Journal of Artificial Intelligence Research 57 (2016) 661-708 Submitted 06/16; published 12/16

# A Disaster Response System based on Human-Agent Collectives

**Sarvapali D. Ramchurn** SDR1@SOTON.AC.UK
**Trung Dong Huynh** TDH@ECS.SOTON.AC.UK
**Feng Wu** WUFENG02@USTC.EDU.CN
**Yuki Ikuno** YI1G13@SOTON.AC.UK
**Jack Flann**
**Luc Moreau** L.MOREAU@ECS.SOTON.AC.UK
*Dept. of Electronics and Computer Science,*
*University of Southampton, Southampton, UK.*

**Joel E. Fischer** JOEL.FISCHER@NOTTINGHAM.AC.UK
**Wenchao Jiang** PSXWJ@NOTTINGHAM.AC.UK
**Tom Rodden** TAR@CS.NOTT.AC.UK
*Mixed-Reality Lab*
*Dept. of Computer Science, University of Nottingham, UK.*

**Edwin Simpson** EDWIN@ROBOTS.OX.AC.UK
**Steven Reece** REECE@ROBOTS.OX.AC.UK
**Stephen Roberts** SJROB@ROBOTS.OX.AC.UK
*Machine Learning Research Group,*
*Department of Engineering Science, University of Oxford, UK.*

**Nicholas R. Jennings** N.JENNINGS@IMPERIAL.AC.UK
*Imperial College London, UK.*

## Abstract

Major natural or man-made disasters such as Hurricane Katrina or the 9/11 terror attacks pose significant challenges for emergency responders. First, they have to develop an understanding of the unfolding event either using their own resources or through third-parties such as the local population and agencies. Second, based on the information gathered, they need to deploy their teams in a flexible manner, ensuring that each team performs tasks in the most effective way. Third, given the dynamic nature of a disaster space, and the uncertainties involved in performing rescue missions, information about the disaster space and the actors within it needs to be managed to ensure that responders are always acting on up-to-date and trusted information. Against this background, this paper proposes a novel disaster response system called HAC-ER. Thus HAC-ER interweaves humans and agents, both robotic and software, in social relationships that augment their individual and collective capabilities. To design HAC-ER, we involved end-users including both experts and volunteers in a several participatory design workshops, lab studies, and field trials of increasingly advanced prototypes of individual components of HAC-ER as well as the overall system. This process generated a number of new quantitative and qualitative results but also raised a number of new research questions. HAC-ER thus demonstrates how such Human-Agent Collectives (HACs) can address key challenges in disaster response. Specifically, we show how HAC-ER utilises crowdsourcing combined with machine learning to obtain most important situational awareness from large streams of reports posted by members of the public and trusted organisations. We then show how this information can inform human-agent teams in coordinating multi-UAV deployments, as well as task planning for responders on the ground. Finally, HAC-ER incorporates an infrastructure and the associated intelligence for tracking and utilising the provenance of information shared across the entire system to ensure its accountability. We individually validate each of these elements of HAC-ER and show how they perform against standard (non-HAC) baselines and also elaborate on the evaluation of the overall system.

## 1. Introduction

In the aftermath of major disasters (natural or man-made), such as the Haiti earthquake of 2010 or the 9/11 terror attacks on the World Trade Centre, emergency response agencies face a number of key challenges (Villaveces, 2011). First, it is vital to gain *situational awareness* of the unfolding event to determine where aid is required and how it can be delivered, given that infrastructure may be damaged. Useful information can come from a variety of sources, including people on the ground, relief agencies, and satellite imagery (e.g., in the aftermath of the Haiti earthquake, maps were constructed rapidly using crowdsourcing — see Section 2.1). However, making sense of this information is a painstaking process, particularly as the sources are liable to noise, bias, and delays. Second, emergency response agencies typically gather more accurate information which can best be achieved by deploying unmanned aerial vehicles (UAVs) equipped with onboard cameras and sensors. Now, using multiple UAVs avoids risking human life, but involves additional complexity in controlling such vehicles and visualising the information they feed back (Cummings, Brzezinski, & Lee, 2007). Specifically, tasks should be allocated to maximise the amount of useful information collected, whilst considering limited battery capacity and ensuring human coordinators are not overwhelmed by the need to manually operate individual UAVs. The third challenge is to use situational awareness to allocate relief tasks to First Responders (FRs). For example, digging people out of the rubble, moving water treatment units to populated areas, or extinguishing fires. It is crucial to consider the time required to travel to each task, as travel time is non-productive in terms of responding to the disaster (Kitano & Tadokoro, 2001). However, the capabilities of individual responders must be considered to ensure that all tasks can be performed effectively and that no one is put in harm's way. For example, it may not be suitable to allocate medics to densely built-up areas where a fire is spreading or to attend casualties during riots.[1] Finally, given that disaster environments are highly uncertain and dynamic due to the unfolding event, it is crucial that emergency response agencies can track and verify the information and decisions that they use, allowing them to modify or reinforce the current course of action whenever new information is detected or previous assumptions are no longer correct (e.g., as verified by trusted actors on the ground or other organisations).

In this work, we consider both practical and theoretical work in AI that can be used to support the work of FRs in all of the above tasks. We draw upon the realisation that such work cannot be done solely by humans or machines working individually. Rather they need to collaborate. In particular, these tasks not only require the computational power to cope with the complexity of information that is generated during major disasters, but also require human understanding of the unfolding situation that allows them to prioritise tasks and oversee the operation of autonomous entities. Hence, we explore novel ways in which humans and agents (including robotic agents) can be teamed up to achieve a level of effectiveness that is greater than the sum of the parts. Specifically, we draw upon the work of Jennings, Moreau, Nicholson, Ramchurn, Roberts, Rodden, and Rogers (2014) which describes such systems of humans and agents as Human-Agent Collectives or HACs. HACs interleave human and machine decision-making in such a way that control shifts flexibly between humans and agents working as part of agile teams. The operation of such collectives is tracked using an information infrastructure that ensures that the system and its actors are accountable for the decisions they make and information they generate during a disaster response operation.

Previous work on disaster response (Kitano & Tadokoro, 2001; Koes, Nourbakhsh, & Sycara, 2006; Ramchurn, Farinelli, Macarthur, & Jennings, 2010) in the AI community has tended to ad-

[1] In Haiti, such information was shared via the Ushahidi platform (`http://www.ushahidi.com`) to help coordinate emergency responders of different types.

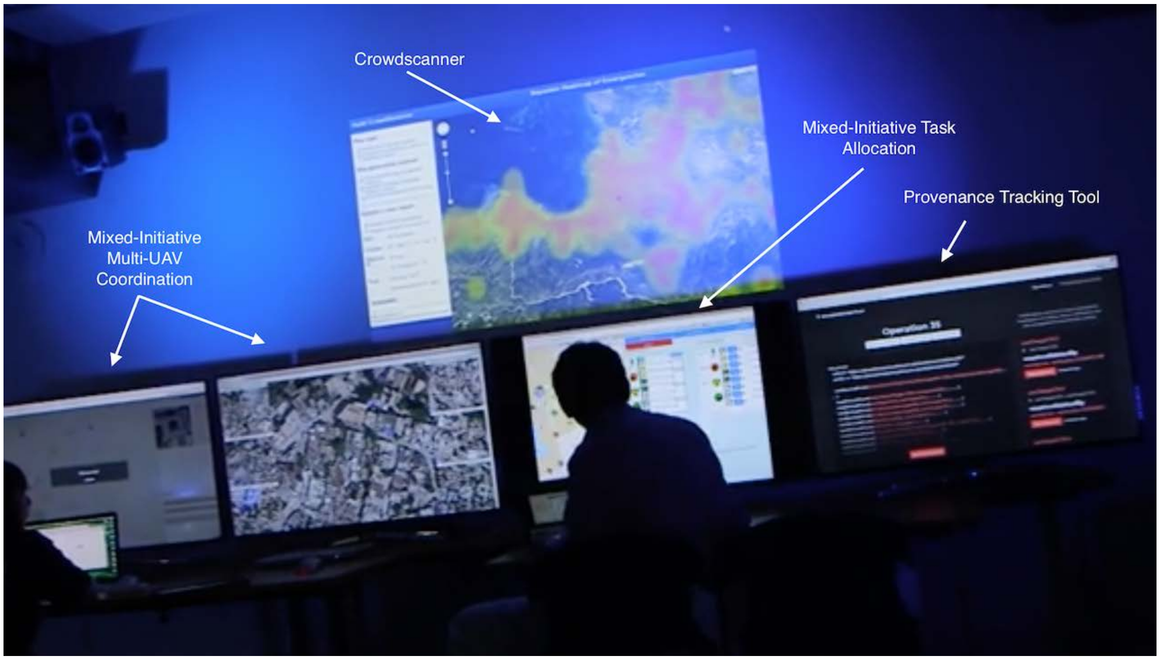

Crowdscanner
Mixed-Initiative Task Allocation
Provenance Tracking Tool
Mixed-Initiative Multi-UAV Coordination

Figure 1: The HAC-ER system in action.

