Noname manuscript No.
(will be inserted by the editor)

Human-Agent Collaboration for Disaster Response

XX,YY,ZZ

Received: date / Accepted: date

Abstract In the aftermath of major disasters, first responders are typically overwhelmed with large numbers of, spatially distributed, search and rescue tasks, each with their own requirements. Moreover, responders have to operate in highly uncertain and dynamic environments where new tasks may appear and hazards may be spreading across the disaster space. Hence, rescue missions may need to be re-planned as new information comes in, tasks are completed, or new hazards are discovered. Hence, finding an optimal allocation of resources to complete all the tasks is a major computational challenge. Indeed, a number of agent-based solutions have been developed to help human commanders perform such planning tasks. However, studies have shown that, unless these solutions minimise the cognitive burden imposed on human commanders and first responders, they may hinder task performance rather than help. It is therefore crucial to deploy such agent-based planning solutions alongside human commanders in order to uncover the interactional issues that arise. Hence, in this paper, we use decision theoretic techniques to solve the task allocation problem posed by emergency response planning and then deploy our solution as part of an agent-based planning tool in real-world field trials. By so doing, we are able to study the interactional issues that arise when humans are guided by an agent. In more detail, we develop an algorithm, based on a Multi-Agent Markov Decision Process representation of the task allocation problem and show that it outperforms standard benchmarks from the literature. We then integrate the algorithm into a planning agent that responds to requests for tasks from participants in a mixed-reality location-based game, called AtomicOrchid, that simulates disaster response settings in the real-world. We then run a number of trials of our planning agent and compare it against a purely human driven system. Our analysis of these trials show that human commanders adapt to the planning agent by taking on a more supervisory role and that, by

XX
XX
E-mail: {XX}

YY
YY
E-mail: {YY}

ZZ
ZZ
E-mail: {ZZ}

providing humans with the flexibility of requesting plans from the agent, allows them to perform more tasks more efficiently than using purely human interactions to allocate tasks. We also discuss how such flexibility could lead to poor performance if left unchecked.

Keywords Human-Agent Interaction, Human-Agent Collectives, Disaster Response.

1 Introduction

In the aftermath of major disasters (man-made or natural), first responders (FRs), such as medics, security personnel and search and rescue teams, are rapidly dispatched to help save lives and infrastructure. In particular, FRs, with different skills, capabilities and experience may be required for the different tasks that need to be performed. For example, finding out where the civilians are trapped requires search and rescue teams and medics, transporting them to safe houses requires ambulances and security personnel and medics, while removing dangerous material from the environment requires safety experts and security personnel. While performing such tasks, FRs often operate in a very dynamic and uncertain environment, where, for example, fires spread, riots start, or the environment floods. Given this, FRs find it difficult to determine the best course of action and which task should be allocated to which team.

To assist in such situations, over the last few years, a number of algorithms and mechanisms have been developed to solve the coordination challenges faced by emergency responders (see Section 2 for more details). For example, [42] provide an algorithm to compute the optimal teams of emergency responders to allocate to tasks that require specific types of skills to complete, while [12,40] distribute such computations in an attempt to reduce the bandwidth required to coordinate. However, none of these approaches consider the inherent uncertainty in the environment or in the first responders' abilities. Crucially, to date, while all of these algorithms have been shown to perform well in simulations, none of them have been *exercised* to guide *real* human responders in real-time rescue missions. Crucially, studies on the deployment of such intelligent technologies in the real-world reveal that they typically impose a cognitive burden on the first responders [35,34] and disrupt task performance. Hence, it is important to develop real-world simulations of disaster response where such technologies can be trialled so that the interactional issues between humans and agents may be explored. Moreover, only through such trials will it become clear whether these algorithms will cope with real-world uncertainties (e.g., communication breakdowns or changes in weather conditions), be acceptable to humans (i.e., take into account their capabilities and preferences to perform certain tasks), and actually augment, rather than hinder, human performance (e.g., providing useful guidance and support rather than intrusive ones).

Against this background, we develop a novel algorithm for team coordination under uncertainty and evaluate it within a real-world mixed-reality game that embodies the simulation of team coordination in disaster response settings. Specifically, we consider a scenario involving rescue tasks (involving carrying a specific object to a safe zone) distributed in a physical space over which a (virtual) radioactive cloud is spreading. Tasks need to be completed by pairs of FRs with specific roles (e.g., medic, soldier, fire fighter, or transporter) that need to plan paths from their individual locations to meet at specific points in the disaster space to undertake these tasks. Moreover, they have to do so before the area is completely covered by the cloud (as FRs will die from radiation exposure) which is spreading according to varying wind speed and direction (that may result in highly uncertain radiation level predictions). Our algorithm captures the uncertainty in the scenario (i.e., in terms of environment and player states) and is able to compute a policy to allocate responders to tasks that minimises task completion time and plans routes for responders to ensure they not exposed to significant radiation. In order to be responsive, our algorithm is designed to return approximate solutions rather than optimal ones (that would take too much time to return solutions in a real-time setting). The algorithm is then used by a planning agent, working alongside a human commander, to guide FRs on the ground. Specifically, the planning agent is integrated into our test platform, AtomicOrchid,[1] that structures the interaction between FRs (i.e., players on the ground), a human commander, and the planning agent in a mixed-reality location-based game. In particular, the planning agent is designed to take over the burden of computing team plans from the human commander (who takes up a more supervisory role) while being responsive to requests to change plans from FRs (e.g., in case they are tired or prefer to do other tasks). By so doing, we are able to study, both quantitatively and qualitatively, the performance of a human-agent collective (i.e., a mixed-initiative team where control can shift between humans and agents) and the interactions between the different actors in the system [22]. In particular, we advance the state of the art in the following ways:

1. We develop a Multi-Agent Markov Decision Process (MMDP) to represent the problem of team coordination (i.e., path planning and task allocation) under uncertainty [7] and provide a novel algorithm to compute approximate solutions to the MMDP. We embed the mechanism to drive a planning agent in the AtomicOrchid game to evaluate it with users in a real-world setting.
2. We present a novel mixed-reality game, AtomicOrchid, to evaluate team coordination under uncertainty, focussing particularly on human-agent collaboration. In AtomicOrchid, human players in the field are supported by our planning agent in their mission to coordinate rescue tasks efficiently by communicating with headquarters and each other via a mobile phone application.
3. We run field trials of our planning agent in AtomicOrchid where instructs field responders through mobile messages in a disaster response scenario in multiple field trials. Our quantitative and qualitative analysis of the results show that providing flexible interactions between human participants and the planning agent improve task performance, particularly when the agent can rapidly respond to human requests for tasks.

When taken together, our results show, for the first time, how agent-based coordination algorithms for disaster response can be integrated and validated with human teams. Moreover, these results allow us to derive a methodology and guidelines for systems involving human-agent collaboration.

The rest of this paper is structured as follows. Section 3 formalises the disaster response problem as a MMDP. Section 4 describes the algorithm to solve the path planning and task allocation problems presented by the MMDP. Section 5 details the AtomicOrchid platform. Section 6 presents our pilot study and the field trial evaluation. Finally, Section 7 concludes.

2 Background and Related Work

In this section we discuss related work and provide a short background on the techniques used in this paper. As our work lies at the intersection between Multi-Agent Coordination and Human-Computer Interaction (HCI) for disaster management applications, we discuss relevant algorithms for multi-agent coordination to support emergency responders and then

[1] `http://bit.ly/1ebNYty`.

describe the HCI techniques and approaches for the same purpose. We then go on to discuss the challenges relevant to human-agent collaboration and then justify our use of mixed-reality games to evaluate this form of collaboration. Through our analysis we also identify the challenges that pertain to the coordination of human emergency responders under significant uncertainty and therefore, conclude with a survey on decision-theoretic approaches to solving the coordination problem under uncertainty in order to justify our agent-based solution for the team coordination problem.

2.1 Human Team Coordination for Disaster Response

Team coordination focuses on managing interdependencies between activities performed by individual team members to achieve the team's goal [31]. In emergency response situations, failures in team coordination can often occur due to the complexities of such interdependencies, and such failures are widely acknowledged as the most significant factor that can cost human lives [55, p. 2]. In particular, related work studies the challenges that arise when team members aim to create a shared understanding (e.g., of what needs to be done) [14], develop situation awareness (i.e., knowledge of the environment and the actors within it) [3], and align cooperative action through on-going communication [55].

Moreover, we note the work of [13] that highlighted that a key characteristic of large-scale disasters is the presence of multiple, spatially distributed incidents. To deal with multiple incidents, the disaster response team has to coordinate spatially distributed resources and personnel to carry out operations (e.g. search, rescue, and evacuation). Therefore, it is necessary to optimise the coordination of teams by allocating tasks to teams in time and space efficiently and sufficiently. Given this, a number of tools and system architectures have been developed to support such team coordination [33,38,14]. However, while these approaches focus on providing tools to human teams to better share information and formulate plans, they do not consider how such team coordination could be *optimised* using agent-based planning. Hence, in the next section, we survey a number of agent-based solutions to the task allocation problem in disaster response.

2.2 Agent-Based Planning for Disaster Response

Kitano et al [24] were the first to propose disaster response as a key application area for multi-agent systems. Since then, a number of algorithms and simulation platforms have been developed to solve the computational challenges involved. For example, algorithms have been developed to efficiently allocate emergency responders to rescue tasks (e.g., to rescue civilians, extinguish fires, or unblock roads) for (i) decentralised coordination: where emergency responders need to choose their actions based on local knowledge [12,40], (ii) coordination by a central authority: where a command centre is able to choose actions for all the members of the team given complete knowledge of the system [27,47,23], and (iii) coalition formation: where sub-teams can perform tasks with different levels of efficiency, as defined by the synergies between their capabilities (e.g., when two policemen help rescue a civilian from rubble they would be less effective than a fire and rescue officer and a medic) [42]. We adopt a centralised approach to the coordination problem and additionally consider the uncertainty in the environment (see more details in Section 2.5).

