Human-Agent Collaboration for Real-World Disaster Response

Tracking No. 577

ABSTRACT

The allocation of emergency responders to rescue tasks is a key application area for agent-based coordination algorithms. However, to date, none of the proposed approaches take into account the inherent uncertainty in disaster scenarios and, crucially, none have been field-trialled in order to understand how humans perform when instructed by an agent. Hence, in this paper, we propose a novel representation and an algorithm, using Multi-agent Markov Decision Processes, to solve the problem of coordinating emergency responders in uncertain environments. Moreover, we deploy our algorithm in a mixed-reality game called AtomicOrchid, with an agent that works alongside a human commander to guide human players to complete rescue tasks. In our trials, our algorithm is shown to improve performance significantly (compared to a no-agent setting) and our results allow us to elucidate some of the key challenges faced when developing systems involving human-agent collaboration.

Keywords

Human-Agent Interaction, Agents for improving human cooperative activities, Disaster Response.

1. INTRODUCTION

The coordination of teams of first responders in search and rescue missions is a grand challenge for multi-agent systems research [7]. In such settings, responders with different capabilities (e.g., fire-fighting or life support) have to form teams in order to perform rescue tasks (e.g., extinguishing a fire or providing first aid) to minimise loss of life and costs (e.g., time or money). Thus, responders have to plan their paths to the tasks (as these may be distributed in space) and form specific teams to complete them. These teams, in turn, may need to disband and reform in different configurations to complete new tasks, taking into account the status of the current tasks (e.g., health of victims or building fire) and the environment (e.g., if a fire or radioactive cloud is spreading). Furthermore, uncertainty in the environment (e.g., wind direction or speed) or in the responders' abilities to complete tasks (e.g., some may be tired or get hurt) means that plans are likely to change continually to reflect the prevailing assessment of the situation.

To address these challenges, a number of algorithms and mechanisms have been developed to form teams and allocate tasks. For example, [12, 13] and [4], devised centralised and decentralised optimisation algorithms respectively to allocate rescue tasks efficiently to teams of first responders with different capabilities. However, none of these approaches considered 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 (representing responders as computational entities), none of them have been *exercised* to guide *real* human responders in real-time rescue missions. Thus, it is still unclear whether these algorithms will cope with real-world uncertainties (e.g., communication breakdowns or changes in wind direction), be acceptable to humans (i.e., be clear for humans and take into account their capabilities), and actually augment, rather than hinder, human performance.

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. In more detail, we consider a scenario involving rescue tasks distributed in a physical space over which a (virtual) radioactive cloud is spreading. Tasks need to be completed by the responders before the area is completely covered by the cloud (as responders will die from radiation exposure) which is spreading according to varying wind speed and direction. 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 ensures responders are not exposed to significant radiation. The algorithm is then used by an agent to guide human responders. Specifically, it is integrated into our test platform, AtomicOrchid, that structures the interaction between human responders, a human commander, and the planning agent in a mixed-reality location-based game. 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. In particular, we advance the state of the art in the following ways:

1. We develop a novel representation for team coordination (i.e., path planning and task allocation) under uncertainty using Multi-agent Markov Decision Processes (MMDP) [3]. Moreover, we provide an algorithm to approximate solutions to the MMDP and show how it is adaptive to human requests to replan task allocations within a team.

2. We present AtomicOrchid, a novel game to evaluate team coordination under uncertainty using the concept of mixed-reality games. AtomicOrchid allows an agent, using our task planning algorithm, to coordinate, in real-time, human players using mobile phone-based messaging, to complete rescue tasks efficiently.

Appears in: *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014), Lomuscio, Scerri, Bazzan, Huhns (eds.), May, 5–9, 2014, Paris, France.*

3. We provide a real-world evaluation of a task allocation agent in a disaster response scenario in multiple field trials and present both quantitative and qualitative results.

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 to for systems involving human-agent collaboration.

The rest of this paper is structured as follows. Section 2 formalises the disaster response problem as a MMDP. Section 3 describes the algorithm to solve the path planning and task allocation problems presented by the MMDP. Section 4 details the AtomicOrchid platform. Section 5 presents our pilot study and the field trial evaluation. Finally, Section 6 concludes.

2. THE DISASTER SCENARIO

We consider a disaster scenario in which a satellite, powered by radioactive fuel, has crashed in a sub-urban area.[1] 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. Hence, emergency services (including medics, soldiers, and fire-fighters) are deployed to evacuate the casualties and key assets (e.g., food reserves, medication, fuel), each requiring different teams of responders, 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.

2.1 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.[2] 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 first responders 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 responders 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, 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.

[1]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 4).

[2]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.

