Planning Search and Rescue Missions for UAV Teams

Chris A. B. Baker and **Sarvapali Ramchurn** and **W. T. Luke Teacy**[1] and **Nicholas R. Jennings**[2]

Abstract. The coordination of multiple Unmanned Aerial Vehicles (UAVs) to carry out aerial surveys is a major challenge for emergency responders. In particular, UAVs have to fly over kilometre-scale areas while trying to discover casualties as quickly as possible. To aid in this process, it is desirable to exploit the increasing availability of data about a disaster from sources such as crowd reports, satellite remote sensing, or manned reconnaissance. In particular, such information can be a valuable resource to drive the planning of UAV flight paths over a space in order to discover people who are in danger. However challenges of computational tractability remain when planning over the very large action spaces that result. To overcome these, we introduce the survivor discovery problem and present as our solution, the first example of a continuous factored coordinated Monte Carlo tree search algorithm. Our evaluation against state of the art benchmarks show that our algorithm, Co-CMCTS, is able to localise more casualties faster than standard approaches by 7% or more on simulations with real-world data.

1 Introduction

The increased prevalence of low-cost, robust, commercially available Unmanned Aerial Vehicles (UAVs) has led to concerted efforts to utilise these platforms in order to aid first responders with collecting sensory data without putting human lives at risk [1]. In particular, work has focused on developing autonomous systems to minimise the involvement of overstretched first responder personnel, and to ensure UAVs can take action quickly [22, 24]. Key to this work, is the idea of enabling coordinated UAVs to explore a disaster space to discover the spatial location of casualties: a difficult task given the spatial extent to explore and the continuous action-space represented by a UAV's axes of motion.[3]

To enable this exploration, advances in data collection have created new sources of information about disaster scenarios that contribute to increased awareness of the situation on the ground during a disaster. In particular, data gathered by crowd-sourcing is becoming more readily available because of the speed with which it can be generated, and its ability to directly reflect the experiences of the people on-scene; people who can often give a very accurate report on the hazards in their vicinity and the number of potential casualties [21]. However, despite these advancements, at present there is little work that seeks to use information on *danger* and the spatial locations of *people* to inform the paths of UAVs through a disaster space, in order to maximise the number of observations made of possible casualties.

Currently, the state of the art for UAV path planning algorithms focus on areas such as target tracking for surveillance [5, 17] which, although related to the exploration of a disaster space, are designed to find a known number of targets that are in motion, rather than an unknown number of survivors distributed over an area: a very different problem since algorithms must be able to make predictive estimations of people that *might* be present in as-yet unvisited locations. Other developments in path planning focus on trying to reach a set goal location [9, 19], or working with single autonomous vehicles [8, 18]; neither of which fulfil the need for algorithms that coordinate multiple vehicles in an *explorative* traversal of the disaster space, rather than aiming for a particular final location. Moreover, specific challenges exist in coordinating multiple vehicles. For example, there is often no benefit to multiple UAVs providing sensor data of the same location: there must be coordination between the vehicles to allow them to find survivors in a disaster, without all attending the same locations. Additionally, UAVs must be able to coordinate over actions temporally: visiting the same location at the same time might be straightforward to avoid, but planning a UAVs current action to account for the future action of other nearby UAVs is a non-trivial problem, particularly as the number of UAVs in an environment increases.

Against this background, some work has recently been performed on planning problems that involve the coordination of multiple vehicles, including in disaster scenarios [2, 4]. However, as yet both these approaches require some degree of simplification before planning can commence by discretising the environment into a number of *cells* to be examined. Since locations in the real-world are not discretised in this way, this requires some additional processing of incoming data before UAVs can begin their exploration. Furthermore, by simplifying data in this way it is inevitable that some information is lost when a UAV can only be considered as visiting single cells, rather than being able to plan according to a *continuous* range of motion.

In this paper we seek to address these shortcomings in planning UAV searches of disaster areas, specifically with a view to future applications and field-tests as part of the MOSAIC collaboration[4] and ongoing work with Rescue Global.[5] In particular we make the following contributions to the state of the art:

1. We introduce a novel formulation of the *survivor discovery problem*; specifically modelled on a likely real-world scenario with the goal of locating an unknown number of people, over a wide area, by detection of mobile phone signals and where the diminishing survivability of the people with time is incorporated into the reward function.
2. We develop a novel decentralised algorithm that allows multiple UAVs to coordinate to explore a large continuous disaster space

[1] Agents Interactions and Complexity Group, University of Southampton, UK, email {cabb1g08, sdr, wtlt}@ecs.soton.ac.uk

[2] Department of Computing, Imperial College London, UK, email n.jennings@imperial.ac.uk

[3] Although this work is framed in terms of disaster response, the same coordination algorithms could be applied to other UAV applications: such as geophysical surveying, security operations, or ecological monitoring.