Now, many of these algorithms are actually evaluated in simulation using the RoboCupRescue disaster simulation platform [53]. In this platform, emergency response tasks are modelled computationally (e.g., functions describing speed and efficiency of agents at completing tasks) and the emergency responders are modelled as agents that automatically implement the outputs of a given task allocation algorithm [25,42]. While such evaluations are useful to determine extreme scenarios (e.g., best case when all agents implement all tasks perfectly or worst case when they do not), they are prone to misrepresentations of human decision making since they do not capture all the subtleties of human interactions and perception. In particular, they assume that all the actors in the system perfectly understand the messages exchanged, the information presented on their communication devices, and that they always follow the instructions received perfectly. In contrast, in a disaster response setting, responders may not always understand the plans computed by an agent nor obey instructions they receive from an agent (e.g., if they are part of different agencies, misunderstand the instructions, or are just tired).

2.3 Challenges for Human Agent Collaboration

Many multi-agent coordination algorithms have the potential to be applied to support task assignment of responder teams. However, before we use those algorithms to build agent planning support systems, there is a need to understand how human and agent can effectively collaborate. Human factors researchers have conducted controlled experiments to identify key aspects of human agent collaboration [10,37,54,56], propose transfer-of-control policies to shift control between humans and agents [49], and evaluate strategies of agent support for teams [30]. In particular, prior research has recognised that interaction design is vital for the performance of socio-technical human-agent systems [35], particularly where an agent directly instructs humans [34]. With inappropriate interaction design, agent-based planning support may function inefficiently, or at worst, hinder the performance of human teams. Although there is much literature in planning support, task assignment, and human-agent collaboration, to the exception of [50,48], no real world studies of how human emergency response teams actually handle agent support have been carried out. In fact, [50,48] mainly focus on humans acting as peers to agents in computational simulations rather than real-world deployments in the field.

Moverover, [9] have shown that an ill-designed work-flow management/automation system can lead to undesirable results, not only fail to improve work efficiency but also hinders human performance. Bowers et al. found that extreme difficulties might be encountered when introducing new technology support for human teams. It turns out that new technologies might not support, but disrupt smooth workflow if they are designed in an organisationally unacceptable way [1]. This further highlights the need to evaluate these technologies with human users before they are deployed in real-world settings. In particular, in this work, we turn to the use of gamification to run such evaluations.

2.4 Disaster Simulation and Games

Computational simulations, particularly agent-based simulations, are the predominant approach in the computing literature to predict the consequences of certain courses of action in disasters [20], to model information flow among first responders [46], to model the logistic distribution of emergency relief supplies [29]. However, as hinted above, these simulations are a poor substitute for real-world field trials. For example, Simonovic highlights that simulations may rely on unrealistic geographical topography, and most importantly, may not "account for human psychosocial characteristics and individual movement, and (...) learning ability" [52]. Moreover, the impact of emotional and physical responses likely in a disaster, such as stress, fear, exertion or panic [16] remains absent in most approaches that rely purely on computational simulation.

To combat this, in our study, we adopt a serious mixed-reality game approach to provide a setting in which people experience realistic cognitive and physical stress [17]. Mixed-reality games are recreational experiences that make use of pervasive technologies such as smart phones, wireless technologies and sensors with the aim of blending game events into a real world environment [5] . Arguably, they have become an established vehicle to explore socio-technical issues in complex real world settings [15]. The major advantage of mixed-reality games is the fact that they are situated in the real world, which leads to increased efficacy of the behavioural observations when compared to computational simulations.

By adopting the mixed reality games approach, for the first time, we are also able to run repeated trials interfaces for humans to use to control and receive instructions from an agent. In our particular context, the planning agent works alongside a human commander at central command (who can visualise its outputs and override them) in order to instruct human players on the ground.

2.5 Background on Decision-Theoretic Multi-Agent Planning

Decision theoretic planning is typically solved using Markov Decision Processes [32]. A *Markov decision process* (MDP) is a mathematical framework for sequential decision making under uncertainty where the problem is specified as a set of states and transition functions (specifying links between these states based on the actions taken). Then, the solution to an MDP specifies what action should be taken in each state given the possible transitions from a given state. In the presence of multiple agents, this model has been extended to *multi-agent MDP* (MMDP) [7] where the action chosen at any state consists of individual action components performed by the agents. Theoretically, any algorithm such as *linear programming*, *value iteration*, or *policy iteration* that can solve MDPs can also be used to solve MMDPs. However, these are likely to be very inefficient because the action space grows exponentially with the number of agents. In an attempt to combat this complexity, [8] show how domain structure can be exploited and thus introduced the *factored MDP* (FMDP) in which the state space is described by a set of variables and the transition model is defined by a *dynamic Bayesian network* (DBN). When the agents can only observe partial information about the state, this problem can be modelled by *multi-agent partially observable MDPs* (MPOMDP) [41]. Similar to MMDP, MPOMDP can be treated as an extension of single-agent POMDP to multi-agent domains. This analogy is useful because MPOMDPs can be solved as belief-state MMDPs where a belief state is a probability distribution over the states. All the above models assume that there is a centralised unit that will select a joint action for the team and distribute each action component to the corresponding agent. *Decentralised POMDP* (DEC-POMDP) [6] is a more general model where the agents are controlled in a decentralised manner. In other words, there is no centralised unit for distributing the actions and each agent must choose its own action based on the local observation.

In this paper, we restrict ourself to model our problem as the MMDP because other models do not fit the characteristics of our domain or are too difficult to be solved with the size of our problem. Specifically, in our domain, we consider a central controller (at headquarters) that will collect all the information and distribute the commands to each field responder. Therefore, it is not necessary to assume that the information is only partial (as in

MPOMDPs) or the decision must be made locally by the responders (as in DEC-POMDPs). Furthermore, those models are much harder than MMDPs and the existing algorithms can only solve very small problems. Moreover, we do not use the FMDP because most of the algorithms for solving this model require that the value function can be factored additively into a set of localized value functions [28, 18, 19] and our problem does not have such structures. For example, in our domain, several tasks may depend on the same responder. If she is delayed in one task, this may affect the completion of the other tasks. In other words, the completion of one task may depend on the completion of the other tasks, and so, the value function can not be factored on the basis of the local task states. Our settings are also different from the one in [12] where they assume that the responders are self-interested and need to negotiate with each other on the task that they want to perform next.

As discussed above, any algorithm that can solve large MDPs can be used to solve MMDPs. However, most of the existing approaches are offline algorithms (see the most recent survey [32] for more detail). The main disadvantage of offline algorithms is that they must compute a complete action mapping for all possible states in the policy. This is intractable for problems with huge state space as in our domain. In contrast to offline approaches, online algorithms interleave planning with execution and only need to compute the best action for the current state instead of the entire state space. Specifically, we adopt the basic framework of *Monte-Carlo tree search* (MCTS) [26], which is currently the leading online planning algorithm for large MDPs, and divide our online algorithm into two levels: task planning and path planning. It is worth pointing out that our method is different from the hierarchical planning for MMDPs [36] because it requires the task hierarchy to be part of the model and our problem does not have such task hierarchy for the responders. Indeed, our problem is more closely related to the *coalition formation with spatial and temporal constraints* (CFST) problem where agents form coalitions to complete tasks, each with different demands. However, existing work on CFST often assumes that there is no uncertainty on the agents' actions and the environment [42].

3 The Disaster Scenario

To develop a scenario for the evaluation of agent-based planning support, we held a number of consultations with emergency response organisations such as Rescue Global[2] and Hampshire County Council.[3] Specifically, these discussions informed the design of decision-making challenges (e.g., hazard avoidance, path planning, team construction) that mimic those that pervade real-world disaster response planning and execution while making reasonable assumptions about the environment and the participants.[4] In more detail, we consider a disaster scenario in which a satellite, powered by radioactive fuel, has crashed in a sub-urban area.[5] Debris is strewn around a large area, damaging buildings and causing accidents and injuring civilians. Moreover, radioactive particles discharged from the debris are gradually spreading over the area, threatening to contaminate food reserves and people. As the movement of this radioactive cloud is dependent on wind speed and direction,

[2] `http://www.rescuegloba.org.`

[3] `http://www3.hants.gov.uk/emergencyplanning.htm.`

[4] As access to emergency responders is either limited or costly for field trials, it was considered reasonable to hire volunteers that were taught to use the tools we gave them. The design of a fully-fledged training tool for disaster responder would be beyond the scope of this paper.

[5] Given the invisibility of radiation, it is possible to create a believable and challenging environment for the responders to solve in our mixed-reality game (see Section 5).

radiation levels may change drastically across the disaster space over time. Hence, emergency services (including medics, soldiers, transporters, and fire-fighters) are deployed to evacuate the casualties and key assets (e.g., food reserves, medication, fuel), each requiring different teams of FRs, before they are engulfed by the radioactive cloud. In what follows, we first model this scenario formally. Second, we describe the use of a planning agent at headquarters to help coordinate the team. Third, we formalise the optimisation problem the human-agent team faces (i.e., including fire-fighters, medics, and soldiers) in trying to save as many lives and assets as possible.

