quad + \mathbb{E}_\gamma\left[\sum_{i=1}^K \tilde{T}_i(t)\bar{X}_{i,t}\right], \end{aligned}$$

since $0 \leq \bar{X}_{i,t} \leq 1$, the last term is bounded by $O(K\sqrt{\mathbb{E}[\Upsilon_t]/t}(C_p^2 \log t + T_0 + T_p))$. Again, since $X_{i,t} \leq 1$, we can deduce that $\mathbb{E}_\gamma\left[T^*(t)\bar{X}_t^*\right] \leq \mathbb{E}_\gamma\left[T^*(t)\right] \leq \sum_{u=1}^t \mathbb{E}_\gamma[X_u^*] \leq t$ and upper bound the first term by

$$\begin{aligned} \mathbb{E}_\gamma[t - T^*(t)] &= \sum_{i=1}^K \mathbb{E}_\gamma\left[\tilde{T}_i(t)\right] \\ &= O(K\sqrt{\mathbb{E}[\Upsilon_t]/t}(C_p^2 \log t + T_0 + T_p)). \end{aligned}$$

Collecting terms yields the bound in (15). □

A.1.3 Proof of Lemma 3

Proof. We modify the proof of Theorem 5 of Kocsis et al. (2006) slightly by using Lemma 5 in Appendix A.2 to account for the bound on $\mathbb{E}[\tilde{T}_i(t)]$.

Using the same notation as Kocsis et al. (2006), Z_t is the indicator variable that a suboptimal arm was pulled at time t. It is important to note that this result follows by letting $t_i \geq T_p$, i.e., we are concerned with the process $\{Z_t\}_{t \geq T_p}$. At this stage, Z_t is independent of $Z_{t+1}, \ldots, Z_t$, given $Z_1, \ldots, Z_{t-1}$ and thus Lemma 11 of Kocsis et al. (2006) holds. Then, following Theorem 5 of Kocsis et al. (2006), it will suffice to prove that there exists a t such that $a_t \leq (2/9)\Gamma_t$ and $R_t \leq (4/9)\Gamma_t$.

By Lemma 1, $\mathbb{E}[\sum_{u=1}^t Z_u] \leq O(K\sqrt{\mathbb{E}[\Upsilon_t]/t}(\log t + T_0 + T_p))$, hence $a_t, R_t = O(K\sqrt{\mathbb{E}[\Upsilon_t]/t}(\log t + T_0 + T_p))$. Thus, since $\Gamma_t = O(\mathbb{E}[\Upsilon_t]t)$ and $a_t, R_t = O(\sqrt{\mathbb{E}[\Upsilon_t]t}\log t)$, the index t exists. □

A.1.4 Proof of Lemma 4

Proof. Since $\lim_{t\to\infty} \Upsilon_t = \sup_t \Upsilon_t < \infty$ and $\lim_{t\to\infty} \gamma_t = 1$, for large t w.l.o.g. we have a unique

"best arm" $i^* = i_t^*$ and the algorithm becomes UCB1 applied to a non-stationary bandit problem. Theorems 4 and 6 of Kocsis et al. (2006) yields the result. □

A.2 Technical Results

Lemma 5. *Let $Z_i, \mathcal{F}_i, a_i$ be as in Lemma 13 of Kocsis et al. (2006). Let $\{X_i\}$ be an i.i.d. sequence with mean μ, and $\{Y_i\}$ an $\mathcal{F}_i$-adapted process. We assume that both X_i and Y_i lie in the $[0,1]$ interval. Consider the partial sums*

$$S_t = \sum_{u=1}^{t} (1 - Z_u) X_u + Z_u Y_u.$$

Fix an arbitrary $0 < \varepsilon \leq 1$, let $\Gamma_t = 9\mathbb{E}[\Upsilon_t] t \sqrt{2\log(2/\varepsilon)}$ and let

$$R_t = \mathbb{E}\left[\sum_u X_u\right] - \mathbb{E}[S_t].$$

Then for t such that $a_t \leq (2/9)\Gamma_t$ and $|R_t| \leq (4/9)\Gamma_t$,

$$\mathbb{P}(|S_t - \mathbb{E}[S_t]| \geq \Gamma_t) \leq \varepsilon \tag{21}$$

Proof. The proof follows Lemma 14 of Kocsis et al. (2006).

We will show that $\mathbb{P}(S_t - \mathbb{E}[S_t] \geq \Gamma_t) \leq \varepsilon$ as $\mathbb{P}(S_t - \mathbb{E}[S_t] \leq \Gamma_t) \leq \varepsilon$ is proved analogously. Let $p = \mathbb{P}(S_t \geq \mathbb{E}[S_t] + \Gamma_t)$. We have $S_t = \sum_{u=1}^t X_u + \sum_{u=1}^t Z_u(Y_u - X_u) \leq \sum_{u=1}^t X_u + 2\sum_{u=1}^t Z_u$. Therefore

$$p \leq \mathbb{P}\left(\sum_{u=1}^{t} X_u + 2\sum_{u=1}^{t} Z_u \geq \mathbb{E}\left[\sum_{u=1}^{t} X_u\right] - R_t + \Gamma_t\right).$$

Using the inequality $\mathbb{I}(A + B \geq \Gamma) \leq \mathbb{I}(A \geq \alpha\Gamma) + \mathbb{I}(B \geq (1-\alpha)\Gamma)$ that holds for any $A, B \geq 0, 0 \leq \alpha \leq 1$ we get

$$p \leq \mathbb{P}\left(\sum_{u=1}^{t} X_u \geq \mathbb{E}\left[\sum_{u=1}^{t} X_u\right] + (1/9)\Gamma_t\right) + \mathbb{P}\left(2\sum_{u=1}^{t} Z_u \geq (8/9)\Gamma - R_t\right).$$

Using the Hoeffding-Azuma inequality, the first term can be bounded by

$$\begin{aligned}\mathbb{P}\left(\sum_{u=1}^{t} X_u \geq \mathbb{E}\left[\sum_{u=1}^{t} X_u\right] + (1/9)\Gamma_t t\right) &\leq \exp\left(-\frac{2(\Gamma_t/9)^2}{t}\right) \\ &= (\varepsilon/2)^{4t} \\ &\leq \varepsilon/2,\end{aligned}$$

for $n \geq 1$ and $0 < \varepsilon < 1$. Since by assumption $|R_t| \leq (4/9)\Gamma$, the second term can be upper bounded by

$$\mathbb{P}\left(2\sum_{u=1}^{t} Z_u \geq (4/9)\Gamma_t\right) = \mathbb{P}\left(\sum_{u=1}^{t} Z_u \geq (2/9)\Gamma_t\right).$$

By Lemma 13 of Kocsis et al. (2006), this term is bounded by $(\varepsilon/2)^n \leq \varepsilon/2$ for $t \geq 1$ and $0 < \varepsilon < 1$. Collecting terms yields the first inequality (21). □