[4] An EPSRC funded collaboration between the University of Southampton and the University of Sheffield: `http://www.mosaicproject.info`

[5] Information available at: `http://www.rescueglobal.org`

under spatial and temporal constraints. Our approach utilises a *belief map*—a term referred to the mapping of spatial locations onto some function that represents numerical data—to represent the presence of people and danger in the environment, and to form the basis of the rewards calculated in the planning algorithm.

3. Furthermore, in order to demonstrate the applicability of our scenario to potential future disasters, we test and evaluate our approach on real-world data (gathered from the 2010 Haiti earthquake) simulating a very large action space, showing consistent gains in survivor discovery of at least 7% compared to benchmarks, with higher gains of around 20% for scenarios with additional UAVs.

In the following sections we first outline the current state of the art areas of research in autonomous path-planning and exploration. Next, we introduce the specific formulation of the environment model and UAV behaviour considered in our simulations, before introducing the continuous form of our coordinated Monte Carlo tree-search algorithm. Following this, we present empirical evidence to substantiate the benefits of this approach, before concluding and discussing future work and applications we will explore.

2 Background

In order to best use UAVs to aid responders in disasters they must be able to plan paths autonomously, as a group, in a decentralised manner. This is particularly important given the implications of a UAV failing in the field: any centralised system that relies on a single UAV (or other central coordinator) will fail entirely if that central point fails; whereas a decentralised system can—in principle—continue to function as UAVs are removed. Furthermore, as we have already indicated, it is beneficial to use prior information about the area to inform the flight paths of UAVs in order to maximise the likelihood of discovering survivors. Currently, work on path planning in robotics focusses primarily on reaching goal locations and frequently formulates path planning as a control problem [14]. Conversely, in a disaster scenario there need not be any final end-point to a UAV's path planning; rather the length of the exploration may be constrained by—for example—battery life, and the number of people to be discovered must be maximised over the length of the path. Alternatively, much work has also been done to enable the use of vision algorithms and belief data to track mobile targets or map an area [20]. However, this area of research often focusses on locating a known number of targets, or covering a bounded space for the purposes of mapping. These foci are not relevant in a scenario where an unknown number of people are distributed over an already-mapped (but very large) area, as is frequently the case with displaced populations during and after disasters.

Against this background, we find closer similarities with work on solving Markov Decision Processes (MDPs); specifically where locality of UAVs can be used to reduce calculation overheads. In particular, the generality of MDP formulations suits our construction of a simulation environment where we are provided with numerical data used for a belief map; and MDPs in general have a number of well-explored algorithmic solutions available. Specifically, work by [2] utilises factored tree-searches for partially observable MDP solutions; exploiting problem structure to allow factorisation in a way reflective of local state spaces and interactions. In a similar way, we use factored trees in this paper to represent the available actions of UAVs in a disaster environment, factoring the value of locating people between UAVs within spatial proximity of each other. However, our work deals with a continuous state-space (in this case representing the continuous range of actions available to a UAV at any given time); as well as a detailed model of the scenario specifically geared towards UAV applications. This requires significant changes to the existing approach in [2] and [4], primarily in applying sampling of the continuous action-space while still allowing UAVs to coordinate with one another. Crucially, discrete approaches to MDP solutions require some form of recognition that a particular set of actions has been entirely explored: clearly where a continuum of actions exists this cannot be said to be true, since the number of individual actions available for selection is infinite. As such we have had to adapt different approaches usually applied to planners dealing with continuous spaces [11, 25] into the regime of coordinated exploration.

3 Scenario Model

The overarching aim of disaster response work is to minimise the loss to human life in the disaster area. Currently, we perceive there to be a lack of suitable environment models that fully characterise this problem and express it in terms of sensing technology and UAV behaviour that already exists. To this end, we introduce a novel formulation for the exploration of a disaster situation where we require that UAVs focus on areas of perceived danger, but also where these regions intersect with likely occupation by people. The rationale here, is that data about a region containing known hazards (for example, high levels of radiation or presence of fire) is only useful in preserving human life when it is known or believed that there are likely to be persons in the vicinity of a hazard; or will be at some future time. We give the example of radiation as a possible manifestation of danger in a disaster scenario. In principle, we can extend this to any phenomenon that is present over an extended area and represents some risk to human life. This general approach could then represent several types of hazard in a disaster area, such as flooding, chemical spills, or risk from earthquake-damaged buildings. At this point, we consider the location of such hazards as static. Such an assumption can be justified in scenarios with slow-changing conditions (relative to the time taken by the UAVs to explore).