3.1 Formal Model

Let G denote a grid overlaid on top of the disaster space, and assume the satellite debris, casualties, assets, and actors are located at various coordinates $(x, y) \in G$ in this grid. The radioactive cloud induces a radioactivity level $l \in [0, 100]$ at every point it covers (100 corresponds to maximum radiation and 0 to no radiation). While the exact radiation levels can be measured by responders on the ground (at a given location) using their geiger counter, we assume that additional information is available from existing sensors in the area.[6] However, this information is uncertain due to the poor positioning of the sensors and the variations in wind speed and direction (we show how this uncertainty is captured in the next section). A number of safe zones $G' \subseteq G$ are defined where the responders can drop off assets and casualties (i.e., *targets* to be rescued). Let the set of FRs be denoted as $I = \{p_1, \cdots p_i \cdots, p_n\}$, where $|I| = n$ and the set of targets to be rescued (i.e., rescue tasks) be denoted as $T = \{t_1, \cdots, t_j, \cdots, t_m\}$, where $|T| = m$. A rescue task is performed by picking the target up, carrying it to a safe zone, and dropping it off. As FRs perform rescue tasks, they may become tired, get injured, or receive radiation doses that may, at worst, be life threatening. Hence, we assign each responder a health level $h_i \in [0, 100]$ that decreases based on its radiation dose (h_i is decreased by $0.02 \times l$ per second given a radiation level l) and assume that its decision to pick up and carry the target allocated to it is liable to some uncertainty (e.g., they may not want to pick a target because it is too far away or it is unclear how long it is going to take them to carry it back to a safe zone). Moreover, let Θ denote the types of responders (e.g., fire brigade, soldier, transporter, or medic) and assume a responder's type determines the capabilities she has and therefore the tasks she can perform. We denote as $\theta_i \in \Theta$ the type of responder p_i. In turn, to complete a given task t_j, a set of responders $C \subseteq I$ with specific types $\Theta_{t_j} \subseteq \Theta$ is required to pick up t_j. Thus, a task can only be completed by a team of responders C_j if $\{\theta_i | p_i \in C_j\} = \Theta_{t_j}$. Given the distribution of responders across the physical space, different sub-teams will perform to different levels (as they have to travel different distances) and this poses a challenge for the commander and to find the best teams needed to perform the tasks.

3.2 Human-Agent Collaboration

In line with practice in many countries, we assume that the FRs are coordinated from a headquarters (HQ) headed by a human coordinator H. In our case, H is assisted by an agent-based planning agent PA (more details in Section 4), that can receive input from,

[6] This assumption is not central to our problem and only serves to inform the decision making of the agent as we see later. It is also possible to obtain similar information about radiation levels by fusing the responders' geiger counter readings, but this is beyond the scope of the paper.

and direct, the FRs. Both H and PA can communicate their instructions (task plans to pick up targets) directly to the responders using an instant messaging system (or walkie talkie). While these instructions may be in natural language for H, PA instructs them with simple requests such as "Pick up target X at position Y with team-mates Z" messages. In turn, the responders may not want to do some tasks (e.g., if they are too tired, prefer to work with specific peers, or are prioritise some tasks over others) and may therefore simply accept or reject the received instruction from PA or H.[7] However, H can query the responders' decisions and request more information about their status (e.g., fatigue or health) and goals (e.g., meeting with team-mate at position X or going for task Y). Instead, if a task is rejected by the responders, PA records this as a constraint on its task allocation procedure (see details in Section 4.1.3) and returns a new plan. Thus on the one hand, richer interactions are possible between H and the first responders than between them and PA. On the other hand, PA runs an sophisticated task allocation algorithm that can compute an efficient allocation, possibly better than the one computable by H (particularly when many responders need to be managed). Note that, in contrast to previous work that suggested *transfer-of-control* regimes [49], our approach to handling requests from FRs to change the agent-based plan, does not constrain transfers of control to target specific decision points in the operation of the system. Rather, our interaction mechanisms are designed (see Section 5) to be more flexible to allow human control at any point (and our results in Section 6 validate this approach), along with human supervision (to implement possible corrective actions) by H.

4 Team Coordination Algorithm

Unfortunately, as in most MDP-based approaches to solving team coordination problems (see Section 2.5, our MMDP model results in a very large search space, even for small-sized problems. For example, with 8 responders and 17 tasks in a 50×55 grid, the number of possible states is more than 2×10^{400}. Therefore, it is practically impossible to compute the optimal solution. In such cases, we need to consider approximate solutions that result in high quality allocations. To this end, we develop an approximate solution using the observation that responders first need to *cooperatively* form teams (i.e., agree on who will do what), and that they can then *independently* compute the best path to the task. In our planning algorithm, we use this observation to decompose the decision-making process into a hierarchical structure with two levels: at the top level, a task planning algorithm is run for the whole team to assign the best task to each responder given the current state of the world; at the lower level, given a task, a path planning algorithm is run by each responder to find the best path to the task from her current location.

Furthermore, not all states of MMDPs are relevant to the problem (e.g., if a responder gets injured, she is incapable of doing any task in the future and therefore her state is irrelevant to other responders) and we only need to consider the reachable states given the current global state s of the problem. Hence, given the current state, we compute the policy online only for reachable states. This saves a considerable amount of computation because the size of the reachable states is usually much smaller than the overall state space. For example, given the current location of a responder, the one-step reachable locations are the 8 neighbouring locations plus the current locations, which are 9 locations out of the 50×55 grid. Jointly, the reduction is huge, from $(50 \times 55)^8$ to 9^8 for 8 responders. Another advantage

[7] While some agencies may be trained to obey orders (e.g., military or fire-fighting), others (e.g., transport providers or medics) are not always trained to do so [21].

Algorithm 1: Team Coordination Algorithm

Input: the MMDP model and the current state s.
Output: the best joint action $\mathbf{a}$.

```
// The task planning
```
$\{t^i\} \leftarrow$ compute the best task for each responder $p_i \in I$;
foreach $p_i \in I$ **do**
```
    // The path planning
```
    $a_i \leftarrow$ compute the best path to task t^i ;
return $\mathbf{a}$

of online planning is that it allows us to refine the model as more information is obtained or unexpected events happen. For example, given that the wind speed or direction may change, the uncertainty about the radioactive cloud may increase. If a responder becomes tired, the outcome of her actions may be liable to greater uncertainty.

The main process of our online hierarchical planning algorithm is outlined in Algorithm 1. The following sections describe the procedures of each level in more detail.

4.1 Task Planning

As described in Section 3.1, each responder p_i is of a specific type $\theta_i \in \Theta$ that determines which task she can perform and a task t can only be completed by a team of responders with the required types Θ_t. If, at some point in the execution of a plan, a responder p_i is incapable of performing a task (e.g., because she is tired or suffered a high radiation dose), she will be removed from the set of responders under consideration (that is $I \rightarrow I \setminus p_i$). This information can be obtained from the state $s \in S$. When a task is completed by a chosen team, the task is simply removed from the set (that is $T \rightarrow T \setminus t_k$ if t_k has been completed).

Now, to capture the efficiency of groupings of responders at performing tasks, we define the value of a team $v(C_{jk})$ that reflects the level of performance of team C_k in performing task t_j. This is computed from the estimated rewards the team obtains for performing t_j (as we show below). Then, the goal of the task planning algorithm is to assign a task to each team that maximises the overall team performance given the current state s, i.e., $\sum_{j=1}^{m} v(C_j)$ where C_j is a team for task t_j and $\{C_1, \cdots, C_m\}$ is a *partition* of I ($\forall j \neq j', C_j \bigcap C_{j'} = \emptyset$ and $\bigcup_{j=1}^{m} C_j = I$). In what follows, we first detail the procedure to compute the value of all teams that are valid in a given state and then proceed to detail the main algorithm to allocate tasks. Note that these algorithms take into account the uncertainties captured by the transition function of the MMDP.

4.1.1 Team Value Calculation

The computation of $v(C_{jk})$ for each team C_{jk} is challenging because not all tasks can be completed by one allocation (there are usually more targets than responders). Moreover, the policy after completing task t_j must also be computed by the agent, which is time-consuming given the number of states and joint actions. Given this, we propose to estimate $v(C_{jk})$ through several simulations. This is much cheaper computationally as it avoids computing the complete policy to come up with a good estimate of the team value, though we may not be able to evaluate all possible future outcomes. According to the law of large numbers, if the number of simulations is sufficiently large, the estimated value will converge to the true $v(C_{jk})$. This process is outlined in Algorithm 2.

Algorithm 2: Team Value Calculation

Input: the current state s, a set of unfinished tasks T, and a set of free responders I.
Output: a task assignment for all responders.

1 $\{C_{jk}\} \leftarrow$ compute all possible teams of I for T ;
2 **foreach** $C_{jk} \in \{C_{jk}\}$ **do**
  `// The` N `trial simulations`
3   **for** $i = 1$ **to** N **do**
4     $(r, s') \leftarrow$ simulate the process with the starting state s until task k is completed by the responders in C_{jk} ;
5   **if** s' *is a terminal state* **then**
6       $v_i(C_{jk}) \leftarrow r$;
7   **else**
8       $V(s') \leftarrow$ estimate the value of s' with MCTS ;
9       $v_i(C_{jk}) \leftarrow r + \gamma V(s')$;
10   $v(\bar{C}_{jk}) \leftarrow \frac{1}{N} \sum_{i=1}^{N} v_i(C_{jk})$;
11 **return** the task assignment computed by Equation 2

In each simulation of Algorithm 2, we first assign the responders in C_{jk} to task t_j and run the simulator starting from the current state s (Line 4). After task t_j is completed, the simulator returns the sum of the rewards r and the new state s' (Line 4). If all the responders in C_{jk} are incapable of doing other tasks (e.g., suffered radiation burns), the simulation is terminated (Line 5). Otherwise, we estimate the expected value of s' using Monte-Carlo Tree Search (MCTS) [26] (Line 8), which provides a good trade-off between exploitation and exploration of the policy space and has been shown to be efficient for large MDPs.[8] After N simulations, the average value is returned as an approximation of the team value (Line 10).