2.2 Human-Agent Collaboration

In line with practice in many countries, we assume that the first responders are coordinated from a headquarters (HQ) headed by a human coordinator H. In our case, H may be assisted by an autonomous task planning agent PA (more details in Section 3), that can receive input from, and direct, the first responders. 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 (for reasons outlined above) and may therefore simply accept or reject the received instruction from PA or H.[3] 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 3.1.3) 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).

It is important to note that our model captures different types of flexible control: (i) agent-based: when PA directly instructs the responders and they can use multiple iterations of accept/reject to collaboratively converge to a definite plan (ii) human-based: when H communicates goals to the responders in open-ended interactions that may permit H to gain a better understanding of the context than PA and therefore formulate corrective measures faster. Moreover, in contrast to previous work that suggested *transfer-of-control* regimes [14], our approach 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 4) to allow human control at any point (and our results in Section 5 validate this approach).

2.3 The Optimisation Problem

Previous agent-based models for team coordination in disaster response typically assume deterministic task executions and environments [12, 13]. However, in order to evaluate agent-guided coordination in a real-world environment, it is important to consider uncertainties due to player behaviours and the environment (as discussed in the previous section). Given this, we propose a new representation for the task allocation problem in disaster response that

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

does take into account such uncertainties. More specifically, we represent this problem using an MMDP that captures the uncertainties of the radioactive cloud and the responders' behaviours. We model the spreading of the radioactive cloud as a random process over the disaster space and allow the actions requested from the responders to fail (because they decline to go to a task) or incur delays (because they are too slow) during the rescue process. Thus in the MMDP model, we represent task executions as stochastic processes of state transitions, while the uncertainties of the radioactive cloud and the responders' behaviours can be easily captured with transition probabilities. More formally, the MMDP is represented by tuple $\mathcal{M} = \langle I, S, \{A_i\}, P, R\rangle$, where I is the set of actors as defined in the previous section, S is the state space, A_i is a set of responder p_i's actions, P is the transition function, and R is the reward function. We elaborate on each of these below.

In more detail, $S = S_r^G \times S_{p_1} \times \cdots \times S_{p_n} \times S_{t_1} \times \cdots \times S_{t_m}$ where $S_r^G = \{l_{(x,y)} | (x,y) \in G\}$ is the state variable of the radioactive cloud that specifies the radioactive level $l_{(x,y)} \in [0, 100]$ at every point $(x, y) \in G$. $S_{p_i} = \langle h_i, (x_i, y_i), t_j\rangle$ is the state variable for each responder p_i that specifies her health level $h_i \in [0, 100]$, her present position (x_i, y_i), and the task t_j she is carrying out. $S_{t_j} = \langle \mathtt{st_j}, (x_j, y_j)\rangle$ is then the state variable for task t_j to specify its status $\mathtt{st_j}$ (i.e., the target is picked up, dropped off, or idle) and position (x_j, y_j).

The three types of actions (in set A_i) a responder can take are: (i) *stay* in the current location (x_i, y_i), (ii) *move* to the 8 neighbouring locations, or (iii) *complete* a task located at (x_i, y_i). A joint action $\vec{a} = \langle a_1, \cdots, a_n\rangle$ is a set of actions where $a_i \in A_i$, one for each responder (a responder may just *stay* at its current position if it has no targets to rescue). The transition function P is defined in more detail as: $P = P_r \times P_{p_1} \times \cdots \times P_{p_n} \times P_{t_1} \times \cdots \times P_{t_m}$ where:

- $P_r(s'_r|s_r)$ is the probability the radioactive cloud spreads from state $s_r \in S_r^G$ to $s'_r \in S_r^G$. It captures the uncertainty of the radiation levels in the environment due to noisy sensor readings and the variations in wind speed and direction.
- $P_{p_i}(s'_{p_i}|s, a_i)$ is the probability responder p_i transitions to a new state $s'_{p_i} \in S_{p_i}$ when executing action a_i. For example, when a responder is asked to go to a new location, she may not end up there because she is tired, gets injured, or receives radiation doses that are life threatening.
- $P_{t_j}(s'_{t_j}|s, \vec{a})$ is the probability of task t_j being completed. A task t_j can only be completed by a team of responders with the required types (Θ_{t_j}) located at the same position as t_j.

Now, if t_j is completed (i.e., in $\mathtt{st_j} \in S_{t_j}$, the status $\mathtt{st_j}$ is marked as "dropped off" and its position (x_j, y_j) is within a safe zone), the team will be rewarded using function R. The team is penalised if a responder p_i gets injured or receives a high dose of radiation (i.e., in s_{p_i}, the health level h_i is 0). Moreover, we attribute a cost to each of the responders' actions since each action requires them to exert some effort (e.g., running or carrying objects).