We will now outline the formulations describing the state of the environment, and the actions available to the UAVs in our simulations.

3.1 Environment Model

In considering a model for the distribution in space of a number of civilian casualties, we exploit the ubiquity of mobile phone ownership and assume the use of mobile phone signals as proxies for the presence of a person. As well as having precedent in previous use in disaster scenarios [15, 26], this has the specific advantage of allowing identification of individual sources using unique identifiers associated with each handset. While a priori knowledge of the number of victims in a given area might be unavailable, first responders can maintain a belief distribution over unobserved victims while also attempting to isolate signals that have been observed, in order to reduce the uncertainty in the location of victims.

As a result, we explicitly envisage a scenario where UAVs are equipped with some form of detector capable of providing a (noisy) estimate of the range of individual unique phone signals. Specifically, we seek to localise the expected position of victims in order to reduce the time taken by search and rescue teams to find (and subsequently rescue) them. In more detail, we associate the uncertainty of a person's location with time taken to search the area for that person. By moving the detectors around the space, the expected location

value can be determined with higher precision; effectively reducing the area to be covered (and thus the time taken) by rescue services from a large initial area to a much smaller location.

We consider a search area containing a number of signals $s \in \mathcal{S}$ indicating the presence of people in some *danger* (mapped spatially by a two dimensional scalar function $\mathcal{D} : \mathbb{R}^2 \mapsto [0, 1]$), corresponding to their expected likelihood of dying within the next time-step t. The reward we gain R is related to the number of people we hope to observe, their likelihood of survival, and a discovery time t indicating how long it would take to rescue any victim:

$$R = \sum_{p_i, d_i \forall s_i \in \mathcal{S}} p_i \cdot (1 - d_i)^t$$

where $d_i \in \mathcal{D}$ and $p_i \in \mathcal{P}$ represents the expected number of people for a given signal $s_i \in \mathcal{S}$. Each signal s_i is mutually distinguishable from other signals, and the magnitude of each can be sensed by UAVs within a set radius. In the first instance we assume a relatively flat prior belief of victim position, implying a long time to locate an individual. However, with a set of observations (O) of—for example—the strength of a mobile phone signal some estimate can be made of the location of a person; effectively reducing the time to locate them. We denote this using a *time to find* parameter t_f that decreases linearly with the estimated area of the location of an individual phone signal.

As such, at any given time our reward is then:

$$\sum_{p_i, d_i \forall s_i \in \mathcal{S}} p_i \cdot (1 - d_i)^{t_f(O)}$$

whereas projecting reward to any arbitrary time in the future we have for a series of observations at time t:

$$R_{known} = \sum_{p_i, d_i \forall s_i \in \mathcal{S}} p_i \cdot (1 - d_i)^{t_f(O_t)+t}$$

By collecting information on signal sources we can use a population Monte-Carlo (PMC) [7] to model the likely locations of a person, which increases in precision as more measurements are taken, explicitly reducing t_f. Thus reward is fundamentally a function related to the danger at a given location believed to contain a victim, and the precision with which the location of that person is known. Planning can be performed by simulating the result of measurements on the probability distribution for each person and extrapolating the effect on t_f in each instance.

To account for signals we have yet to observe, we include a term for the *expected* number of victims outside of the range of observations. In principle the search area for such victims would be the entirety of the area over which observations have yet to be recorded, since the positions of the victims are known with no localisation whatsoever. Thus for a continuous distribution of expected people (p_e):

$$R_{unknown} = \int p_e(x, y) \cdot (1 - d(x, y))^{t_f} \, dxdy$$

The global reward function at time t then simply becomes the sum of the two components:

$$R_{total} = R_{known} + R_{unknown} \tag{1}$$

It is worth noting that since the formulation of reward from the population Monte-Carlo simulation is essentially separate from the tree-search function outlined below, the two are (in general cases) separately applicable.