The basic idea of MCTS is to maintain a search tree where each node is associated with a state s and each branch is a task assignment for all responders. To implement MCTS, the main step is to compute an assignment for the free responders (a responder is free when she is capable of doing tasks but not assigned to any) at each node of the search tree. This can be computed by Equation 2 using the team values estimated by the UCB1 heuristic [2] to balance exploitation and exploration:

$$v(C_{jk}) = \overline{v(C_{jk})} + c\sqrt{\frac{2N(s)}{N(s, C_{jk})}} \tag{1}$$

where $\overline{v(C_{jk})}$ is the averaged value of team C_{jk} at state s so far, c is a trade-off constant, $N(s)$ is the visiting frequency of state s, and $N(s, C_{jk})$ is the frequency that team C_{jk} has been selected at state s. Intuitively, if a team C_{jk} has a higher average value in the trials so far or is rarely selected in the previous visits, it has higher chance of being selected in the next visit of the tree node.

[8] Other methods such as sequential greedy assignment or swap-based hill climbing [39] may also be useful. However, they do not explore the policy space as well as MCTS [26].

As we assume that the type of a responder and the role requirements of each task are static, we can compute all possible team values offline. Therefore, in the online phase, we only need to filter out the teams for completed tasks and those containing incapacitated responders to compute the team set $\{C_{jk}\}$.

4.1.2 Coordinated Task Allocation

Given the team values computed above, we then solve the following optimisation problem to find the best solution:

$$\begin{array}{ll} \max\limits_{x_{jk}} & \sum_{j,k} x_{jk} \cdot v(C_{jk}) \\ \text{s.t.} & x_{jk} \in \{0,1\} \\ & \forall j, \sum_{k} x_{jk} \leq 1 \qquad \text{(i)} \\ & \forall i, \sum_{j,k} \delta_i(C_{jk}) \leq 1 \ \text{(ii)} \end{array} \tag{2}$$

where x_{jk} is the boolean variable to indicate whether team C_{jk} is selected for task t_j or not, $v(C_{jk})$ is the value of team C_{jk}, and $\delta_i(C_{jk}) = 1$ if responder $p_i \in C_{jk}$ and 0 otherwise. In the optimisation, constraint (i) ensures that a task t_j is allocated to at most one team (a task does not need more than one group of responders) and constraint (ii) ensures that a responder p_i is assigned to only one task (a responder cannot do more than one task at the same time). This is a standard Mixed Integer Linear Program (MILP) that can be efficiently solved using off-the-shelf solvers (e.g., IBM CPLEX or lp_solve).

4.1.3 Adapting to Responder Requests

An important characteristic of our approach is that it can easily incorporate the preferences of the responders. For example, if a responder declines a task allocated to it by the planning agent, we simply filter out the teams for the task that contain this responder. By so doing, the responder will not be assigned to the task. Moreover, if a responder prefers to do the tasks with another responder, we can increase the weights of the teams that contain them in Equation 2 (by default, all teams have identical weights of 1.0). Thus, our approach is adaptive to the preferences of human responders.

4.2 Path Planning

In the path planning phase, we compute the best path for a responder to her assigned task. Our approach accommodates the uncertainties in the radioactive cloud and the responders' actions. We model this problem as a single-agent MDP that can be defined as a tuple, $\mathcal{M}_i = \langle S_i, A_i, P_i, R_i \rangle$, where: (1) $S_i = S_r^G \times S_{p_i}$ is the state space, (2) A_i is the set of p_i's actions, (3) $P_i = P_r \times P_{p_i}$ is the transition function, and (4) R_i is the reward function. In this level, responder p_i only needs to consider the states of the radioactive cloud S_r^G and her own states S_{p_i} and her moving actions. Similarly, the transition function only needs to consider the spreading of the radioactive cloud P_r and the changes of her locations and health levels when moving in the field P_{p_i}, and the reward function only needs to consider the cost of moving to a task and the penalty of receiving high radiation doses. This is a typical MDP that can be solved by many existing solvers (see the most recent survey [32]). We choose Real-Time Dynamic Programming (RTDP) [4] because it is simple and particularly fits our problem, that is, a goal-directed MDP with large number of states. However, other approaches for solving large MDPs could equally be used here.

There are several techniques we use to speed up the convergence of RTDP. In our problem, the map is static. Thus, we can initialize the value function $V(s)$ using the cost of the shortest path between the current location and the task location on the map, which can be computed offline without considering the radioactive cloud. This helps RTDP quickly navigate among the obstacles (e.g., buildings, water pools, blocked roads) without getting trapped in dead-ends during the search.

Since, in this paper, we focus on the integration and validation of the algorithm in a real-world deployment, we leave the presentation of computational simulation results and comparisons with other agent-based planning solutions (using our MMDP formulation) to Appendix C. As argued earlier (see Section 2.3), while computational simulations are useful to exemplify extreme cases, they do not explain how human FRs and the planning agent actually collaborate. We resolve this issue via a real-world trial of our algorithm as part of a planning agent next.

5 The AtomicOrchid Platform

In this section we describe the AtomicOrchid environment used to embed the planning agent in order to trial mixed-initiative coordination. In more detail, AtomicOrchid is a location-based mobile game based on the fictitious scenario described in Section 3. First responders are assigned a specific type: medic, fire-fighter, soldier, or transporter. Their mission is to evacuate all four types of targets: victim (requires medic and fire-fighter), animal (requires medic and transporter), fuel (requires soldier and fire-fighter), or other resource (requires soldier and transporter) before the disaster space is covered by a radioactive cloud (which we simulate using a diffusion process described in Section A). The first responders are supported by (at least) one person in a centrally located HQ room, and the planning agent PA that sends the next task (as described in the previous section) to the team of first responders. In what follows, we present the player interfaces used and the interactions with the planning agent in the game. A video of the operation of AtomicOrchid can be viewed at: `http://bit.ly/1ebNYty`.

5.1 Player Interfaces

First responders are equipped with a 'mobile responder tool' providing sensing and awareness capabilities in three tabs (geiger counter, map, messaging and tasks; see Figure 1). The first tab shows a reading of radioactivity, player health level (based on exposure), and a GPS-enabled map of the game area to locate fellow responders, the targets to be rescued and the drop off zones for the targets. The second tab provides a broadcast messaging interface to communicate with fellow first responders and the commander H. The third tab shows the team and task allocation dynamically provided by the agent PA that can be accepted or rejected. Notifications are used to alert both to new messages and task allocations.

H has at her disposal an 'HQ dashboard' that provides an overview of the game area, including real-time information of the players' locations (see Figure 2). The dashboard provides a broadcast messaging widget, and a player status widget so that the responders' exposure and health levels can be monitored. H can further monitor the current team and task allocations to individual responders by PA (by clicking on a button). The radioactive cloud (see Section A for more details) is graphically depicted as a heatmap ('Hotter' (red) zones correspond to higher levels of radiation). Crucially, only H can 'see' the entire cloud, while

Fig. 1 Mobile responder tool.

field responders are restricted to seeing a reading for their current location on their Geiger counters — this is a deliberate design choice to require frequent communication between HQ and field responders.

5.2 System Architecture

AtomicOrchid is based on the open-sourced geo-fencing game MapAttack[9] that has been iteratively developed for a responsive, (relatively) scalable experience. The location-based game is underpinned by client-server architecture, that relies on real-time data streaming between client and server. Client-side requests for less dynamic content use HTTP. Frequent events (e.g., location updates and radiation exposure) are streamed to clients to avoid the overhead of HTTP. In this way, first responders are kept informed in near real-time. Finally, to build the mobile app, we adapted the existing MapAttack Android app.

5.3 Integrating the Planning Agent

The planning agent PA takes the game status (i.e., positions of players, known status of the cloud, and messages received from players) as input and produces a plan for each responder for the current state. PA is deployed on a separate server. The AtomicOrchid server requests a plan from the agent via a stateless HTTP interface by transmitting the game status in JSON format. Polling (and thus re-planning) is triggered by two types of game events:

– *Completion of task*. On successful rescue of a target, a new plan (i.e., allocation of tasks to each responder) is requested from the agent.

[9] `http://mapattack.org`.

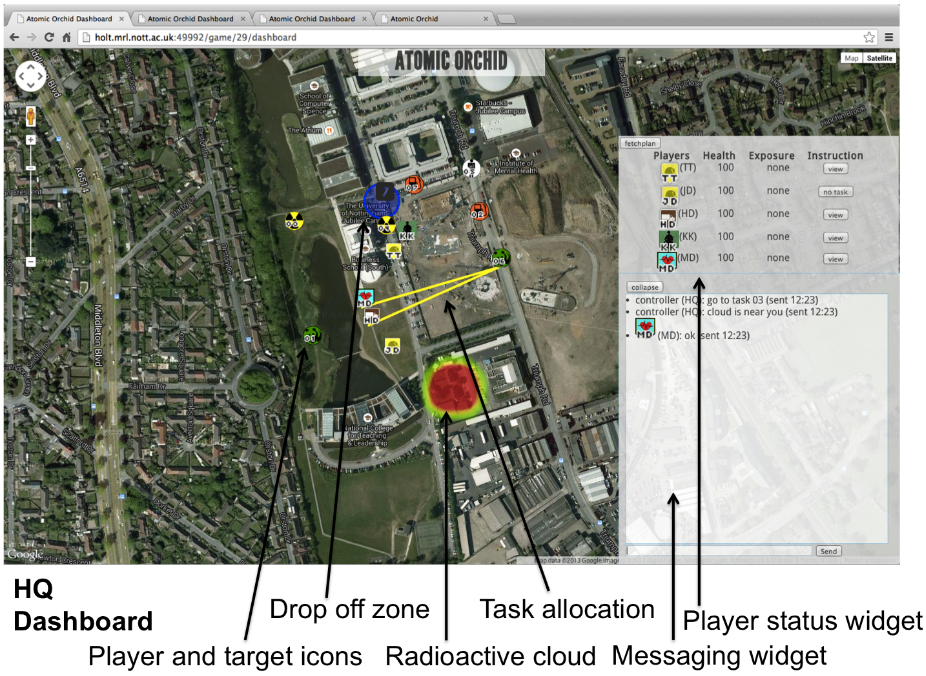


Fig. 2 HQ interface.