dress some of these problems in isolation and typically under simulation. For example, a number of approaches to managing information from crowdsourcing initiatives have focused on the algorithmics of more efficient and accurate information gathering (Kamar, Hacker, & Horvitz, 2012) or how to extract sentiment information from reports (Simpson, Venanzi, Reece, Kohli, Guiver, Roberts, & Jennings, 2015), while ignoring how such information will be assimilated by FRs. In terms of task allocation and coordination of FRs, a number of decentralised coordination algorithms have been developed by the multi-agent systems community (e.g., Amador, Okamoto, & Zivan, 2014; Delle Fave, Rogers, Xu, Sukkarieh, & Jennings, 2012; Macarthur, Stranders, Ramchurn, & Jennings, 2011; Wu & Jennings, 2014; Chen, Wu, Shen, Chen, & Ramchurn, 2015). However, none of these approaches has been integrated into live tools that can be used by non-expert operators. Moreover, although there is much literature in planning support, task assignment, and human-agent collaboration, very few real world studies of how human emergency response teams actually handle agent support have been carried out. The most relevant work is by Scerri et al., (2003) and Schurr et al., (2005), but they both focus on humans acting as peers to agents in computational simulations rather than real-world deployments in the field. Hence, to determine the usefulness of AI-based solutions, there is an urgent need to evaluate these technologies with human users before they are deployed in real-world settings. As such, our aim in this work is to bridge the gap that exists between such technologies and their application in the real-world. In particular, we envisage that different forms of HACs, interleaving human and agents in different kinds of relationships (e.g., human as commander and agent as advisor or humans as workers and agent as task master), or *interactional arrangements* will be necessary to address the different coordination challenges posed by crowdsourcing systems, multi-UAV deployments, and the teaming of responders on the ground.

Against this background, we propose a prototype disaster management system called Human-Agent Collectives for Emergency Response or HAC-ER (pronounced ‘hacker’) that demonstrates how humans and agents can coalesce into HACs to address the above challenges. We designed our system collaboratively with FRs from Rescue Global (an NGO specialised in disaster response), Hampshire Fire and Rescue, and other organisations in the UK, and trialled our system with over 100 users, to determine how HACs can support emergency response in different activities. The system was iteratively developed through a participatory design process involving a number of workshops and lab studies with both experts and volunteers. These activities generated both quantitative and qualitative results that led to a coherent design for the whole system but also raised new questions about how such complex systems involving human and machine decision making should be developed. In more detail, we develop a number of HACs as follows (as shown in Figure 1):

- Crowdscanner: a HAC that relies on crowdsourcing to gather, interpret and fuse information from both trusted agencies and members of the public on the ground, and thereby determine priority areas for responders. This component is evaluated on real-world data from the Haiti earthquake and the results show that it is able to predict emergencies (e.g., fires, request for help etc.) accurately. This component of HAC-ER was also deployed in a real-world context during the Nepal earthquake in 2015.
- Mixed-Initiative Multi-UAV Coordination: a HAC that involves agents running a distributed algorithm to coordinate multiple UAVs, as well as human operators, with different roles. Humans and agents interact through planning and monitoring interfaces and the overall system allows them to verify information reported by civilians and prioritise search areas.
- Mixed-Initiative Task Allocation: a HAC composed of a planning agent and responders at the disaster response headquarters and on the ground, who work together to determine a schedule for the completion of tasks as identified by the multi-UAV system. To demonstrate how such a HAC can be evaluated in the wild, we report on field trials of this HAC. From both qualitative and quantitative results from these trials, we derive new design guidelines for agent-based team planning.

In order to provide accountability for both human and agent-based decision making, the provenance of information and decisions generated by the above HACs are fully tracked by a provenance infrastructure, which was built on top of the provenance standards set by the World Wide Web Consortium (see Groth & Moreau, 2013). Hence, another key contribution of this paper is a provenance agent to monitor the recorded provenance data to keep stakeholders in the system aware of changes in the situational awareness of the collective during an operation. This, in turn, helps them react to new information in a timely manner.

In general, HAC-ER was the successful outcome of a large multi-institution and multi-disciplinary project that lasted 5 years. It involved researchers from diverse backgrounds including HCI, Design Ethnography, Agent-Based Computing, and Machine learning. The success of such large projects can be largely attributed, among other things, to the fact that the investigators involved provided leadership in going beyond their traditional research areas to collaboratively break new ground at the intersection of multiple disciplines (see Section 9 for more detailed reflections).

The rest of this paper is structured as follows. Section 2 discusses the related work, while Section 3 discusses the decision-making challenges addressed by HAC-ER and the system architecture that brings together all of its components. Section 4 presents our crowdsourcing support tool, Section 5 describes our mixed-initiative UAV command interfaces, and Section 6 shows how we allocate FRs to rescue targets. Section 7 describes how information and decisions are tracked and monitored by a provenance agent to guarantee accountability and system reliability in dynamic environments. Section 8 provides an overall evaluation of HAC-ER, and finally, Section 9 concludes the paper with a discussion of the barriers to the deployment of HACs of the type studied in this investigation and describe the future research that arises from the open questions we have been unable to address so far.

## 2. Related Work

In this section, we discuss the literature underpinning the key technical and practical contributions of the paper. In particular, we discuss crowdsourcing techniques typically deployed in disaster response, as well as the use of supervisory control systems to manage teams of UAVs to gather situational awareness. We then elaborate on the different aspects of task allocation for human responders, namely (i) how humans and agents collaborate for rescue mission planning and (ii) how a planner agent models the uncertainty in the environment and the behaviour of FRs when receiving requests to perform tasks. Finally, we provide some brief background on the provenance tracking methods we employ within HAC-ER.

### 2.1 Crowdsourcing for Situational Awareness

With increasingly widespread access to mobile phones and the Internet, people affected by a disaster now routinely post text messages and photographs to social media sites, requesting assistance and describing the situation on the ground. Platforms such as Ushahidi (Morrow, Mock, Papendieck, & Kocmich, 2011) and Micromappers (micromappers.wordpress.com) aggregate these reports from social media and SMS, providing real-time updates from across a disaster zone before FRs even arrive. First-hand reports by members of the public have therefore become a key source of information during a disaster, in addition to reports from FRs (Meier, 2014; Crowley & Chan, 2010).

The idea of gathering information from members of the public (i.e., crowdsourcing) has also been used to extract information from aerial imagery taken by satellites or UAVs (Meier, 2014; Crowley & Chan, 2010). For example, projects such as Tomnod (www.tomnod.com) or Collabmap (Ramchurn, Huynh, Venanzi, & Shi, 2013) provide a web interface that allows volunteers around the world to mark buildings, assess damage, and identify other features of interest to first responders. Large crowds can process images very quickly (within minutes), covering a wide area and providing data that complements reports from people on the ground (Meier, 2014).

To date, however, such tools do not allow FRs to comprehend the levels of reliability of, and correlations between, crowdsourced reports. This prevents them from trusting the reports and formulating plans that are coherent with the situation on the ground (e.g., multiple reports of riots in areas close to each other could be the same riot but could be seen as different events requiring more resources than really needed). In contrast, in our work we extract the spatial distribution of the disaster from the crowdsourced reports and this allows us to correlate closely related events, thus reducing the amount of information that responders have to assimilate and act upon. To this end, we employ machine learning techniques to extract the sentiment of reports and interpolate between them (see Section 4 for more details) and then visually represent the information extracted for FRs to assimilate. Such information can then be used by the responders to send out UAVs to verify the crowdsourced reports and confirm key targets to send their teams to.

## 2.2 Supervisory Control for Unmanned Systems

Supervisory control interfaces for autonomous systems has been an active area of research in the Human Factors (HF), Human-Robot Interaction (HRI), and Human-Computer Interaction (HCI) domains. A key issue in this area is that plans computed by autonomous systems are typically brittle as they strictly conform to initially set design decisions and ignore the contextual decisions that humans need to make (e.g., weather conditions or threats that may lead to UAVs dropping out or the changing priorities of the mission) (Silverman, 1992; Smith, McCoy, & Layton, 1997). For example, Miller and Parasuraman (2007) developed a ‘Playbook’ of tasks for automated agents to perform when faced with certain situations (upon request from human controllers). Moreover, Lewis, Sycara, and Scerri (2009) developed interfaces to help an operator interact with large numbers of UAVs (hundreds). Bertuccelli, Beckers, and Cummings (2010) instead developed operator models for UAV control and, under simulations, study the performance of their ‘human-in-the-loop’ algorithms whereby operators are unreliable detectors and the algorithm may not perform well in search tasks. While all these approaches relate to our context, they do not specifically study how UAVs employing decentralised coordination schemes (that allow the UAVs to autonomously allocate tasks to each other) can be managed to achieve the objectives of the FRs.

Closer to our work, Cummings et al. (2010) evaluate a mixed-initiative system with a *single* pilot and with an auction-based task allocation scheme. However, they focus on how *often* an operator should be asked to re-plan, and through a set of lab studies, show that operator performance drops with too many frequent re-planning requests. In contrast, our approach considers re-planning as a human-driven exercise and lets the operator decide on the level of autonomy given to agents, from waypoint specification to task allocation. This allows humans to always feel in control of the system and let them specify new objectives on the fly.