3.2 UAV Behaviour Formulation

We consider simple UAV flight dynamics—including minimal constraints on performance—since the focus of our work is on planning rather than constraint optimisation, and because restrictions on UAV behaviour can be included in subsequent iterations of the model as constraints on the reward function. Thus the set of UAVs $U = \{u_1, \ldots, u_m\}$ traverse the space in iterations of a fixed distance δ per time-step t (i.e. at fixed speeds and altitudes); with a continuous domain of available angles available to determine the direction of the next action. The action vectors enabling UAV u_k to move at the next time step is defined as $a_k = (a_{k\alpha}, a_{k\beta}, a_{k\gamma}, \ldots)$ where each Greek index can be interpreted as an angle between 0^{o} and 360^{o}. In theory the cardinality of a_k is infinite, but as detailed below we use continuous space tree-search methods to restrict our search to finite subsets. Each UAV selects a sequence of actions to produce its trajectory $T_k = [a_k(t = 1), a_k(2), \ldots, a_k(t_{end})]$ (for a trajectory that ends at time t_{end}); which together form the set of all trajectories $\mathbf{T} = \{T_1, \ldots, T_m\}$. Thus the collective goal of the UAVs is to plan a set of trajectories to satisfy: $\mathbf{T}^* = \arg\max_R R(\mathbf{T})$.

4 The Coordinated Continuous Monte Carlo Tree-Search Algorithm

In choosing Monte-Carlo tree search as the basis for our solution, we note its ability to sample very quickly from large state spaces (traditionally used in solving games), and the flexibility with which it can be applied to general problems [6]. To do this we exploit locality between UAVs to factor the search space into local joint-action trees. Furthermore, we allow trees to coordinate over shared factors (that is, UAVs in multiple trees) using the max-sum algorithm [23] by exchanging messages to express the local reward gained by UAVs taking particular actions at future times. In other words, when selecting which node in a tree to expand the individual trees are coordinated over their shared UAVs to select mutually nonconflicting actions that are maximally beneficial to both trees.

At this stage, we design the algorithm to plan on-line and recalculate the optimum action at each time-step. In this way we have built-in robustness to temporal changes in the map (as well as not requiring a priori knowledge of future coordination requirements). We currently run simulations in a centralised fashion—insofar as they are performed on a single computer—but with allowances for multiple parallel threads representing the different individual calculations for each portion of the factored utility. In addition, we note that the nature of the max-sum coordination is such that UAVs are not required to have perfect information: it is sufficient that they know their local utility and are able to share this with local neighbours.

Specifically, we introduce an additional step to the standard MCTS process of tree growth. This growth is typically summarised: node *selection*, *expansion*, *rollout* or simulation, and *backpropagation* [6, 16]. Most significantly, we modify the selection process to determine which node to expand by coordinating in parallel between trees via max-sum. We detail our approach in the following subsections.

4.1 Tree Construction

At each timestep in the simulation, the coordinated MCTS (Co-MCTS) algorithm begins by calculating which UAVs require coordination with their neighbours, leading to the form of the UAV-based factor graph constructed in the joint-action creation function $\mathcal{J}$ (Line 3), detailed fully in Algorithm 1. This is performed to establish whether coordination is needed in a given UAV's locality: where

a UAV is spatially isolated from neighbouring UAVs, a local tree is grown. The resulting groups of UAVs will form the basis of the factor graph used in the max-sum calculation (Line 13 in Algorithm 1). The result of $\mathcal{J}$ is represented formally by a set $N = \{n_1, \ldots, n_f\}$ that represents the domain of the factor nodes to be coordinated. Specifically, each member of N contains a set of actions corresponding to a group of UAVs that require coordination. In this case, the corresponding element in N—say n_i—would be the set of actions available to these UAVs: $n_i = \{a_1, a_2, \ldots, a_k\}$. Trees are grown for each n_i in N, each of which in turn represents the factors in the max-sum graph connected to the variables representing the available actions of the UAVs. In more detail, trees are grown for each joint-utility between interacting UAVs, and coordination between trees is performed when trees share access to a given UAV. Individual nodes in the tree n_i will be indicated as $n_i^{(k)}$, or from any arbitrary tree by $n^{(k)}$.

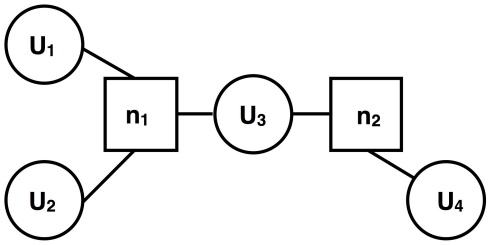

Figure 1. An example of four UAVs (u_{1-4}) interacting via a max-sum factor graph. Trees are grown for n_1 and n_2.

An example max-sum factor graph is represented in Figure 1. Here, four UAVs may interact at the next time-step in the following sub-sets: $\{u_1, u_2, u_3\}$ and $\{u_3, u_4\}$. As such, the algorithm maintains two joint trees for these sets, represented as the utility nodes n_1 and n_2, which must coordinate over the action of the UAV common to both: u_3. Framing this in terms of the action selected as a result of the tree-search, the coordination serves to ensure the two trees select an action for a_3 that is both mutually beneficial to both factored reward functions, and also the same; since a_3 can only take a single action at the next time step, the two trees must "agree" on what this action is.