Give the above definitions, a policy for the MMDP is a mapping from states to joint actions, $\pi : S \rightarrow \vec{A}$ so that the responders know which actions to take given the current state of the problem. The quality of a policy π is measured by its expected value V^π, which can be computed recursively by the Bellman equation:

$$V^\pi(s^\tau) = R(s^\tau, \pi(s^\tau)) + \sum_{s^{\tau+1} \in S} P(s^{\tau+1}|s^\tau, \pi(s^t))V^\pi(s^{\tau+1}) \quad (1)$$

where τ denotes the current time point and $\pi(s^\tau)$ is a joint action given s^τ. The goal of solving the MMDP is to find an optimal policy π^* that maximises the expected value with the initial state s^0, $\pi^* = \arg\max_\pi V^\pi(s^0)$.

Algorithm 1: Team Coordination Algorithm

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

```
// The task planning
{t^i} ← compute the best task for each responder p_i ∈ I
foreach p_i ∈ I do
    // The path planning
    a_i ← compute the best path to task t^i
return a⃗
```

At each decision step, we assume the planning agent can fully observe the state of the environment s by collecting sensor readings of the radioactive cloud and GPS locations of the responders. Given a policy π of the MMDP, a joint action $\vec{a} = \pi(s)$ can be selected and broadcast to the responders (as mentioned earlier).

3. TEAM COORDINATION ALGORITHM

Unfortunately, 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 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.

3.1 Task Planning

As described in Section 2.3, 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 respon-

ders 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.

3.1.1 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 central limit theorem, 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 $v(C_{jk}) \leftarrow \frac{1}{N} \sum_{i=1}^{N} v_i(C_{jk})$
11 **return** the task assignment computed by Equation 3

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) [8] (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.[4] 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 3 using the team values estimated by the UCB1 heuristic [1] to balance exploitation and exploration:

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

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.

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 $\{C_{jk}\}$.

3.1.2 Coordinated Task Allocation

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

$$\begin{array}{ll} \max\limits_{x_{jk}} & \sum_{j,k} x_{jk} \cdot v(C_{jk}) \\ \text{s.t.} & x_{jk} \in \{0, 1\} \\ & \forall j, \sum_k x_{jk} \leq 1 \quad \text{(i)} \\ & \forall i, \sum_{j,k} \delta_i(C_{jk}) \leq 1 \quad \text{(ii)} \end{array} \tag{3}$$

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 solvers (e.g., IBM CPLEX or lp_solve).

3.1.3 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 3 (by default, all teams have identical weights of 1.0). Thus, our approach is adaptive to the preferences of human responders.

3.2 Path Planning

In the path planning phase, we compute the best path for a responder to her assigned task. This phase is stochastic as there are 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

[4] Other methods such as sequential greedy assignment or swap-based hill climbing [11] may also be useful. However, they do not explore the policy space as well as MCTS.

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 [9]). We choose Real-Time Dynamic Programming (RTDP) [2] 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. Another speed up is also possible if, when traversing the possible states, we only consider the responder's current location and the neighbouring points. This will further speed up the algorithm where the main bottleneck is the huge state space.

3.3 Simulation Results

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 1 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 1). 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 1: 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%

4. THE AtomicOrchid GAME

In this section we describe the AtomicOrchid game used to embed the planning agent in order to trial mixed-initiative coordination. We adopt a serious mixed-reality games approach to counteract the limitations of computational simulations [6]. 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" [16]. The impact of emotional and physical responses likely in a disaster, such as stress, fear, exertion or panic remains understudied in approaches that rely purely on computational simulation [5]. In contrast, our approach creates a realistic setting in the sense that participants experience physical exertion and stress through bodily activity and time pressure, mirroring aspects of a real disaster setting [10]. This, in turn, provides greater confidence in the efficacy of behavioural observations regarding team coordination supported by a planning agent.

In more detail, AtomicOrchid is a location-based mobile game based on the fictitious scenario described in Section 2. First responders are assigned a specific type: medic, fire-fighter, transporter, soldier, or ambulance. Their mission is to evacuate all four types of targets: victim (requires medic and fire-fighter), animal (requires medic and ambulance), fuel (requires soldier and fire-fighter), or other resource (requires soldier and ambulance). The first responders are supported by (at least) one person H 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, the interactions with the planning agent, and the modelling of the radiation cloud in the game. A video of the operation of AtomicOrchid can be viewed at: `http://bit.ly/1ebNYty`.

4.1 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.

Figure 1: Mobile first responder and HQ interfaces.