In a similar vein, Goodrich, McLain, Anderson, Sun, and Crandall (2007) study how different levels of autonomy (as per Sheridan & Verplank, 1978) given to teams of agents can impact on performance and workload. Specifically, they show that reliance on team autonomy (i.e., a team allocates tasks amongst its members independently of human control) results in neglect from the operator though it reduces workload. Hence, they suggest there should be shifts between different levels of autonomy as per the requirements of the tasks. In line with Jennings et al. (2014), we capture these shifts in autonomy levels through the notion of flexible autonomy in our work and further demonstrate how this notion can be extended to planning support for the deployment of human responders on the ground.

## 2.3 Human-Agent Collaboration for Rescue Mission Planning

Although there is much literature dealing with planning support, task assignment, and human-agent collaboration, very few real world studies of how human emergency response teams actually handle agent support have been carried out. For example, Wilkins, Lee, and Berry (2003) propose a framework for humans to monitor large teams of agents and robots, but they ignore the interactional challenges when such teams need to work in different interactional arrangements (e.g., agent as commander or team member). The most relevant work, by Scerri et al. (2003) and Schurr et al. (2005), focuses on humans acting as peers to agents in computational simulations rather than real-world deployments in the field. In addition, many multi-agent coordination algorithms have the potential to be applied to support task assignment of responder teams. Indeed, work by Tambe et al. (Tambe, 2011; Delle Fave, Shieh, Jain, Jiang, Rosoff, Tambe, & Sullivan, 2014) has shown how humans may be able to implement plans computed as solutions to a Stackelberg game. While their solutions have been deployed with various human teams (e.g., guards at LAX airport or security teams in the Boston/New York/LA harbours), they do not consider how such plans can be generated in real-time in collaboration with humans (i.e., taking into account human input dynamically) nor do they study how humans react to plans suggested by planning agents. However, before we use those algorithms to build agent-based planning support systems, there is a need to understand how humans and agents can effectively collaborate. Controlled experiments designed by the Human Factors community have sought to identify key aspects of human-agent collaboration (Bradshaw, Feltovich, & Johnson, 2011; Nancy, Cooke & Pederson Harry, 2006; Sukthankar, Sycara, Giampapa, & Burnett, 2009; Wagner, Phelps, Guralnik, & VanRiper, 2004), propose transfer-of-control policies to shift control between humans and agents (Scerri, Tambe, & Pynadath, 2002), and evaluate strategies of agent support for human teams (Lenox, Payne, Hahn, Lewis, & Sycara, 2000).

In our previous work (Ramchurn, Wu, Jiang, Fischer, Reece, Roberts, Rodden, Greenhalgh, & Jennings, 2015), we built upon such results and developed an initial prototype (Ramchurn, Wu, Jiang, Fischer, Reece, Greenhalgh, Rodden, Jennings, & Roberts, 2014) of a human-agent collaboration interface for FRs. In that prototype, a human commander worked with an agent-based planner (using decision-theoretic multi-agent planning algorithms) to deliver instructions to players of a mixed-reality disaster response game. The *interactional arrangement* between the human commander and the agent-based planner was such that the agent took control of which instructions to send to FRs on the ground. Results of field trials of the system showed that FRs on the ground were not always compliant with instructions sent to them as the agent was not aware of their level of tiredness, their preferences for tasks, or their preference to work with specific team-mates. In contrast, however, in this paper, we develop a new version of the planning interface that provides better control to the human commander and people on the ground. As such, the interface allows for a less rigid collaboration model, in the use of the autonomous planning agent to suggest plans that can be refined by the humans. We further evaluate this interface with both experts and non-experts in field trials to determine its effectiveness.

### 2.4 Multi-agent Planning under Uncertainty

The problem of planning under uncertainty is often modelled by some form of Markov Decision Process (MDP) (Bellman, 1957; Howard, 1960; Puterman, 1994; Nair & Tambe, 2005). In the presence of multiple agents, the MDP model has been extended to Multi-agent MDPs (MMDPs) (Boutilier, 1996) with a joint action space that represents the range of possible moves that agents can make in their environment. In the past decades, many algorithms[2] have been proposed to solve large MDPs such as RTDP (Barto, Bradtke, & Singh, 1995), LAO* (Hansen & Zilberstein, 2001), and UCT (Kocsis & Szepesvári, 2006) and its multi-agent extensions (Wu, Zilberstein, & Chen, 2010). However, they all assume that the agents completely follow the policy during execution time. Instead, as mentioned earlier, in previous field trials (as in Ramchurn et al., 2015), it was found that the actors in a disaster response scenario may not find plans suggested to them acceptable, and as a consequence, *reject* these plans until an acceptable alternative is presented to them. With agents rejecting plans, plan repair (Fox, Gerevini, Long, & Serina, 2006) or plan strengthening (Hiatt, Zimmermanz, Smithz, & Simmons, 2009) may be useful for deterministic domains but it is very difficult for MDPs because the best action for one state may depend on the actions selected in all the other states. Given this, we pre-plan for the rejections, similar to fault-tolerant planning (Dubash, ling Yen, & Bastani, 1992; Meuleau & Smith, 2003; Jensen, Veloso, & Bryant, 2004; Pineda, Lu, Zil-

[2] See a recent survey article (Mausam & Kolobov, 2012) for more details.

berstein, & Goldman, 2013). Now, the key challenges in our problem are *how to define the rejection model for a team of agents* and how to handle the huge augmented state space given rejections.

Among the early efforts in the DARPA Coordinators program, the work by Musliner, Durfee, Wu, Dolgov, Goldman, and Boddy (2006) is most related to ours, as they use MMDPs to compute a joint policy for human participants. However, they require the transition probabilities to be fully specified (they use the Bellman backup for *informed unrolling*), which is difficult for problems like ours that involve a complex physical process (e.g., spreading of fires or other hazards in the disaster space). Furthermore, they do not model people's preferences and therefore do not explicitly handle rejections. This is crucial especially when some tasks may be harmful to the participants (e.g., going close to a fire or a riot).

### 2.5 Information and Decision Accountability

A key consideration that is often ignored by the various systems described above for disaster response is the need to ensure that the system is accountable (Weitzner, Abelson, Berners-Lee, Feigenbaum, Hendler, & Sussman, 2008). This means that all information generated by the various actors in the system, as well as the decisions they make, needs to be tracked and verified; otherwise, FRs could be acting on erroneous or outdated information. For example, in the Ushahidi platform, volunteers are employed to verify that reports coming from crowds are correct before being put on a map. However, such a manual approach to tracking and verification can be, at times, cumbersome. In fact, a recent report on the Haiti earthquake response efforts unearthed significant issues with information flows, placing significant emphasis on the development of tools that can help reduce ambiguity in the information gathered from multiple sources during the response effort (Altay & Labonte, 2014).

In particular, from requirements gathered during our interactions with emergency responders it was found that it is important to know the source of reports, how such reports have been turned into decisions (to send out UAVs or personnel on the ground), and whether they are complemented by other reports from other, potentially more or less trusted sources. With similar requirements, the database research community has previously investigated automated ways to determine *why*-, *where*-, and *how*-provenance of database query results (Cheney, Chiticariu, & Tan, 2009), e.g. "why was a record part of the result", "where in the input database did this value come from", and "how the result was derived". The challenge is that tracking such dependencies cannot easily be done manually as the number of actors and information sources grows during a major response effort. To this end, Naja, Moreau, and Rogers (2010) attempted to model the provenance of decision making in emergency response situations with the aim of supporting and explaining the rationale of decisions made by software agents in the RoboCup Rescue Simulation. Based on the PROV standards (Groth & Moreau, 2013), we go a step further by deploying a provenance tracking infrastructure to capture provenance of information and decisions in HAC-ER across its various distributed components. We demonstrate how to exploit the provenance information in HAC-ER to support *in-operation* decision making. By actively monitoring the dependencies between information sources and actors (agents, humans, and UAVs) during an operation, our provenance-tracking agent is able to detect and alert emergency responders whenever new reports may impact their performance on the ground. In addition, to ensure accountability for the whole system, such information is also useful for posthoc provenance analysis to identify trends and patterns of behaviours as previously done by Huynh, Ebden, Venanzi, Ramchurn, Roberts, and Moreau (2013) and Moreau (2015).

## 3. Decision Making in Disaster Response

Emergency response agencies are typically hierarchical, military-style organisations that employ the OODA framework[3] (Grant, 2005; Grant & Kooter, 2005). A command and control structure is established whereby decision making is divided into strategic, tactical, and operational levels. The teams responsible for each are sometimes referred to as Gold, Silver, and Bronze respectively.[4] At the strategic (Gold) level, decision makers from all major response agencies involved decide on the main objectives of the response effort. At the tactical level, based on the specified objectives, the Silver command team decides on the allocation of resources and tasks to be carried out, while at the operational level, Bronze FRs, on the ground, determine the logistics required to carry out those tasks. Information gathered from the ground is also passed back up from Bronze, through Silver, to Gold.