4.2 Node Selection and Expansion

Algorithm 1 begins with the creation of the root nodes representative of each factor seen in Line 6, which are recorded in the set N_r (Line 4). Following this, the creation and growth of branches is performed Δ times inside the loop beginning at Line 8. This begins by exploring down each tree, starting from the root node, to determine which node to branch on next. Node selection is performed in accordance with standard progressive widening [11, 25], where we sample randomly from the action set of a node up to a limit of K actions per node, with K defined as in [10] to be a parameter of constants $C > 0$ and $\alpha \in (0, 1)$ and the time of simulation t: $K = Ct^{\alpha}$.

Line 9 introduces the current set of nodes (across all trees) to be expanded next, N_{next}, and Line 10 creates the set of previously expanded nodes N_{prev}. At Line 13 the max-sum algorithm is used to maximise the value of rewards (as per Equation 1) the actions over each n_i, returning a vector of favourable actions $a^* = (a_1^*, a_2^*, \ldots, a_m^* \mid a_k^* \in a_k)$. Since each n_i depends on a subset of actions, the function select$(n^{(k)}, a^*)$ serves to return only the actions corresponding to a given $n^{(k)}$. This is then used as the argument to create the new expansion to a node in N_{next} in Line 16.

Algorithm 1 Coordinated MCTS

```
CoMCTS (U, D, t = 0)
1.  for t in [1, ..., t_end]
2.    //Creation of factor graphs given UAV locations//
3.    N ← J (U)
4.    N_r ← ∅
5.    for n_i in N
6.      append(N_r) ← n_i^(0)
7.    endfor
8.    for eachstep in [1, ..., Δ]
9.      N_next ← N_r
10.     N_prev ← ∅
11.     while N_next ≠ ∅
12.       //Max-sum coordination returns best actions for shared factors//
13.       a* ← maxsum (N, N_next)
14.       for n^(k) in N_next
15.         //Actions relevant to n^(k) selected//
16.         n_new^(k) ← expand (n^(k), select (n^(k), a*))
17.         if expansions (n^(k)) ≥ K
18.           remove (n^(k), N_next)
19.           append (n^(k), N_prev)
20.         endif
21.       endfor
22.     endwhile
23.     for n^(k) in N_prev
24.       //Simulation (rollout) and backpropagation of results//
25.       rollout(n_new^(k))
26.       backpropagate(n_new^(k))
27.     endfor
28.   endfor
29.   for n_i in N
30.     a*_i = (bestactions (n_i))
31.   endfor
32.   t ← t + 1
33. endfor
```

Although not explicitly tested, we note that this approach ensures communications between UAVs need not be excessive. We note existing literature has shown at-length that max-sum in particular is robust to low bandwidth and irregular message-passing [12, 13, 23]. In practice we envisage that where UAVs share a tree a single UAV will handle the growth and planning of the joint actions.

4.3 Rollout and Backpropagation

The rollout portion of the MCTS (line 25 in Algorithm 1) is traditionally a coarse estimate of the affect of future actions as the result of exploring a particular node in the action space, although some more recent work has focussed on principled simulations using existing MCTS techniques [3]. In this example, we base the rollout on a random-walk through the action space starting at the node just expanded, biased in the direction of the last action taken. This method has the benefit of showing not just the contribution of any random series of actions, but of taking more actions similar to the one represented by the frontier node (for each UAV). Intuitively, a purely random rollout from one node in a joint action tree will be insignificantly different from a rollout from any similar node. Conversely, our rollout policy contributes to the exploration value of a node by indicating possible future reward through continued tree expansion with a preference for repetitions of the action itself.

Finally the rewards calculated at the leaf nodes are backpropagated up the tree towards the root by iteratively updating cumulative average rewards for each upstream node (Line 26). This is unchanged from classic MCTS.

5 Empirical Evaluation

To verify the performance of our algorithm on data relevant to real-world disaster scenarios, we used data from the Ushahidi project [21] produced from crowd-sourced information during the 2010 Haiti earthquake to generate $\mathcal{D}$ and $\mathcal{S}$.[6] This dataset was selected as it represents one of the largest available sources of information about spatial distributions of people and damage to buildings from any recent natural disaster. Furthermore, with increased interest in simple systems that allow crowds to provide data using their phones very quickly after a disaster takes place, the prevalence of such datasets will invariably increase in future; further underscoring the importance of testing our algorithm on this type of data. Specifically, we extracted the level of damage and coordinates of buildings in a 2km square centred on the capital, Port-au-Prince. Damage was rated based on crowd reports on a scale from 1 to 5, with 5 being most severe.