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 1). 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). Crucially, only H and PA have a view of the radioactive cloud, graphically depicted as a heatmap ('Hotter' (red) zones correspond to higher levels of radiation).

4.2 System Architecture

AtomicOrchid is based on the open-sourced geo-fencing game MapAttack[5] 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.

4.3 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.
- *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 3.1.3). 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.

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.

4.4 Radiation Cloud Modelling

The radiation cloud is assumed to be monitored using a number of sensors on the ground (within the disaster space) that collect readings of the radiation cloud intensity and wind velocity every minute of the game. These sensors can be at fixed locations or held by mobile agents. The radiation cloud diffusion process is modelled using the Smoluchowski drift-diffusion equation,

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

where D is the material derivative, $\text{Rad}(\mathbf{z},\tau)$ 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 2.3). 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).

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.

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

5. THE FIELD-TRIALS

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.

5.1 Participants and Procedure

A total of 24 participants (17 males and 5 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.

5.2 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 players (8 each). Thus, players were unique to a session to avoid learning effects between sessions. 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.

5.3 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 [15] 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.
- Directives: *speech acts that are meant to cause the hearer to take a particular action*, e.g. requests, commands and advice, including task and team allocation messages.

5.4 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). In fact, one responder 'died' in Session A. 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).

	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

Table 2: Message classification. FR: First Responder.

Table 2 shows the remote message directives (mainly related to task allocation and execution) and assertives (mainly related to situational awareness) sent in the sessions. The following discussion draws on how these messages were handled to give a sense of mixed-initiative coordination in the game sessions.

In particular, Figure 2 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. 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.

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

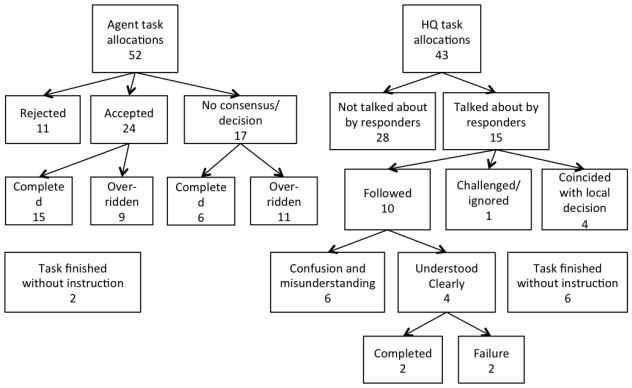

Figure 2: How task allocations were handled by first responders in the version with agent (left), and without agent (right).

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 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 3) 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 particular, we propose the following design guidelines for human-agent collaborations:

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 dynamically. We believe this makes the algorithm more acceptable to the users.

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.

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 [14]) and, therefore, left the coordinator the option of taking control when she judged it was needed. 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.

6. 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 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. Finally our results suggest that systems involving human-agent collaboration should be adaptive, involve simple interactions between humans and agents, and allow for flexible autonomy. Future work will look at running AtomicOrchid with expert responders and exploring different interactional arrangements of humans and agents, in particular where control may be distributed across the team.

7. REFERENCES

[1] 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.

[2] 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.

[3] C. Boutilier. Planning, learning and coordination in multi-agent decision processes. In *Proc. of TARK 1996*, pages 195–210, 1996.

[4] 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.

[5] 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*, 48(3):487–506, 2009.

[6] J. E. Fischer, M. Flintham, D. Price, J. Goulding, N. Pantidi, and T. Rodden. Serious mixed reality games. In *Mixed Reality games Workshop at the 2012 ACM Conference on Computer Supported Cooperative Work*, 2012.

[7] H. Kitano and S. Tadokoro. Robocup rescue: A grand challenge for multiagent and intelligent systems. *AI Magazine*, 22(1):39–52, 2001.

[8] L. Kocsis and C. Szepesvári. Bandit based monte-carlo planning. In *Proc. of ECML 2006*, pages 282–293, 2006.

[9] Mausam and A. Kolobov. Planning with markov decision processes: An AI perspective. *Synthesis Lectures on AI and Machine Learning*, 6(1):1–210, 2012.

[10] Pan American Health Organization. Stress management in disasters, 2001.

[11] S. Proper and P. Tadepalli. Solving multi-agent assignment Markov decision processes. In *Proc. of AAMAS 2009*, pages 681–688, 2009.

[12] 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.

[13] P. Scerri, A. Farinelli, S. Okamoto, and M. Tambe. Allocating tasks in extreme teams. In *Proc. of AAMAS*, pages 727–734. ACM, 2005.

[14] 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.

[15] 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.

[16] S. P. Simonovic. Systems approach to management of disasters: Methods and applications. In *Disaster Prevention and Management*, volume 20. Wiley, 2009.