– *Explicit reject*. On rejection of a task allocation by any of the allocated first responders, a new plan is requested. More importantly, the rejected allocation is, in turn, used as a constraint within the optimisation run by the planner agent (as described in Section 4.1.3). For example, if two responders, one medic and one soldier, were allocated a task and the medic rejected it, the planning agent would rerun the optimisation algorithm with the constraint that this medic should not be allocated this task. If another medic is free (and accepts) the soldier and this medic can go ahead and complete the task. Otherwise, a new plan is created for the soldier. We provide an overview of the possible interactions between PA, H, and the FRs in Figure 3.

Once a plan is received from PA, the AtomicOrchid game engine splits the plan for a given team into individual task allocations for each player and sends them to their mobile responder app. The app presents the task allocation in the task tab, detailing: i) the responder to team up with, ii) the allocated target (using target id), and iii) approximate direction of the target (e.g., north, east). Once a player accepts a task, an acknowledgement is sent to their teammate, while rejecting a task triggers a new assignment from PA.

Now, in order to create a challenging, dynamic and pressured setting for the game players, H, and PA, we simulate a radioactive cloud that moves according to simulated wind conditions that mimics real-world cloud movement behaviours. The model we use for this purpose is described in Appendix 1.

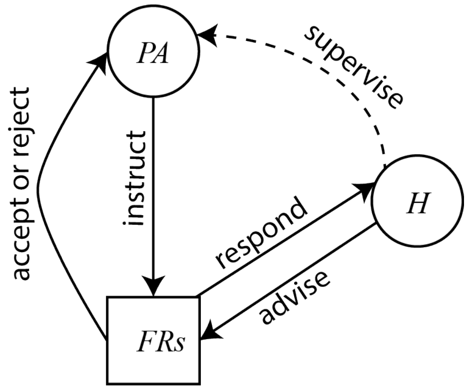


Fig. 3 Interactions between PA, FRs, and H.

6 Field Trialling AtomicOrchid

We ran three sessions of AtomicOrchid with participants recruited from the local university to trial mixed-initiative coordination in a disaster response scenario. The following sections describe the participants, procedure, session configuration and methods used to collect and analyse quantitative and qualitative data.

6.1 Participants and Procedure

Our approach to field trials is strongly influenced by [11, p. 1663] who state that "participants in field trials are used as experts on their own activity, attempting to predict what might happen with a particular technology, to develop insight based on their use". In particular, they point to the fact that running multiple trials (in an attempt to obtain statistically significant results) has no relationship to insightfulness. Crucially, they suggest that participants can be seen as investigators themselves in the trial. Hence, in line with this argument, we do not attempt to run tens or hundreds of trials to generate statistically significant results. Rather, we focus on running enough trials to gain an insight into how participants perceive and interact with our planning agent.

Hence, a total of 24 participants (17 males and 7 females) were recruited through posters and emails, and reimbursed with £15 for 1.5-2 hours of study. The majority were students. The procedure consisted of 30 minutes of game play, and about 1 hour in total of pre-game briefing, consent forms, a short training session, and a post-game group discussion.

At the end of the briefing, in which mission objectives and rules were outlined, responder types were randomly assigned to all participants (fire-fighter, medic, transporter, soldier). The HQ was staffed by a different member of the research team in each session in order to mimic an experienced H, while avoiding the same person running the HQ every time. Moreover, responders were provided with a smartphone and H with a laptop. The team was given 5 minutes to discuss a common game strategy.

First responders were then accompanied to the starting point within the designated game area, about 1 minute walk from headquarters. Once first responders were ready to start, H sent a 'game start' message. After 30 minutes of game play the first responders returned to the HQ where a group interview was conducted, before participants were debriefed and dismissed.

6.2 Game Sessions

We ran one session without PA, and two sessions with PA to be able to compare team performance in the two versions. Each session involved *different* sets of first responders (8 each). Thus, players were unique to a session to avoid learning effects between sessions. While we cannot rule out learning effects *within* sessions, we accept this as an integral part of working with humans in the real world.

We also ran a pilot study for each condition to fine tune game configuration. The 8 first responders in each session were randomly allocated a role so that the whole team had two of each of the four types of responders. The terrain of the 400x400 metre game area includes grassland, a lake, buildings, roads, footpaths and lawns. There were two drop-off zones and 16 targets in each session. There were four targets for each of the four target types. The target locations, pattern of cloud movement and expansion were kept constant for all game sessions. The pilot study showed that this was a challenging, yet not too overwhelming configuration of game area size, and number of targets to collect in a 30 min game session.

6.3 Data Collection and Analysis

We developed a log file replay tool to triangulate video recordings of game action with the timestamped system logs that contain a complete record of the game play, including responders' GPS location, their health status and radioactive exposure, messages, cloud location, locations of target objects and task status.

To assess how humans interact with each other and with PA, we focused on collecting data relevant to PA's task allocations and remote messages that are used to support coordination. In particular, we use speech-act theory [51] to classify messages sent between and among responders and H. We focus on the most relevant types of acts in this paper (which are also the most frequently used in AtomicOrchid):

- Assertives: *speech acts that commit a speaker to the truth of the expressed proposition*; these were a common category as they include messages that contain situational information (e.g., You are next to the radiation cloud or the task is North of your current position).
- Directives: *speech acts that are meant to cause the hearer to take a particular action*; requests, commands and advice, including task and team allocation messages (e.g., X go to task 1, Y go with X).

6.4 Results

Overall, 8 targets were rescued in the non-agent condition (Session A), and respectively 12 targets (Session B) and 11 targets (Session C) were rescued in the agent condition. Teams (re-)formed six times in session A, four times in session B and nine times in session C. Average player health after the game was much higher (more than double) for the agent-assisted sessions (80 for Session B and 82 for Session C) compared to the non-agent assisted session (40 in Session A) (see table 1). In fact, one responder 'died' in Session A. This suggests that the agent's more conservative approach not to send field responders into 'harms way' paid off. Moreover, surprisingly this more conservative approach did also not result in less numbers of targets saved than with a perhaps less risk averse approach by HQ in Session A. This may probably be explained by more timely instructions by the agent, as these were generated automatically after a target had been dropped off.

PA dynamically re-planned 14 times in session B and 18 times in session C. In most cases, this was triggered when a target was dropped off in the safe zone (24 times) – as this frees up resources for PA to recompute an allocation. In the remaining cases, this was triggered by a player declining the agent's task allocation (8 times).

Table 1 Health of Field Responders

	Minimum health	Maximun health	Average health	Standard deviation
Session A	0	95	40	26.95
Session B	64	100	91	13.41
Session C	41	99	72	24.99

In particular, Figure 4 shows how first responders handled task allocations in the agent and non-agent conditions. In the non-agent condition, the HQ commander sent 43 task allocation directives (see Table 2 for details of each run) . Of these, the recipient first responders addressed only 15 messages (bringing them up in conversation). Of these 15, responders chose to ignore the instructions only once. The responders ignored the instruction because they were engaged in another task and did not want to abandon it. A further 4 H instructions were consistent with a decision to rescue a certain target that had already been agreed locally by the responders. In the remaining 10 cases, first responders chose to follow the instructions. Although players were willing to follow H's instructions, they failed to correctly follow the instructions due to confusion and misunderstanding in the communication. In fact, only 2 instances of directives from H led to task completion. The first responders performed 6 rescue operations (tasks) without being instructed by H.

Table 2 Message classification. FR: First Responder.

	no agent		agent				Total
Speech acts	Session A		Session B		Session C		
	HQ	FR	HQ	FR	HQ	FR	
Directives	89	0	34	2	34	0	159
Assertives	33	6	26	16	24	16	121
Total	122	6	60	18	58	16	280

In contrast, when task allocation was handled by the agent (52 tasks allocated in two trials on average), responders explicitly accepted 24 tasks, of which they completed 15 successfully. Although there was either no response or no consensus between the responders (in 17 tasks allocated), they still completed 6 of these tasks successfully. In total, 20 task allocations were withdrawn by the agent as a result of re-planning.