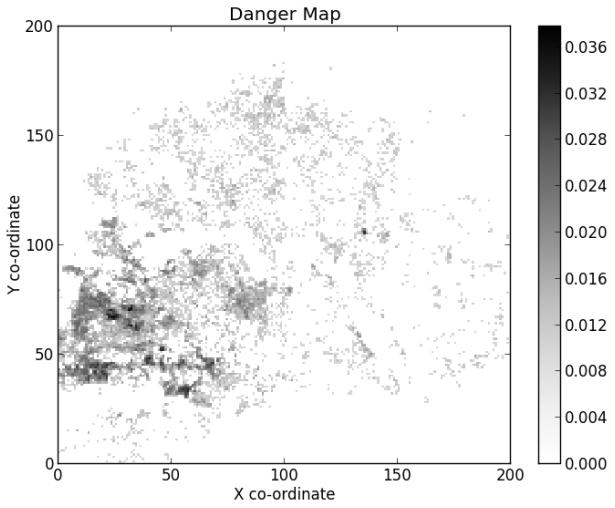

Figure 2. Danger $\mathcal{D}$ as a function of position, created from Ushahidi dataset centred over Port-au-Prince. Dimensions of $2km$ along each side.

We then constructed a decomposed grid world of size 200×200 of $10m \times 10m$ cells, to form the basis of the danger function $\mathcal{D}$. Since damaged buildings represent an estimate of the damage in an area and thus, the danger to the victims on the ground, we formed a belief map of danger to the populace by summing the total number of buildings above a threshold level (set to a crowd report of damage 3 and above) in each cell, before multiplying by a common factor to convert the data into a map representative of expected fatalities (noting the constraint imposed by the domain of $\mathcal{D}$). The environment is displayed in Figure 2 with a scale showing the value of d in each location. In calculating values for use in the reward functions, the value of danger is based on the mean expected position of the signal based on the data collected. Where the spatial location is not yet clearly established we have found that empirically the change in spatial danger was smooth enough that nearby values tended to be close to the final estimated value of d_i in most cases.

Assuming a UAV speed—typical of quad rotor vehicles—of $10ms^{-1}$ amounts to the traversal of one action of moving $\delta = 10m$ in one timestep of one second. We typically simulate UAV searches over time horizons of $t_{end} = 1000$.

The performance metric used is the percentage reduction in the time for total cumulative discovery time t_f (averaged over the number of UAVs) since it best reflects the ability of the UAV search to pinpoint victims for rescue. We benchmark against a similarly coordinated—but discrete—MCTS implementation, where the action space of the UAVs is restricted to moving between the cells forming the danger-function environment. This scenario poses similar challenges of coordination in large action spaces but benefits from existing work that deals with factored finite-space tree-search [2], and has already shown its efficacy in planning over the Ushahidi dataset [4].

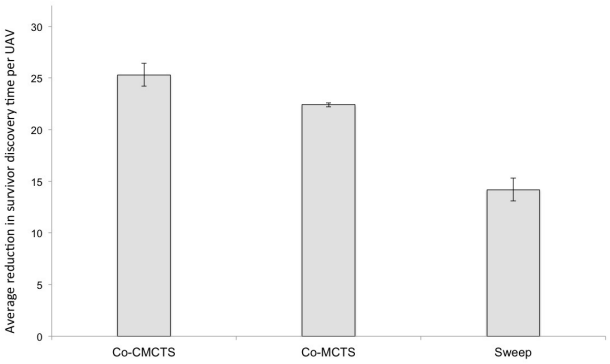

Figure 3. Result of randomised starting position tests for each of the continuous coordinated MCTS, discretised (cellular) coordinated MCTS and a simple lawnmower sweep-search; performed 10^6 times. Results indicate reduction in t_f averaged over the four UAVs in the scenario.

An initial simulation with four UAVs in randomised start locations on the map shown in Figure 2 is shown performing against a discretised coordinated MCTS implementation (as described in [4]) and a simple lawnmower-style sweep search over the area for comparison. This shows a gain of around ~7% over a discretised search space (Figure 3). We note that computation time of tree growth on the order of hundreds of nodes typically took less than half a second, demonstrating that computational complexity is not excessive.