In this paper, we focus on the most difficult coordination and information gathering challenges faced by the Silver and Bronze levels of these organisations. The Silver commanders need to gain *situational awareness*, that is, gather information from the disaster scene, ensure this information is reliable, and then efficiently schedule resources and tasks on the ground to meet their objectives. Situational awareness may be gathered using a combination of: (i) crowdsourced reports from members of the public, (ii) deployments of UAVs in collaboration with Bronze FRs to collect aerial imagery and locate key targets and (iii) deployments of FRs to gather first-hand information, while carrying out response efforts in collaboration with other agencies and citizens.

These different information sources (e.g., sensors, phones, or volunteers) come with different levels of reliability and at different costs. For example, gathering data from online crowdsourcing platforms such as Twitter (www.twitter.com) or Ushahidi (www.ushahidi.com) requires relatively cheap web technology, but reports from such platforms can be posted by unverified sources. In fact, as discussed in our design workshops, emergency responders tend to prefer information that they gather first-hand as this has a much higher level of trust. This can either be done through the use of UAVs (with onboard cameras and sensors) or by deploying FRs on the ground. Using UAVs avoids putting humans in harm's way but also poses a challenge in coordinating them and processing the information coming back from them in real-time (which agents may be useful for). Hence, once a general picture of the disaster has been constructed from crowdsourced and UAV-generated information, FRs can be deployed in safer ways to verify and act on the information gathered. Figure 2 (left box) describes these two steps as part of an OODA loop, where the information gathered from the crowd (Observe) is used to decide on a plan for the UAV deployment (Orientate/Decide), which is then is carried out (Act). Similarly, another loop involves orienting to information from UAVs in order to create targets for FRs, before allocating teams of FRs to perform tasks on the ground (Figure 2 (right box)),

To support such decision-making processes, we designed HAC-ER, which is composed of a number of agent-based and human roles. Figure 3 describes the system architecture of HAC-ER. In more detail, we first develop a machine learning agent to annotate crowdsourced reports and generate heatmaps for Silver commanders to visualise and identify key priority areas (see Section 4). We then show how information from heatmaps can be used in determining UAV deployment plans that are generated using a decentralised coordination algorithm (see Section 5).

[3] OODA stands for Observe-Orientate-Decide-Act. It is a well established information gathering and decision making process for deployments in dynamic environments.

[4] Variants of this organisational structure do exist but we find that the Gold, Silver, Bronze model is the most prevalent from our interactions with emergency responders.

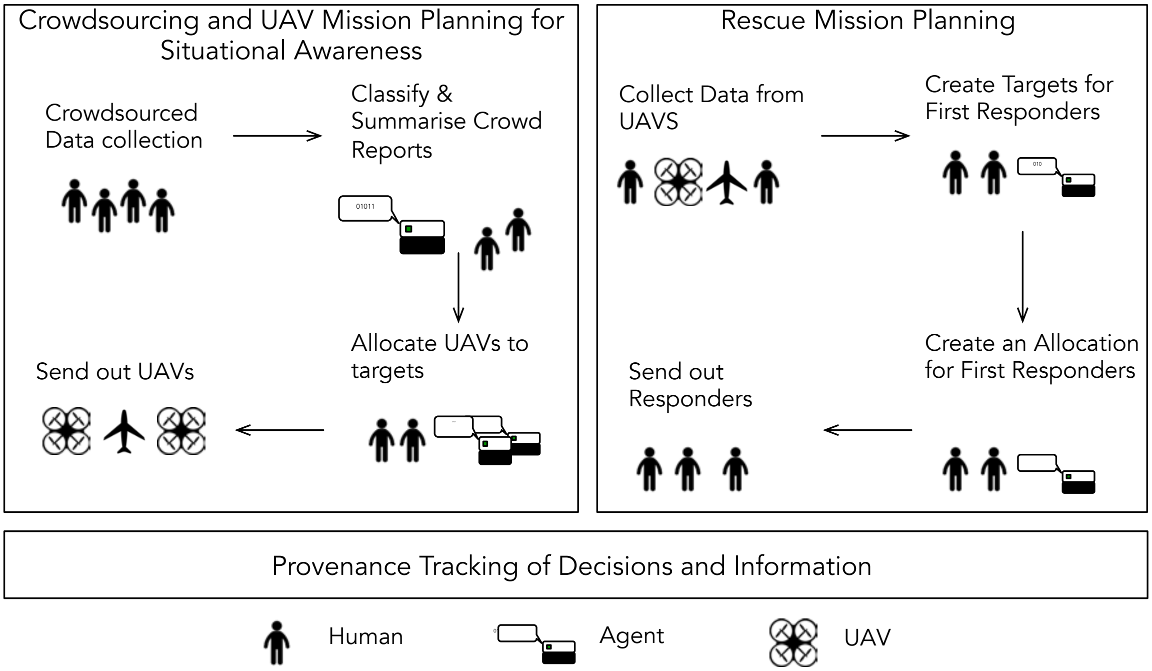

Crowdsourcing and UAV Mission Planning for Situational Awareness
Crowdsourced Data collection
Classify & Summarise Crowd Reports
01011
Allocate UAVs to targets
Send out UAVs
Rescue Mission Planning
Collect Data from UAVS
Create Targets for First Responders
010
Create an Allocation for First Responders
Send out Responders
Provenance Tracking of Decisions and Information
Human
Agent
UAV

Figure 2: The information gathering and decision making processes in HAC-ER. The left box describes the information gathering OODA process while the right box describes the task allocation process. They are underpinned by the provenance tracking infrastructure to ensure that the whole system is accountable.

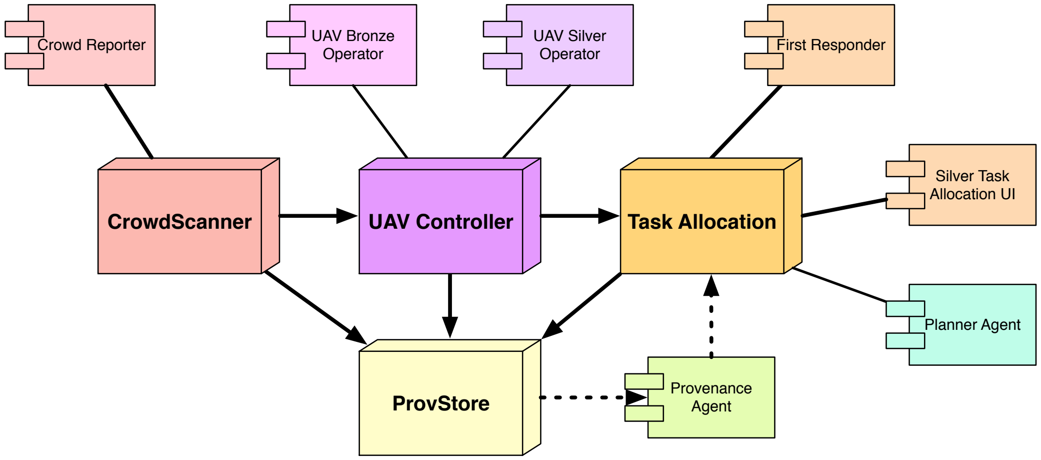


Figure 3: The information flows between HAC-ER's components.

During UAV missions, Silver operators at headquarters will typically monitor the video feeds coming back from the UAVs, while Bronze operators will supervise individual UAVs, and, at times, teleoperate them to gather more detailed information. To support such work we developed a mixed-initiative UAV controller that embeds decentralised coordination algorithms and a number of user interfaces that allow operators to control the autonomy levels at which the UAVs operate.

As targets on the ground (e.g., casualties, collapsed buildings, fuel sources) are identified through this process, these targets are used by Silver commanders to allocate tasks to FRs. To help make these decisions, we developed interfaces for mixed-initiative task allocation (provided as a free tool at www.hacplanning.com), whereby human commanders interact with a planning agent running coordination algorithms that exploit sensor data. Through this interaction, the planning agent can compute plans that are efficient and *acceptable*, i.e. satisfy human preferences.

In general, a large amount of information is generated by various actors (humans, software agents, sensors, and UAVs) in a disaster response operation and a number of key decisions are made based on such information. Hence, a major contribution of this paper is the method by which provenance of such information is tracked and used across the HAC-ER components to improve the decision-making process, providing accountability and ensuring dependencies between information and decisions are continuously recorded. This tracking system (composed of the ProvStore and the Provenance Agent) underlies all the decision-making processes in our disaster management system.

In this section, we have described how different components of HAC-ER fit into the organisation of emergency response agencies during major disasters. A short video describing how the system works is provided at `http://goo.gl/CVSCxa`. The following sections elaborate on each component of the HAC-ER system, namely a HAC-based crowdsourcing platform called the CrowdScanner, the Mixed-Initiative UAV Controller, the Mixed-Initiative Task Allocation System, and the Provenance Agent.