In terms of task rejections, first responders rejected PA's task allocation 11 times in the agent version. All of the rejections happened when the task allocation would have *split existing teams*, or instructed responders to team up with *physically more distant responders*. In most cases (9 out of 11), they triggered re-planning by rejection and *adjusted the task*

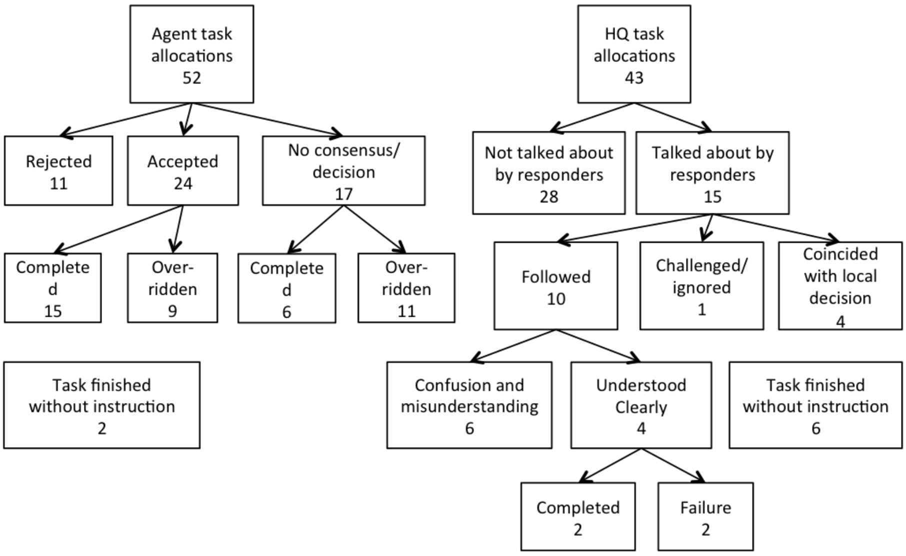


Fig. 4 How task allocations were handled by first responders in the version with agent (left), and without agent (right).

allocation to become consistent with the responder's current team. In the other two cases, the responders rejected the task allocation one more time before receiving the desired task allocation. For accepted instructions, the average distance between suggested teammates was 12 metres. For rejected instructions, the average was 86 metres.

The results above show that the simple mechanism to get PA to re-plan (i.e., reject/accept) was more successful (more tasks completed and less confusion) than the open-ended interactions between H and the responders (that were open to confusion). Moreover, the fact that many of the rejections were due to the long distance to travel and teammate preference, implies that players chose to do the tasks they *preferred* rather than those deemed optimal by the agent. This indicates there may be an issue of trust in the agent, but also that it may be easier for a responder to impose (through a reject) such preferences on an agent (and indirectly to other team members) rather than expressing this to H or directly to teammates. It is also important to note that in the agent-assisted setting, H frequently *monitored* the allocation of tasks returned by the agent (57 clicks on 'show task' in UI responder status widget). Whereas 43 directives out of 68 in the non-agent session were task allocations, only 16 out of 68 were directly related to task allocations in the agent version. Out of these, H directly reinforced the agent's instruction 6 times (e.g., "SS and LT retrieve 09"), and complemented (i.e., added to or elaborated) PA's task allocation 5 times (e.g., "DP and SS, as soon as you can head to 20 before the radiation cloud gets there first"). H did 'override' PA's instruction in 5 cases.

In the agent version, most of H's directives (52 out of 68) and assertives (49 out of 51) focussed on providing situational awareness and routing the responders to avoid exposing them to radiation. For example, "NK and JL approach drop off 6 by navigating via 10 and 09.", or "Radiation cloud is at the east of the National College".

In summary, these results suggest three key observations with regard to human-agent coordination in the trial:

1. First responders performed better (rescued more targets) and maintained higher health levels when supported by the agent. These results echo those obtained under simulation (see Section 4) and may reflect the better forward-planning capability of the planning agent compared to human responders.
2. Rejecting tasks was relatively frequently employed to trigger re-planning to obtain new task allocations aligned with responder preferences. In each case, the planning agent was able to adapt to provide an alternative that was acceptable to the responders. Without this facility we believe the responders would have chosen to ignore the plan. Task rejection seemed to be linked to changes to established teams, especially when members were relatively distant. Consequently, these kinds of allocations may need particularly support (e.g., explanation) or might be less preferentially selected by PA.
3. When task allocation was handled by PA, H could focus on providing vital situational awareness to safely route first responders around danger zones: thus demonstrating effective division of labour and complementary collaboration between humans and agents.

Given the above observation we argue that a planning agent for team formation should not only model the uncertainty in player behaviours and in the environment, but that interactional challenges also need to be addressed if such a technology is to be accepted in practice. In the next section we elaborate on how our study results lead to new design guidelines for Human-Agent Collectives.

7 Conclusions

In this paper we developed a novel approach for integrating and evaluating agent-based coordination algorithms that allocate teams of emergency responders in dynamic and uncertain environments. In particular, we developed a planning agent (using an MMDP approach), and conducted field-trials of a task planning agent using a mixed-reality game called AtomicOrchid in order to focus on the issues that arise in human-agent collaboration in team coordination.

Results from our study indicate the planning agent instructed players to carry out successful plans (outperforming a no-agent setting in terms of tasks completed and responders unharmed). The agent's ability to re-plan as per responders' preferences and constraints was particularly effective. In particular, based on our analysis, we propose the following design guidelines for human-agent collaboration in Human-Agent Collectives:

Adaptivity: our experiences suggest that planning algorithms should be designed to take in human input, and more importantly, be *responsive* to the needs of the users. As we saw in AtomicOrchid, players repeatedly requested new tasks and this would not have been possible unless our algorithm was computationally efficient but could dynamically assimilate updates, requests, and constraints. We believe this makes the algorithm more acceptable to the users. However, this adaptivity does cost the system in terms of efficiency as the rejection of tasks may lead the problem to be so constrained that the algorithm cannot return any solutions. To alleviate such issues, we believe human mediation may be important in nudging the players to justify their rejection of tasks or to nudge them not to do so too frequently.

Interaction Simplicity: our agent was designed to issue simple commands (Do X with Y) and respond to simple requests (OK or Reject Task). Such simple messages were shown to be far more effective at guiding players to do the right task than the unstructured human communication in the non-agent assisted case that was fraught with inconsistencies and inaccuracies. In fact, we would suggest that agents should be designed with minimal options to simplify the reasoning users have to do to interact with the agent, particularly when they are under pressure to act. However, interaction simplicity in this context, to us also means providing human responders with interactive abilities to do what they are good at: dealing with unforeseen contingencies. Hence, it is important to provide unconstrained communication means such as chat, walkie talkies or mobile phones in addition to the 'simple' instructions that the agent provides. In effect, we are designing an interactional setting in which the agent is dealing with the routine and predictable aspects of the setting (repetitive tasks assignments), and the human coordinators in the HQ are freed up to deal with contingencies and the less predictable complexities as and when they arise.

Flexbile autonomy: the HQ dashboard proved to be a key tool for the HQ coordinator H to *check* and *correct for* the allocations of PA, taking into account the real-world constraints that the players on the ground faced. In particular, letting the human oversee the agent (i.e., "on-the-loop") at times and actively instructing the players (and bypassing the agent) at other times (i.e., "in-the-loop") as and when needed, was seen to be particularly effective. This was achieved by H *without the agent* defining when such transfers of control should happen (as in [49]) and, therefore, left the coordinator the option of taking control when she judged it was needed. However, while this allows humans to *choose* what to do, it is not clear whether they would have been better off going with the agent's plan. Hence, we suggest that such deployed autonomous systems should be built for flexible autonomy. Specifically, interfaces should be designed to pass control *seamlessly* between humans and agents and the implications of human-based "corrective" actions should be made explicit to the humans to ensure they know when to take control, and when to let the agent decide.

Much remains to be done to further validate agent-based planning in real-world disaster response given that field trials of AtomicOrchid are limited to using volunteers and in settings that only approximate the typical environments faced by emergency responders. Hence, in future work, we aim to deploy our planning agent to support expert emergency responders from RescueGlobal during their annual multi-national disaster response training exercise (Angel Thunder[10]). By so doing, we will develop new requirements for agent-based technologies for settings where users are highly trained and the actions taken by the agent can have major impact on the performance of the team (e.g., leading to loss of lives or waste of real resources).

References

1. K. R. Abbott and S. K. Sarin. Experiences with workflow management: issues for the next generation. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work (CSCW)*, pages 113–120, 1994.
2. P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multi-armed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.
3. T. Bader, A. Meissner, and R. Tscherney. Digital map table with fovea-tablett®: Smart furniture for emergency operation centers. In *proceedings of the 5th International Conference on Information Systems for Crisis Response and Management*, pages 679–688, 2008.
4. A. G. Barto, S. J. Bradtke, and S. P. Singh. Learning to act using real-time dynamic programming. *Artificial Intelligence*, 72(1):81–138, 1995.
5. S. Benford, C. Magerkurth, and P. Ljungstrand. Bridging the physical and digital in pervasive gaming. *Communications of the ACM*, 48(3):54, Mar. 2005.

[10] http://www.dm.af.mil/library/angelthunder2013.asp.