Furthermore, we are able to demonstrate the consistency of our approach on addition of further UAVs to the simulation. Intuitively the reward gained by each UAV in a well-coordinated algorithm should suffer fewer diminishing returns when adding more to the scenario. In detail, this is because any additional UAVs should still localise close-to the same number of people as other UAVs in the environment, if they coordinate the exploration task effectively as a group. If they do not, one would expect additional UAVs would explore the same regions of the disaster space as those already present: which, as discussed previously, offers negligible improvements to the global reward function when compared to localising previously un-seen casualties. We demonstrate in Figure 4 that additional UAVs results in a slower decrease in observed reward than in the discretised action space. Most notably at 5 UAVs the difference in performance per-vehicle is approximately 18% in favour of the continuous algorithm. This benefit is due to the continuum of actions available being less restrictive than in a cellular decomposition of the search area; allowing more effective coordination.

[6] Available from `http://www.ushahidi.com`

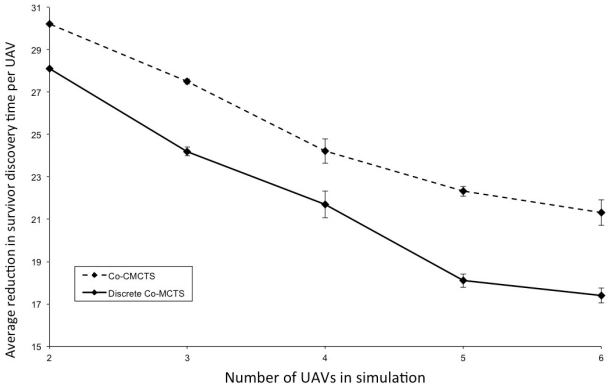

Figure 4. Comparison of continuous and discrete coordinated MCTS in a $t_{end} = 1000$ simulation of varying numbers of UAVs. The continuous space approach is not only better than the discretised approximation, it is more consistent in its reward per-UAV added to the scenario. Results here are averaged per-UAV in the simulation.

6 Conclusions

Motivated by increased availability of belief-data about disaster environments, we have introduced an implementation of a decentralised, factored, coordinated Monte Carlo tree search algorithm for the purpose of discovering survivors in a simulated UAV path planning scenario. Tests were carried out on real-world data from the 2010 Haiti earthquake via the Ushahidi platform; an environment with a continuous action space over a large area. We demonstrated the capability of our Co-CMCTS algorithm in sampling this space and planning paths, and demonstrated consistent performance gains over a discretised algorithm in the number of survivors discovered of up to 18%. Future work will seek to extend these solutions to different densities of survivors, time-varying belief maps, and—as part of ongoing collaborative efforts—will attempt field-trials of the algorithms proposed above on real-world platforms to further demonstrate the efficacy and real-world applicability of our contributions.

References

[1] S. M. Adams and C. J. Friedland, 'A Survey of Unmanned Aerial Vehicle (UAV) Usage for Imagery Collection in Disaster Research and Management', in *Proceedings of the Ninth International Workshop on Remote Sensing for Disaster Response*, volume 9, Stanford, MA, (2012).

[2] C. Amato and F. A. Oliehoek, 'Scalable Planning and Learning for Multiagent POMDPs', in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pp. 1995–2002, Austin, Texas, (2015). AAAI.

[3] H. Baier and M. H. M. Winands, 'Nested Monte-Carlo Tree Search for Online Planning in Large MDPs', in *European Conference on Artificial Intelligence (ECAI)*, volume 242, pp. 109–114. IOS, (2012).

[4] C. A. B. Baker, S. D. Ramchurn, W. L. Teacy, and N. R. Jennings, 'Planning Search and Rescue Missions for Unmanned Aerial Vehicle Teams', in *ICAPS Proceedings of the 4th Workshop on Distributed and Multi-Agent Planning (DMAP-2016)*, London, (2016). AAAI.

[5] S. Bernardini, M. Fox, and D. Long, 'Planning the Behaviour of Low-Cost Quadcopters for Surveillance Missions', in *Proc. of 24th Int. Conference on Automated Planning and Scheduling*, pp. 445–453, Portsmouth, NH, (2014). AAAI.

[6] C. B. Browne, E. J. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, 'A Survey of Monte Carlo Tree Search Methods', *Transactions on Computational Intelligence and AI in Games*, **4**(1), 1–43, (2012).

[7] O. Cappé, A. Guillin, J. M. Marin, and C. P. Robert, 'Population Monte Carlo', *Journal of Computational and Graphical Statistics*, **13**(4), 907–929, (2004).

[8] M. Cashmore, M. Fox, T. Larkworthy, D. Long, and D. Magazzeni, 'AUV Mission Control via Temporal Planning', in *2014 IEEE International Conference on Robotics and Automation*, pp. 6535–6541, Hong Kong, China, (2014). IEEE.