## 4. CrowdScanner: Generating Situation Awareness Heatmaps from Crowd Reports

As a disaster unfolds, social media, satellites, crowdsourcing platforms and FR teams produce vast quantities of data very rapidly, overwhelming the Silver commanders who require situational awareness to plan operations. Some of the information may be irrelevant, erroneous, out of date, or duplicated (e.g. retweets or news reports based on old information, or multiple news channels using the same information source). Furthermore, different sources provide heterogeneous formats of *unstructured data*, such as text reports in various languages, and annotations on satellite images. Large amounts of unstructured data are difficult for Silver commanders to interpret, or for automated tools to extract trends and summaries. To overcome the resulting information overload, we design a HAC for situational awareness, the CrowdScanner, in which a software agent uses machine learning to fuse heterogeneous reports from humans sensors into a common picture of the disaster, or a *heatmap* of incidents and provides interfaces and controls to human operators to comprehend the information received from different sources. Our approach is able to automatically combine information from both unreliable and trusted sources, filter out erroneous data, fill gaps in the data, and help Silver commanders visualise the relevant information via a series of map overlays on a computer screen (see Figure 8).

The Crowdscanner first turns streams of unstructured data from across a disaster zone, including geo-tagged text reports or images, into structured data with the help of a crowd of non-experts. The crowd answers key questions about each report or image and assigns category labels that capture the relevant topics discussed or the features visible in an image. Each category label can either:

1. Identify a particular problem or type of emergency, such as a request for medical assistance or a collapsed building.

2. Indicate no issues at that location.

3. Mark the report as irrelevant.

The crowd may also correct locations associated with the reports if the places that are mentioned in the text do not correspond with their geo-tags. The labelling process can be made more efficient by using the BCCWords algorithm to combine labels from different members of the crowd (Simpson et al., 2015). BCCWords adaptively learns a language model from the crowdsourced labels, which can assist the crowd by reducing their workload and filtering messages automatically.[5] The crowdsourced structured data is then passed to a machine learning algorithm that interprets the data to build a statistical model of the disaster area. The following sections describe the machine learning approach, its application to the Haiti earthquake scenario, its evaluation and finally the visualisation of a heatmap as a tool for situation awareness.

### 4.1 The Bayesian Heatmap

Our approach combines two key machine learning techniques — independent Bayesian classifier combination (IBCC) (Simpson, Roberts, Smith, & Lintott, 2011), and Gaussian Process classification, e.g. the work by Reece and Roberts (2010), into an algorithm that uses the principled Bayesian information framework to achieve the following aims:

- To efficiently combine classifications from a diverse range of sources, remove erroneous information and rectify misclassifications.

- To predict the location of emergencies across an entire disaster area by interpolating between sparse reports, and provide a measure of uncertainty in such predictions. By so doing, it may be possible to decide which areas need urgent verification on the ground or using UAVs.

- To select subsets of reports to pass to the crowd for labelling, thus minimising the work requiring human attention.

The IBCC component of our algorithm combines the crowdsourced category labels at each location to infer the probability of an emergency at those locations. To do this, IBCC learns a *confusion matrix* (Simpson et al., 2011) for each category label from a specific data source, which encodes the correlation between the label and the presence of emergencies. The confusion matrix thereby models the reliability of each information source, as well as the relevance of each type of label to detecting emergencies. For example, tweets from an unknown member of the public and reports published by a trusted NGO would have different confusion matrices, thus accounting for the difference in reliability. Similarly, reports of collapsed buildings may be less strongly correlated with emergencies than requests for medical assistance, therefore IBCC would infer different confusion matrices for each category. By accounting for the varying informativeness of different sources, we can fuse both highly trusted, relevant reports and weaker, error-prone data.

[5] A related approach, AIDR (Imran, Castillo, Lucas, Meier, & Vieweg, 2014), facilitates the crowdsourcing task by learning to filter and tag messages automatically given context-specific training data. As labels are received from the crowd, they can be passed through the BCCWords algorithm or AIDR to produce a set of geo-tagged, categorised reports.

IBCC does not require training with ground truth labels, but is instead an unsupervised approach that fits a model given only crowdsourced data and prior distributions over the confusion matrices and density of emergencies in the disaster zone, $\kappa$. This approach uses a computationally-efficient variational Bayes inference algorithm, which is described in detail by Simpson, Roberts, Psorakis, and Smith (2013). As data is assimilated from the crowd, we update IBCC by running the variational inference algorithm, causing uncertainty to decrease in our posterior distribution over the confusion matrices and $\kappa$.

Now, we note that disasters can impact neighbouring areas in very similar ways. For instance, an earthquake will affect similar infrastructure in neighbouring locations, and a flood can impact adjacent areas in similar ways. Hence, we extend the standard IBCC model to accommodate this insight into a novel algorithm that can make predictions at locations where reports are missing. Our extended model assumes that the *emergency occurrence density*, $\kappa$, varies reasonably smoothly from location to location. To model this, we first assume that a latent function $f$ is drawn from a Gaussian process (GP), which is a distribution over smooth functions defined over the entire spatial disaster zone. At each location with coordinates $(x, y)$, function values $f(x, y)$ are mapped through a sigmoid function to obtain $\kappa(x, y)$ in the range $[0, 1]$, as detailed by Reece, Roberts, Nicholson, and Lloyd (2011). We choose a low order isotropic, stationary Matérn covariance function to model the emergency occurrence probability over the two-dimensional disaster zone as this does not impose a too stringent smoothness on $\kappa$. The GP is fully integrated within the IBCC framework and consequently, the inference of the combined IBCC/GP model is performed efficiently by variational Bayes. The length scales of the GPs are the most likely values found using the Nelder-Mead algorithm to optimise the variational lower bound. The GP enables our algorithm to aggregate neighbouring reports and interpolate between them to determine the posterior distribution of $\kappa(x, y)$ across the entire disaster space.

### 4.2 Haiti Earthquake Data Analysis

In HAC-ER, we use our method to combine real Ushahidi reports written by people in Haiti after the 2010 earthquake, with simulated FR reports. This showcases the ability to fuse information from both noisy and trusted sources. Each FR is given a separate confusion matrix, and can provide binary reports indicating the presence or absence of an emergency at a particular location. We do not have identifiers for the authors of Ushahidi reports, so these are treated as a single source. However, the Ushahidi reports were annotated with a number of different category labels, so we assign each label a separate confusion matrix.[6] We can thereby account for the relevance of each type of report when predicting emergencies. We set the confusion matrix priors for Ushahidi reports so that labels indicating emergencies or related incidents have a weak correlation with the presence of emergencies. The weakness of the priors encodes our initial uncertainty in these noisy reports, and allows patterns in the data to dominate over the priors. For the FRs, we set stronger priors to reflect our greater prior confidence in the accuracy of their reports.

Since the Ushahidi data set contains only reports of emergencies, and no reports stating that no emergency is taking place, we cannot estimate the length-scale of the Gaussian process (GP) from this dataset. The length-scale controls the strength of correlation between neighbouring points in our heatmap, and therefore determines the shape of the density function we estimate, so a good estimate is important. To learn the length-scale of the GP, we can perform maximum likelihood

[6] The Ushahidi dataset does not contain labels indicating that there was no emergency at a particular location; such reports were collected but not marked in the original Ushahidi project, although this would be useful in future.

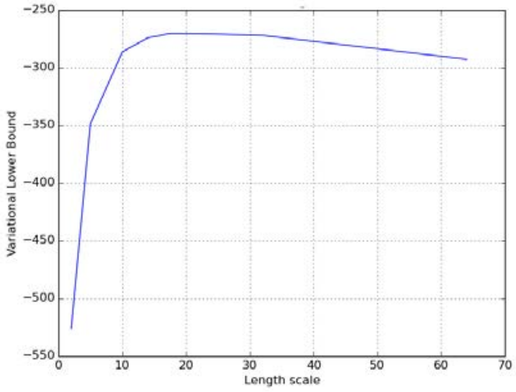


Figure 4: Model evidence variational lower bound for our Bayesian heatmap model learned with different length-scales on Haiti UNOSAT building damage assessment data (Corbane et al., 2011).

optimisation using another dataset with similar characteristics, where the correct classifications are known to a high degree. In our example we use the building damage assessment provided by UNOSAT for the Haiti 2010 earthquake (Corbane, Saito, Dell'Oro, Bjorgo, Gill, Emmanuel Piard, Huyck, Kemper, Lemoine, Spence, et al., 2011) to learn the length-scale for predicting emergencies, as we assume that the length-scales will be approximately the same for both data sets. The spread of emergencies across an earthquake area is likely to be similar to the distribution of building damage due to similar earthquakes, so that the emergency states of locations across an area have a similar degree of correlation to damage states at locations across the area.

In Figure 4 we show the relationship between the length-scale and the model evidence lower bound for the UNOSAT building damages dataset. This plot shows a peak around length-scale=18, indicating the value that is most supported by the building damages data. The model evidence varies by less than 10% with length-scales between 10 and 100, suggesting that the model is reasonably robust to small differences in the length-scale. In a live deployment, data from previous events may be used at first to estimate the length-scale, which will then be updated once sufficient data has been gathered from the current situation, including both positive and negative reports of emergencies. This approach thereby allows the use of background knowledge and data from previous disaster zones to inform the model when reports are sparse. In the next subsection, we show that learning the length-scale from building damage data gives a reasonable model in the emergency prediction scenario.