6. D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein. The complexity of decentralized control of markov decision processes. *Mathematics of operations research*, 27(4):819–840, 2002.
7. C. Boutilier. Planning, learning and coordination in multi-agent decision processes. In *Proc. of TARK 1996*, pages 195–210, 1996.
8. C. Boutilier, R. Dearden, and M. Goldszmidt. Stochastic dynamic programming with factored representations. *Artificial Intelligence*, 121(1):49–107, 2000.
9. J. Bowers, G. Button, and W. Sharrock. Workflow From Within and Without : Technology and Cooperative Work on the Print Industry Shopfloor Introduction : Workflow Systems and Work Practice. In *Fourth European Conference on Computer-Supported Cooperative Work*, pages 51–66, 1994.
10. J. M. Bradshaw, P. Feltovich, and M. Johnson. Human-Agent Interaction. In G. Boy, editor, *Handbook of HumanMachine Interaction*, chapter 13, pages 293–302. Ashgate, 2011.
11. B. Brown, S. Reeves, and S. Sherwood. Into the wild: Challenges and opportunities for field trial methods. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 1657–1666, New York, NY, USA, 2011. ACM.
12. A. Chapman, R. A. Micillo, R. Kota, and N. R. Jennings. Decentralised dynamic task allocation: A practical game-theoretic approach. In *Proc. of AAMAS 2009*, pages 915–922, May 2009.
13. R. Chen, R. Sharman, H. R. Rao, and S. J. Upadhyaya. Design Principles of Coordinated Multi-incident Emergency Response Systems. *Simulation*, 3495:177–202, 2005.
14. G. Convertino, H. M. Mentis, A. Slavkovic, M. B. Rosson, and J. M. Carroll. Supporting common ground and awareness in emergency management planning. *ACM Transactions on Computer-Human Interaction*, 18(4):1–34, Dec. 2011.
15. A. Crabtree, S. Benford, C. Greenhalgh, P. Tennent, M. Chalmers, and B. Brown. Supporting ethnographic studies of ubiquitous computing in the wild. In *Proceedings of the 6th ACM conference on Designing Interactive systems - DIS '06*, page 60, New York, New York, USA, 2006. ACM Press.
16. J. Drury, C. Cocking, and S. Reicher. Everyone for themselves? A comparative study of crowd solidarity among emergency survivors. *The British journal of social psychology / the British Psychological Society*, 48(Pt 3):487–506, Sept. 2009.
17. J. E. Fischer, W. Jiang, A. Kerne, C. Greenhalgh, S. D. Ramchurn, S. Reece, N. Pantidi, and T. Rodden. Supporting team coordination on the ground: Requirements from a mixed reality game. In *COOP 2014-Proceedings of the 11th International Conference on the Design of Cooperative Systems, 27-30 May 2014, Nice (France)*, pages 49–67. Springer, 2014.
18. C. Guestrin, D. Koller, and R. Parr. Multiagent planning with factored mdps. In *NIPS*, volume 1, pages 1523–1530, 2001.
19. C. Guestrin, D. Koller, R. Parr, and S. Venkataraman. Efficient solution algorithms for factored mdps. *J. Artif. Intell. Res.(JAIR)*, 19:399–468, 2003.
20. G. I. Hawe, G. Coates, D. T. Wilson, and R. S. Crouch. Agent-based simulation for large-scale emergency response. *ACM Computing Surveys*, 45(1):1–51, Nov. 2012.
21. H. H. Initiative et al. Disaster relief 2.0: The future of information sharing in humanitarian emergencies. In *Disaster Relief 2.0: The future of information sharing in humanitarian emergencies*. HHI; United Nations Foundation; OCHA; The Vodafone Foundation, 2010.
22. N. R. Jennings, L. Moreau, D. Nicholson, S. D. Ramchurn, S. J. Roberts, T. Rodden, and A. Rogers. On human-agent collectives. *Communications of the ACM (In Press)*, 2014.
23. M. A. Khan, D. Turgut, and L. Bölöni. Optimizing coalition formation for tasks with dynamically evolving rewards and nondeterministic action effects. *Journal of Autonomous Agents and Multi-Agent Systems*, 22(3):415–438, 2011.
24. H. Kitano and S. Tadokoro. Robocup rescue: A grand challenge for multiagent and intelligent systems. *AI Magazine*, 22(1):39–52, 2001.
25. A. Kleiner, A. Farinelli, S. Ramchurn, B. Shi, F. Mafioletti, and R. Refatto. Rmasbench: a benchmarking system for multi-agent coordination in urban search and rescue. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2013)*, 2013.
26. L. Kocsis and C. Szepesvári. Bandit based monte-carlo planning. In *Proc. of ECML 2006*, pages 282–293, 2006.
27. M. Koes, I. Nourbakhsh, and K. Sycara. Constraint optimization coordination architecture for search and rescue robotics. In *Proceedings of IEEE Intl. Conf. on Robotics and Automation*, pages 3977–3982. IEEE, 2006.
28. D. Koller and R. Parr. Policy iteration for factored mdps. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pages 326–334. Morgan Kaufmann Publishers Inc., 2000.
29. Y. M. Lee, S. Ghosh, and M. Ettl. Simulating distribution of emergency relief supplies for disaster response operations. In *Proceedings of the 2009 Winter Simulation Conference (WSC)*, pages 2797–2808. IEEE, Dec. 2009.

30. T. L. Lenox, T. Payne, S. Hahn, M. Lewis, and K. Sycara. Agent-Based Aiding for Individual and Team Planning Tasks. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 44(1):65–68, July 2000.
31. T. W. Malone and K. Crowston. What is coordination theory and how can it help design cooperative work systems? In *Proceedings of the 1990 ACM conference on Computer-supported cooperative work - CSCW '90*, pages 357–370, New York, New York, USA, 1990. ACM Press.
32. Mausam and A. Kolobov. Planning with markov decision processes: An AI perspective. *Synthesis Lectures on AI and Machine Learning*, 6(1):1–210, 2012.
33. A. Monares, S. F. Ochoa, J. A. Pino, V. Herskovic, J. Rodriguez-Covili, and A. Neyem. Mobile computing in urban emergency situations: Improving the support to firefighters in the field. *Expert Systems with Applications*, 38(2):1255–1267, 2011.
34. S. Moran, N. Pantidi, K. Bachour, J. E. Fischer, M. Flintham, T. Rodden, S. Evans, and S. Johnson. Team reactions to voiced agent instructions in a pervasive game. *Proceedings of the 2013 international conference on Intelligent user interfaces - IUI '13*, page 371, 2013.
35. S. Murthy, R. Akkiraju, J. Rachlin, and F. Wu. Agent-based cooperative scheduling. In *Proceedings of AAAI Workshop on Constraints and Agents*, pages 112–117, 1997.
36. D. J. Musliner, E. H. Durfee, J. Wu, D. A. Dolgov, R. P. Goldman, and M. S. Boddy. Coordinated plan management using multiagent mdps. In *AAAI Spring Symposium: Distributed Plan and Schedule Management*, pages 73–80, 2006.
37. G. J. Nancy, Cooke and B. B. Pederson Harry. Distributed mission environments: Effects of geographic distribution on team cognition, process, and performance. In *Towards a science of distributed learning and training.* American Psychological Association, 2006.
38. R. P. Padilha, J. O. Gomes, and J. H. Canós. The Design of Collaboration Support Between Command and Operation Teams during Emergency Response. *Current*, pages 759–763, 2010.
39. S. Proper and P. Tadepalli. Solving multi-agent assignment Markov decision processes. In *Proc. of AAMAS 2009*, pages 681–688, 2009.
40. M. Pujol-Gonzalez, J. Cerquides, A. Farinelli, P. Meseguer, and J. A. Rodríguez-Aguilar. Binary max-sum for multi-team task allocation in robocup rescue. In *Optimisation in Multi-Agent Systems and Distributed Constraint Reasoning (OptMAS-DCR)*, Paris, France, 05/05/2014 2014.
41. D. V. Pynadath and M. Tambe. The communicative multiagent team decision problem: Analyzing teamwork theories and models. *Journal of Artificial Intelligence Research*, 16:389–423, 2002.
42. S. D. Ramchurn, A. Farinelli, K. S. Macarthur, and N. R. Jennings. Decentralized coordination in robocup rescue. *The Computer Journal*, 53(9):1447–1461, 2010.
43. C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.
44. S. Reece, S. Ghosh, S. Roberts, A. Rogers, and N. R. Jennings. Efficient state-space inference of periodic latent force models. *Journal of Machine Learning Research*, 15:2337–2397, 2014.
45. S. Reece and S. Roberts. An introduction to gaussian processes for the kalman filter expert. In *Prof. of Intl. Conf. on Information Fusion (FUSION)*, pages 1–9. IEEE, 2010.
46. C. Robinson and D. Brown. First Responder Information Flow Simulation: A Tool for Technology Assessment. In *Proceedings of the Winter Simulation Conference, 2005.*, pages 919–925. IEEE, 2005.
47. P. Scerri, A. Farinelli, S. Okamoto, and M. Tambe. Allocating tasks in extreme teams. In *Proc. of AAMAS*, pages 727–734. ACM, 2005.
48. P. Scerri, D. Pynadath, L. Johnson, P. Rosenbloom, M. Si, N. Schurr, and M. Tambe. A prototype infrastructure for distributed robot-agent-person teams. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '03, pages 433–440, New York, NY, USA, 2003. ACM.
49. P. Scerri, M. Tambe, and D. V. Pynadath. Towards adjustable autonomy for the real-world. *Journal of Artificial Intelligence Research*, 17(1):171–228, 2002.
50. N. Schurr, J. Marecki, J. P. Lewis, M. Tambe, and P. Scerri. The defacto system: Training tool for incident commanders. In *National Conference on Artificial Intelligence (AAAI)*, pages 1555–1562, 2005.
51. J. Searle. A taxonomy of illocutionary acts. In K. Günderson, editor, *Language, Mind, and Knowledge, (Studies in the Philosophy of Science*, volume 7, pages 344–69. University of Minneapolis Press, 1975.
52. S. P. Simonović. *Systems Approach to Management of Disasters*. John Wiley & Sons, Inc., Hoboken, NJ, USA, Nov. 2010.
53. C. Skinner and S. D. Ramchurn. The robocup rescue simulation platform. In *AAMAS*, pages 1647–1648. IFAAMAS, 2010.
54. G. Sukthankar, K. Sycara, J. A. Giampapa, and C. Burnett. Communications for agent-based human team support. *Handbook of Research on Multi-agent Systems: Semantics and Dynamics of Organizational Models*, page 285, 2009.