[9] Y.-b. Chen, G.-c. Luo, Y.-s. Mei, J.-q. Yu, and X.-l. Su, 'UAV path planning using artificial potential field method updated by optimal control theory', *International Journal of Systems Science*, (October), 1–14, (jun 2014).

[10] A. Couëtoux, J.-B. Hoock, N. Sokolovska, O. Teytaud, and N. Bonnard, 'Continuous Upper Confidence Trees', in *Proceedings of the 5th International Conference on Learning and Intelligent Optimization*, ed., C. A. Coello-Coello, number 5, pp. 433–445, Rome, Italy, (2011). Springer.

[11] R. Coulom, 'Efficient Selectivity and Backup Operators in Monte-Carlo Tree Search', in *5th International Conference on Computer and Games*, eds., P. Ciancarni and H. J. V. D. Herik, volume 4630, pp. 72–83, Turin, Italy, (2006).

[12] F. M. Delle Fave, A. Farinelli, A. Rogers, and N. R. Jennings, 'A Methodology for Deploying the Max-Sum Algorithm and a Case Study on Unmanned Aerial Vehicles', in *The Twenty-Fourth Innovative Applications of Artificial Intelligence Conference*, pp. 2275–2280, Toronto, (2012). AAAI.

[13] A. Farinelli, A. Rogers, and N. R. Jennings, 'Agent-based decentralised coordination for sensor networks using the max-sum algorithm', in *Autonomous Agents and Multi-Agent Systems*, volume 28, pp. 337–380, (2014).

[14] C. Goerzen, Z. Kong, and B. Mettler, 'A Survey of Motion Planning Algorithms from the Perspective of Autonomous UAV Guidance', *Journal of Intelligent and Robotic Systems*, **57**(1-4), 65–100, (nov 2009).

[15] A. Goetz, S. Zorn, R. Rose, G. Fischer, and R. Weigel, 'A Time Difference of Arrival System Architecture for GSM Mobile Phone Localization in Search and Rescue Scenarios', in *Positioning Navigation and Communication (WPNC), 2011 8th Workshop on.*, pp. 1–4. IEEE, (2011).

[16] L. Kocsis and C. Szepesvari, 'Bandit based Monte-Carlo Planning', *Machine Learning: ECML 2006*, **4212**, 282–293, (2006).

[17] A. Kolling and A. Kleiner, 'Multi-UAV Motion Planning for Guaranteed Search', in *Autonomous Agents and Multiagent Systems*, pp. 79–86, (2013).

[18] M. Kothari and I. Postlethwaite, 'A Probabilistically Robust Path Planning Algorithm for UAVs Using Rapidly-Exploring Random Trees', *Journal of Intelligent and Robotic Systems*, **71**(2), 231–253, (sep 2012).

[19] M. Kothari, I. Postlethwaite, and D.-W. Gu, 'Multi-UAV Path Planning in Obstacle Rich Environments Using Rapidly-exploring Random Trees', in *Proceedings of the 48th IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pp. 3069–3074, Shanghai, (dec 2009). IEEE.

[20] Y. C. Liu and Q. H. Dai, 'A Survey of Computer Vision Applied in Aerial Robotic Vehicles Yu-chi', in *OPEE 2010 - 2010 International Conference on Optics, Photonics and Energy Engineering*, number 201, pp. 277–280, Wuhan, China, (2010). IEEE.

[21] N. Morrow, N. Mock, A. Papendieck, and N. Kocmich, 'Independent Evaluation of the Ushahidi Haiti Project', Technical report, Ushahidi, (2011).

[22] R. R. Murphy, 'A Decade of Rescue Robots', in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5448–5449, Vilamoura, Portugal, (2012). IEEE.

[23] A. Rogers, A. Farinelli, R. Stranders, and N. R. Jennings, 'Bounded approximate decentralised coordination via the max-sum algorithm', *Artificial Intelligence*, **175**(2), 730–759, (2011).

[24] United Nations Foundation, 'Disaster relief 2.0', Technical report, United Nations, (2011).

[25] Y. Wang, J.-Y. Audibert, and R. Munos, 'Algorithms for Infinitely Many-Armed Bandits', in *Advances in Neural Information Processing Systems*, eds., D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, 1729–1736, NIPS, 21 edn., (2008).

[26] S. Zorn, R. Rose, A. Goetz, and R. Weigel, 'A Novel Technique for Mobile Phone Localization for Search and Rescue Applications', in *2010 International Conference on Indoor Positioning and Indoor Navigation*, number September, pp. 15–17, Zurich, Switzerland, (2010). IEEE.