### 4.3 Evaluation: Prediction with Sparse Reports

A goal of our method is to predict the state at locations where no reports have been received, by interpolating between neighbouring locations. Given an exhaustive dataset with several reports per data point, the standard IBCC model with no interpolation has been shown to give accurate results (Simpson et al., 2013). Therefore, in the absence of ground truth data, we establish a gold-standard test set by training IBCC on 2723 reports, placed into 675 discrete locations on a $100 \times 100$

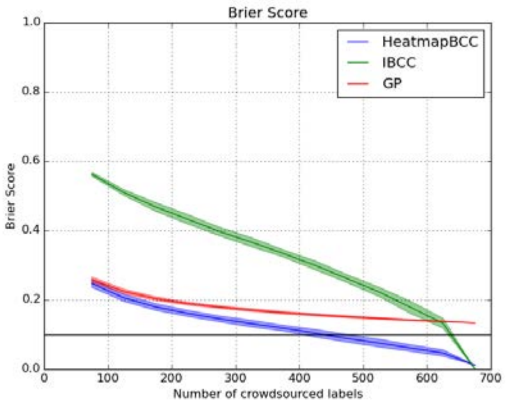


Figure 5: Brier score for Bayesian heatmaps (HeatmapBCC), a Gaussian process density estimator (GP) and independent Bayesian classifier combination (IBCC), evaluated against IBCC predictions with 675 reports on a 100×100 grid. Shaded areas to either side show standard deviation over 20 repeats with different subsets of the data. The black line highlights the point at which Brier score=0.1.

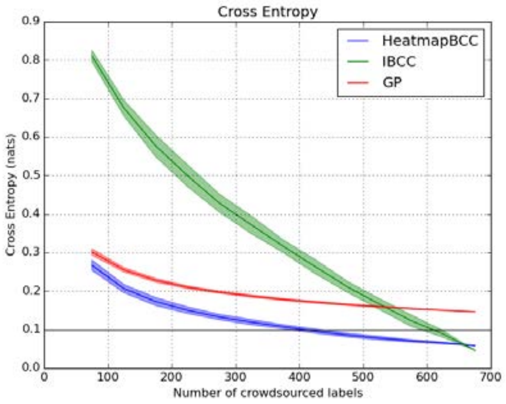
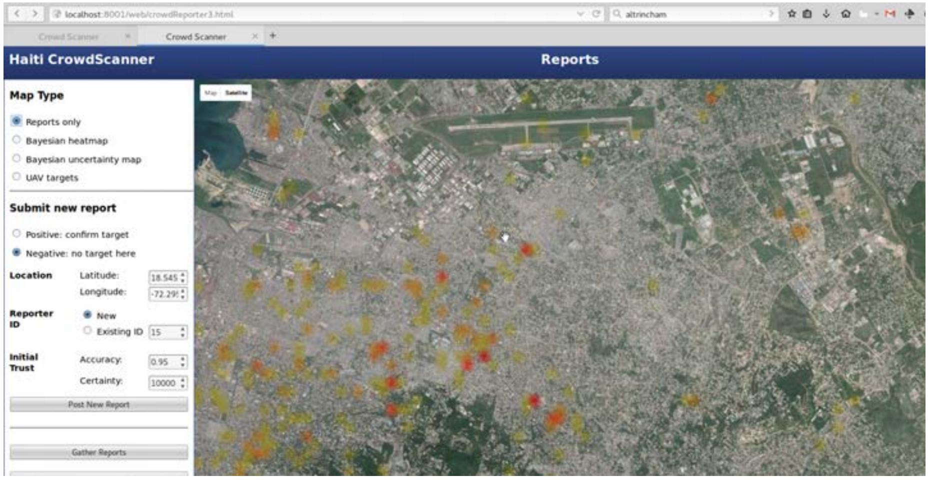


Figure 6: Cross entropy error for Bayesian heatmaps (HeatmapBCC), a Gaussian process density estimator (GP) and independent Bayesian classifier combination (IBCC), evaluated against IBCC predictions with 675 reports on a 100×100 grid. Shaded areas to either side show standard deviation over 20 repeats with different subsets of the data.

grid. Each location has on average approximately 4 reports per grid square. We then evaluate how effective our Bayesian heatmap method is at replicating these results with sparse subsets of the noisy reports.



**Map Type**

● Reports only
○ Bayesian heatmap
○ Bayesian uncertainty map
○ UAV targets

**Submit new report**

○ Positive: confirm target
● Negative: no target here

**Location** Latitude: 18.545 Longitude: -72.29!

**Reporter ID** ● New ○ Existing ID 15

**Initial Trust** Accuracy: 0.95 Certainty: 10000

Figure 7: Locations of Ushahidi reports in a 500×500 grid over Port-au-Prince after the 2010 Earthquake. Red indicates locations with the highest number of reports, while yellow indicates only one report received.

Figure 5 shows the Brier score and Figure 6 shows cross entropy as we increase the subset of reports that we train our model on. The experiment was repeated 20 times using different subsets of the complete Ushahidi dataset. Our Bayesian heatmap method (referred to in Figures 5 and 6 as HeatmapBCC) is compared against a Gaussian process density estimator (Reece et al., 2011) that assumes all reports are reliable, and the IBCC model (Simpson et al., 2011) that performs no interpolation. The Brier score gives the root mean squared error in the probability estimates, while the cross entropy is given by $\int p(x) \log q(x) \mathrm{dx}$, which much more strongly penalises confident misclassifications. The results show that HeatmapBCC has low error rates, even when the reports are sparse, and produces a very similar match to IBCC when the maximum number of labels is provided. The difference between the two methods does not completely vanish because the interpolation performed by HeatmapBCC can still affect the results, even with several reports per grid square, and IBCC is likely to still make some errors. The gold-standard predictions from IBCC also contain some uncertainty, so the cross entropy does not reach zero, even when the model is learned using all crowdsourced labels. The GP method alone is unable to determine the different reliability levels of each report type, so while it is able to interpolate between sparse reports, HeatmapBCC and IBCC detect the reliable data and produce different predictions when more labels are supplied. In summary, HeatmapBCC produces predictions with 439 labels (65%) that are within 0.1 of the gold standard predictions produced using all 675 labels, and reduces cross entropy to 0.1 nats with 400 labels (59%), showing that it is effective at predicting emergency states with reduced numbers of Ushahidi reports.

### 4.4 Heatmap Visualisation Tool

The output of our machine learning method is visualised as a heatmap, providing a tool for situation awareness. We demonstrate the benefits of using the heatmap approach to fusing reports into an overview of the situation as follows. Figure 7 shows the spatial distribution of 765 Ushahidi reports

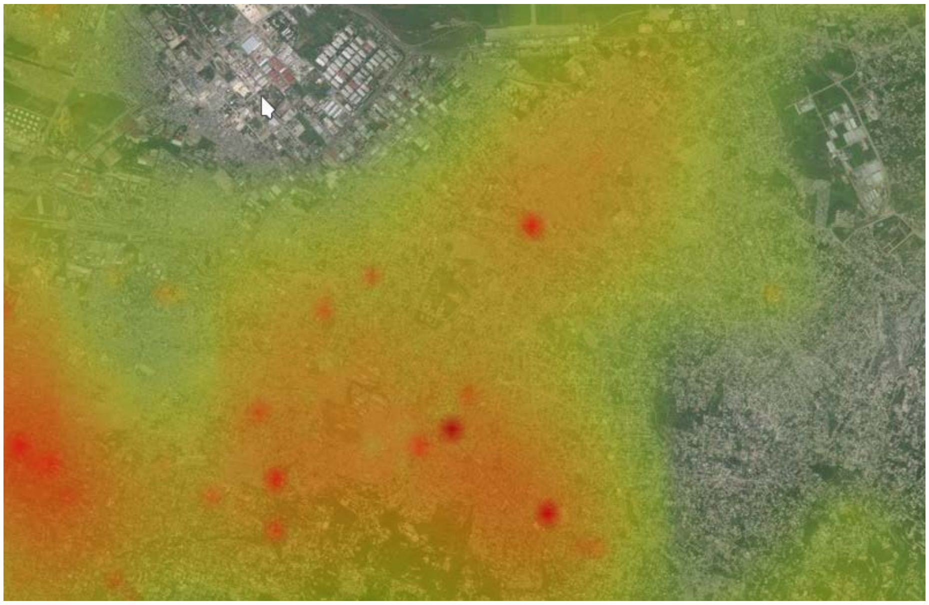
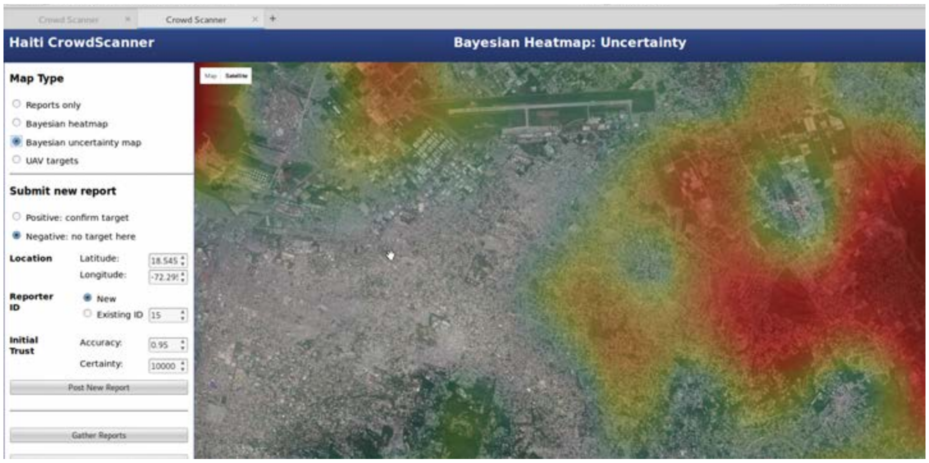


Figure 8: Heatmap user interface for Port-au-Prince after the 2010 Earthquake, inferred from Ushahidi reports only, showing high probability of emergency (dark orange) through to moderate probability of emergencies (yellow). Clear areas have some chance of emergency but the model has not inferred any elevated risk.