55. Z. O. Toups, A. Kerne, and W. A. Hamilton. The team coordination game. *ACM Transactions on Computer-Human Interaction*, 18(4):1–37, Dec. 2011.
56. T. Wagner, J. Phelps, V. Guralnik, and R. VanRiper. An application view of coordinators: Coordination managers for first responders. *AAAI*, 2004.

A Radiation Cloud Modelling

The radiation cloud diffusion process is modelled using the Smoluchowski drift-diffusion equation,

$$\frac{D\mathrm{Rad}(\mathbf{z},\tau)}{D\tau} = \kappa\nabla^2\mathrm{Rad}(\mathbf{z},\tau) - \mathrm{Rad}(\mathbf{z},\tau)\nabla\cdot\mathbf{w}(\mathbf{z},\tau) + \sigma(\mathbf{z},\tau) \tag{3}$$

where D is the material derivative, $\mathrm{Rad}(\mathbf{z},\tau)$ is the radiation cloud intensity at location $\mathbf{z} = (x, y)$ at time τ, κ is a fixed diffusion coefficient and σ is the radiation source(s) emission rate. The diffusion equation is solved on a regular grid defined across the environment with grid coordinates G (as defined in Section 3.1). Furthermore, the grid is solved at discrete time instances τ. The cloud is driven by stochastic wind forces which vary both spatially and temporally. These forces induce anisotropy into the cloud diffusion process proportional to the local average wind velocity, $\mathbf{w}(\mathbf{z},\tau)$. The wind velocity is drawn from two independent Gaussian processes (GP), one GP for each Cartesian coordinate axis, $w_i(\mathbf{z},\tau)$, of $\mathbf{w}(\mathbf{z},\tau)$. The GP captures both the spatial distribution of the wind velocity and the dynamic process resulting from shifting wind patterns (e.g., short term gusts and longer term variations).

In our simulation, each spatial wind velocity component is modelled by an isotropic squared-exponential GP covariance function [43], K, with fixed input and output scales, l and μ, respectively (although any covariance function can be substituted),

$$K(\mathbf{z},\mathbf{z}') = \mu^2 \exp -(\mathbf{z}-\mathbf{z}')^T\mathbf{P}^{-1}(\mathbf{z}-\mathbf{z}')$$

where $\mathbf{P}$ is a diagonal covariance matrix with diagonal elements l^2. This choice of covariance function generates wind conditions which vary smoothly in both magnitude and direction across the terrain. Furthermore, as wind conditions may change over time we introduce a temporal correlation coefficient, ρ, to the covariance function. Thus, for a single component, w_i, of $\mathbf{w}$, defined over grid G at times τ and τ', the wind process covariance function is, $\mathrm{Cov}(w_i(\mathbf{z},\tau), w_i(\mathbf{z}',\tau')) = \rho(\tau,\tau')K(\mathbf{z},\mathbf{z}')$. We note that, when $\rho = 1$ the wind velocities are time invariant (although spatially variant). Values of $\rho < 1$ model wind conditions that change over time.

Using the above model, we are able to create a moving radiation cloud. This poses a real challenge both for the HQ (PA and H) and the responders on the ground as the predictions they make of where the cloud will move to will be prone to uncertainty both due to the simulated wind speed and direction. While it is possible to use radiation readings provided by first responders on the ground, as they move in the disaster space, in our trials, we assumed that these readings are coming from sensors already embedded in the environment to allow the team to focus on path planning for task allocation (which is the focus of this paper) rather than for radiation monitoring. Hence, using such sensor readings, the prediction algorithm provided in Appendix B is then used to provide estimates of the radiation levels across the disaster space during the game. These estimates are displayed as a heat map as described in Section 5.

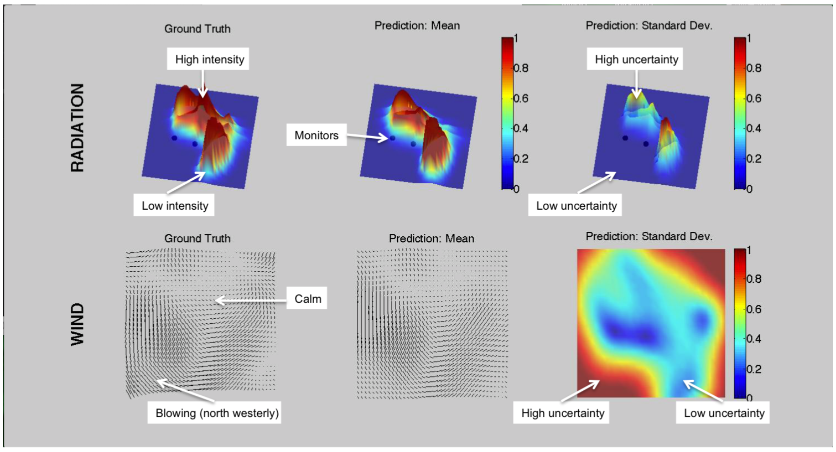
B Predictive Model of Radiation Cloud

Predictions of the clouds location are performed using a latent force model (LFM) [45,44]. The LFM is a Markov model that allows the future state of the cloud and wind conditions to be predicted efficiently from the current state. Predictions are computed using the Extended Kalman filter (EKF) which has a linear computational complexity with regard to the time interval over which the dynamics are predicted forward. [11] The EKF estimates provide both the mean and variance of the state of the cloud and wind conditions. Figure 5 shows example cloud simulations for slow varying (i.e. $\rho = 0.99$) and gusty (i.e. $\rho = 0.90$) wind conditions. The left panes in each subfigure show the ground truth simulation obtained by sampling from the LFM. The middle panes show the mean of the cloud and wind conditions and the right panes show the uncertainty in the conditions.

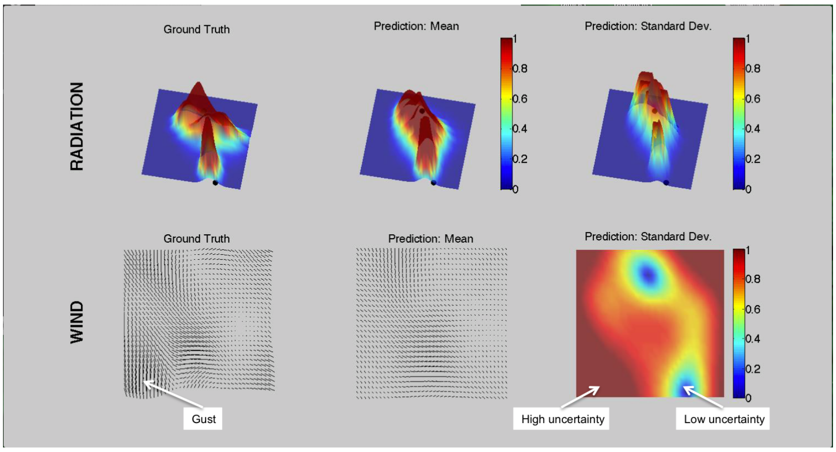
The radiation is monitored using a number of sensors on the ground that collect readings of the radiation cloud intensity and, optionally, wind velocity every minute of the game. These *monitor agents* can be at fixed locations or they can be mobile agents equipped with geiger-counters that inform the user and commander of the local radiation intensity. The measurements can be folded into the EKF and this refines estimates of both the radiation cloud and wind conditions across the grid. Figure 5 shows the impact of such measurements on the uncertainty of the cloud and wind conditions. Location of two monitors are shown as black dots in the upper row of panes in both subfigures. The right most panes show the relative uncertainty in both the

[11] The EKF accommodates the nonlinearities in the radiation dynamics expressed through equation (3).

cloud and wind conditions as a result of current and past measurements. Figure 5(a) shows slow varying wind conditions in which case the radiation cloud can be interpolated accurately using sparse sensor measurements and the LFM model. Alternatively, during gusty conditions the radiation cloud model is more uncertain far from the locations where recent measurements have been taken, as shown in Figure 5(b).



(a) Slowly varying wind conditions



(b) Gusty wind conditions

Fig. 5 Radiation and wind simulation ground truth and EKF estimates obtained using measurements from monitor agents (black dots). Left most panes are ground truth radiation and wind conditions, the middle panes are corresponding estimates and right most panes are state uncertainties: (a) invariant and (b) gusty wind conditions. The radiation levels are normalised to the range $[0,\ 1]$.

C Simulation Results of MMDP Solution

Before deploying our solution (as part of PA) to advise human responders, it is important to test its performance to ensure it can return efficient solutions on simulations of the real-world problem. Given there is no extant solution that takes into account uncertainty in team coordination for emergency response, we compare our algorithm with a greedy and a myopic method to evaluate the benefits of coordination and lookahead. For each method, we use our path planning algorithm to compute the path for each responder. In the greedy method, the responders are uncoordinated and select the closest tasks they can do. In the myopic method, the responders are coordinated to select the tasks but have no lookahead for the future tasks (Line 8 in Algorithm 2). Table 3 shows the results for a problem with 17 tasks and 8 responders on a 50×55 grid. As can be seen, our MMDP algorithm completes more tasks than the myopic and greedy methods (see Table 3). More importantly, our algorithm guarantees the safety of the responders, while in the myopic method only 25% of the responders survive and in the greedy method all responders are killed by the radioactive cloud. More extensive evaluations are beyond the scope of this paper as our focus here is on the use of the algorithm in a field deployment to test how humans take up advice computed by the planning agent PA.

Table 3 Experimental results for the MMDP, myopic, greedy algorithms in simulation.

	MMDP	Myopic	Greedy
No. of completed tasks	71%	65%	41%
No. responders alive at the end	100%	25%	0%