**Map Type**

○ Reports only
○ Bayesian heatmap
● Bayesian uncertainty map
○ UAV targets

**Submit new report**

○ Positive: confirm target
● Negative: no target here

**Location** Latitude: 18.545 Longitude: -72.29!

**Reporter ID** ● New ○ Existing ID 15

**Initial Trust** Accuracy: 0.95 Certainty: 10000

Figure 9: Heatmap user interface for Port-au-Prince after the 2010 Earthquake, showing uncertainty in the state of emergencies (dark red), inferred from Ushahidi reports only.

that directly relate to emergencies. The reports are shown as coloured areas on the map before we have applied our machine learning method to interpolate between them. To obtain a better situation awareness overview, we run our machine learning method and then plot the posterior distribution of emergencies over the entire disaster zone as a heatmap, highlighting areas that require emergency aid. Figure 8 shows the heatmap inferred from the 765 Ushahidi emergency reports before any trusted reports are received from FRs. By fully exploiting information between neighbouring reports, our new method reduces uncertainty in the emergency situation at each report location as well as at intermediate points. Consequently, this decreases the number of reports that must be labelled by the crowd. In a live deployment, the machine learning method can be re-run within seconds to incorporate the latest reports in real-time.

Our statistical model can also highlight regions where there is insufficient information to predict the occurrence of emergencies. Areas with high uncertainty also have high variance in the emergency occurrence density, $\kappa$, which can be plotted as a heatmap, as depicted in Figure 9. This uncertainty heatmap informs the Silver commander where the model's predictions are uncertain, and where it may be necessary to gather more information from people on the ground.

In Figure 10, we simulate the effect of sending trusted FRs to different locations in the map to gather further information. The darker red areas correspond to locations where FRs have confirmed emergencies, while blue areas show where the FRs have reported that there were no issues. Figure 11 shows how the trusted reports have decreased our uncertainty over the situation. Our prior confidence in the FRs means that their reports are trusted, so their inclusion means that the model can infer the confusion matrices for the Ushahidi reports more confidently. This leads to reduced uncertainty in areas where the Ushahidi reports and FRs concur, and increased variance in the centre of the heatmap in Figure 11 where the reports conflict.

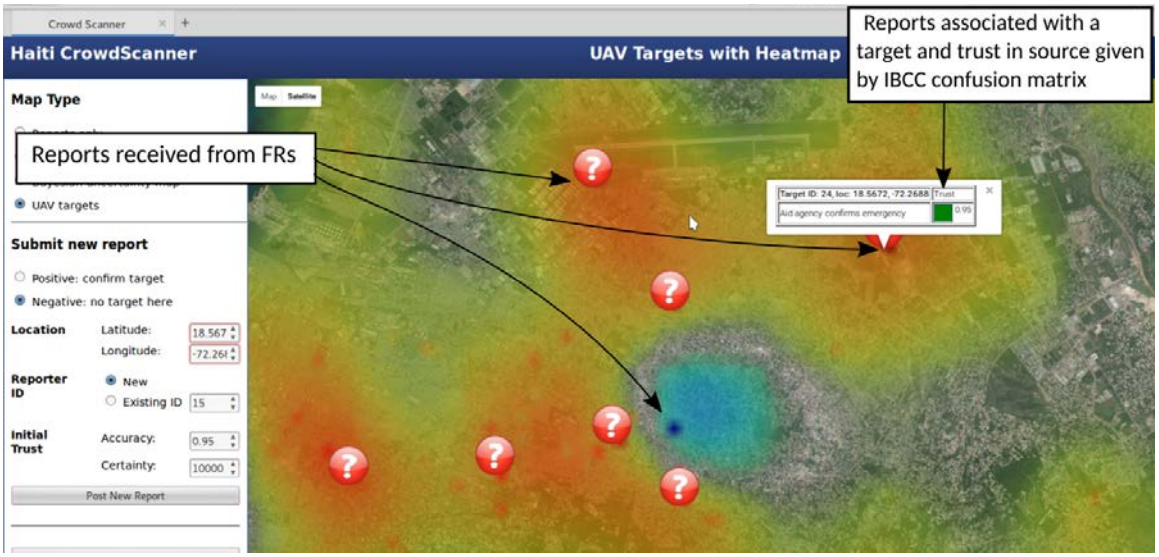
Our approach was deployed in a live situation following the Nepal earthquake in April 2015 (Baraniuk, 2015).[7] In this instance, volunteers used satellite images to identify populated areas where people may be at risk. Each label from a volunteer was, therefore, a category label with a specific location. The model was then used to combine the labels from different members of the crowd, along with structured data extracted from OpenStreetMap. This enabled us to identify regions where OpenStreetMap data was incomplete and predict the situation in those areas.

### 4.5 Using the Bayesian Heatmap

To summarise, the Crowdscanner combines IBCC with GP classification to generate heatmaps that are robust to untrusted reports, and replete when the data is sparse. This section showed how we can use crowdsourcing to interpret unreliable and unstructured reports from members of the public, and integrate these with trusted information sources to provide situation awareness during a disaster relief operation.

An important benefit of our HAC is that we can use its output to guide the process of gathering additional information. Firstly, we prioritise crowd labelling of reports at highly uncertain locations, and secondly, when such reports are not available, we identify locations for further reconnaissance. We also use the heatmaps to automatically identify areas where emergencies are most likely to occur, then use these locations to extract targets for UAVs to gather aerial imagery. The aerial imagery can then be used to either confirm the precise nature of emergencies at that location or to invalidate reports and refocus response efforts where no issues are found. To extract targets, we mark the peaks of the heatmap and order them by the probability of emergency. Then, starting with the peak

[7] See `https://www.newscientist.com/?p=2060153` for a report.



Reports associated with a target and trust in source given by IBCC confusion matrix

Reports received from FRs

● UAV targets

Submit new report

○ Positive: confirm target
● Negative: no target here

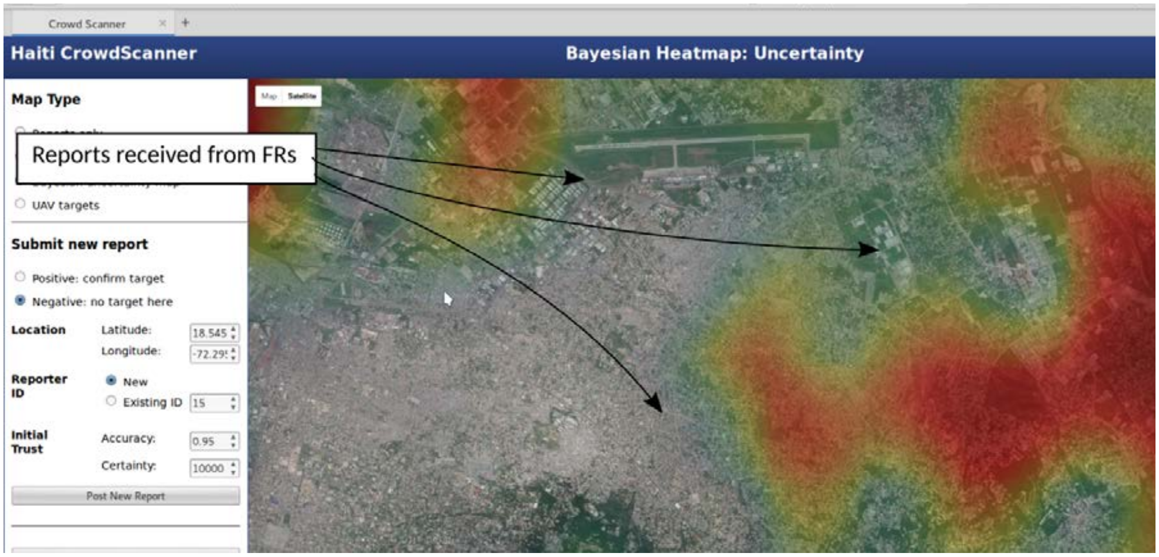
Location Latitude: 18.567 Longitude: -72.26

Reporter ID ● New ○ Existing ID 15

Initial Trust Accuracy: 0.95 Certainty: 10000

| Target ID: 24, loc: 18.5672, -72.2688 | Trust |
|---|---|
| Aid agency confirms emergency | 🟩 0.95 |

Figure 10: Heatmap user interface for Port-au-Prince after the 2010 Earthquake, showing high probability of emergency (dark orange) and an area identified using reports from FRs as having a below average probability of emergency (blue). Clear areas have some chance of emergency but the model has not inferred any elevated risk. Targets for UAVs are marked identified by red '?' icons.



Reports received from FRs

○ UAV targets

Submit new report

○ Positive: confirm target
● Negative: no target here

Location Latitude: 18.545 Longitude: -72.29

Reporter ID ● New ○ Existing ID 15

Initial Trust Accuracy: 0.95 Certainty: 10000

Figure 11: Heatmap user interface for Port-au-Prince after the 2010 Earthquake, showing uncertainty in the state of emergencies (dark red) after including reports from FRs.

with the highest probability, we identify a region around those peaks that is within the flight range of a single UAV, and remove any other peaks in this area from the list. After this process has passed through all the peaks, the remaining list of peaks provides a set of targets, each of which forms the centre an area that can be surveyed by a single UAV. For the Haiti example, these extracted targets are depicted in Figure 10 by red '?' icons. The next section details our HAC approach for deploying UAVs to these targets.

## 5. Mixed-Initiative UAV Controller

The targets suggested by the CrowdScanner come with varying degrees of certainty. Hence, the emergency response team will aim to verify these targets with first-hand knowledge. UAVs are typically used for this purpose to avoid putting personnel in harm's way. However, in most cases, the number of UAV operators available will be limited and the team will aim to send as many UAVs out as possible to gather information as quickly and effectively as possible. Hence, in what follows, we describe HAC-ER's UAV mission planning and command system that provides Silver commanders with supervisory control for allocating multiple UAVs to fly over points of interest in a disaster area so as to verify the potential targets. Moreover, we develop interfaces for low-level UAV teleoperation by individual Bronze operators on the ground to identify specific items of interest from the UAVs' camera feeds.

In more detail, the interaction between Silver and Bronze operators is mediated through voice-based communication as well as interface elements (see Section 5.2). This is an important part of the system as Bronze and Silver operators may have diverging views on how to operate the UAVs. For example, in the dynamic conditions of a disaster scenario, a Bronze operator may ground a UAV if she thinks the weather conditions are inappropriate, or she may take control of a UAV to focus on a particular area to gather imagery. This may disrupt the plan decided by the Silver operators. We address this in Section 5.2.3 by designing the interaction to support dynamic handover of control between Silver and Bronze.

Furthermore, the human team is supported by coordinating agents that are individually in charge of the UAVs. More specifically, agents employ a decentralised coordination algorithm to allocate tasks among themselves. The choice of a decentralised algorithm for this problem is motivated by the fact that the harsh environments within which the UAVs operate may cause communication with the base station to break down. Hence, having a decentralised setup allows us to dynamically reconfigure the way the UAVs optimise the allocation of tasks at any point in time (e.g., run the optimisation on the UAVs if the base station becomes inaccessible). In particular, agents can reconfigure their plans locally and this only needs to involve a subset of the agents rather than all of them.[8]

Thus, Silver operators are able to specify goals for the UAVs, such as fly to a point or scan a region, and the algorithm (distributively run by individual agents) allows the UAVs to decide which of them is best suited for each task. Crucially, our approach to coordination implements the notion of flexible autonomy (Jennings et al., 2014), whereby the agents' plan can be influenced by the human operators. We elaborate on this in the following section and then provide some key insights from trials we ran with practitioners.

[8] The decentralised algorithm we employ also allows the system to scale dynamically, adding in new UAVs and removing others at any point in the peer-to-peer network they may establish. A centralised system would, on the other hand, require registering and configuring any new UAVs with the base station, which can be restrictive.

## 5.1 Flexible Decentralised Coordination

The flexible coordination module continuously monitors the state of the UAVs and tasks defined in the system and dynamically determines a task allocation plan so as to minimise the time that the UAVs take to complete their allocated task(s). We employ the *max-sum* algorithm for decentralised coordination (Rogers, Farinelli, Stranders, & Jennings, 2011; Delle Fave et al., 2012). As shown by Rogers et al. (2011), max-sum provides good approximate solutions to challenging dynamic decentralised optimisation.[9] However, max-sum does not explicitly handle constraints imposed by human operators. For example, if after running max-sum, agent A is tasked to go to point X, agent B to point Y, and agent C to point Z, there is no explicit method for human operators to partially modify the plan such that agent A goes to point Y, and B and C *automatically* re-allocate points Y and Z among themselves in the best way possible. Hence, to cater for such situations, we specify a method to allow the max-sum algorithm to include constraints specified by human operators. In what follows, we first provide a basic description of the max-sum algorithm (see details in Macarthur et al., 2011; Delle Fave et al., 2012) and then explain how it can be modified to take into account human input.

### 5.1.1 THE MAX-SUM ALGORITHM

The max-sum algorithm works by first constructing a factor graph representation of a set of tasks (each representing a point or way-points UAVs are meant to fly to) and the set of agents (each representing a UAV) and then sets a protocol for an exchange of messages between different nodes in the factor-graph. The factor graph is a bipartite graph where vertices represent agents and tasks, and edges the dependencies between them. A detailed description of how such a factor graph is constructed is provided by Rogers et al. (2011). Given this, the max-sum algorithm defines two types of messages that are exchanged between agent (variable) nodes and task (factor) nodes:

- From agent $i$ to task $j$:

$$\forall_{x_i \in D_i}\ q_{i\to j}(x_i) = \alpha_{i\to j} + \sum_{k\in M(i)\backslash j} r_{k\to i}(x_i) \tag{1}$$

  where $x_i$ is a task assignment of agent $i$, $D_i$ is a set of all possible assignments of agent $i$, $M(i)$ denotes the set of indices of the task nodes connected to agent $i$, $r_{k\to i}$ is a message from task $k$ (factor node) to agent $i$ and $\alpha_{i\to j}$ is a scalar chosen such that $\sum_{x_i\in D_i} q_{i\to j}(x_i) = 0$.

- From task $j$ to agent $i$:

$$\forall_{x_i \in D_i}\ r_{j\to i}(x_i) = \max_{\mathbf{x_j}\backslash x_i} \left[ U_j(\mathbf{x_j}) + \sum_{k\in N(j)\backslash i} q_{k\to j}(x_k) \right] \tag{2}$$

  where $U_j \in \Re$ is the utility function of task $j$, $N(j)$ denotes the set of indices of the agent nodes connected to task $j$, and $\mathbf{x_j}$ is a vector of task assignments $\langle x_{j1}, \cdots, x_{jk}\rangle$ for the agents that are relevant to task $j$. The utility function in our scenario includes the travel time to a task, the priority level of the task, and the suitability of the agent(s) for the task (e.g., different agents may have different affinities for a task given its on-board sensors). A task may take more than one agent.

[9] Note that max-sum is by no means the only algorithm we could use here. Other decentralised coordination algorithms could also be used here, e.g. ADOPT or BnB-ADOPT (Gutierrez, Meseguer, & Yeoh, 2011), as we only adapt the graph over which they exchange messages to compute a solution.

Notice that both $q_{i\to j}(x_i)$ and $r_{j\to i}(x_i)$ are scalar functions of assignment $x_i \in D_i$. The largest calculation that any agent performs, as shown in Equation 2, is exponential only in the number of its neighbours, which is typically much less than the total number of agents. Thus, the max-sum algorithm can scale to relatively large problems with many agents depending on their interaction structures.[10]

For an acyclic factor graph (which can be constructed using techniques defined in Rogers et al., 2011), these messages represent the maximum aggregated value for each assignment $x_i \in D_i$ of agent $i$ over the respective components of the graph formed by removing the dependency between task $j$ and agent $i$. Thus, the marginal function of each assignment $x_i$ is calculated by:

$$z_i(x_i) = \sum_{j\in M(i)} r_{j\to i}(x_i) = \max_{\mathbf{x}\setminus x_i} \sum_{j=1}^{m} U_j(\mathbf{x_j}) \tag{3}$$

after which the assignment of $x_i$ can be selected by:

$$x_i^* = \arg\max_{x_i\in D_i} z_i(x_i) \tag{4}$$

In our system, Equation 1 is computed by the UAV agents while the more computationally-intensive calculations (i.e., Equation 2) are processed by a base station. This is critical because the computational resources of the UAVs are usually very limited. In each step, the base station submits tasks to the UAVs so they know their possible assignments and who their neighbours are (i.e., the task nodes) in the factor graph. Then, the UAVs and the base station exchange messages as detailed by Equations 1 and 2. Finally, the best assignment is selected autonomously by each UAV using Equation 4.

#### 5.1.2 INCORPORATING HUMAN INPUT

Given a set of tasks and agents, max-sum computes shortest paths for each UAV to the tasks, as well as taking into account the priorities set for each task and the type of UAV required. However, this assignment may not be deemed appropriate by the human operators as it may conflict with their priorities and expectations about flight paths. For example, a UAV may be allocated by max-sum to fly from its position in the East to a task in the West but the human operators may, instead, prefer a UAV to fly from the South to the same task in order to provide imagery over the area covered by that path, which may be more important than the lateral traversal from East to West. Moreover, if the priority levels are not granular enough, max-sum will not differentiate between tasks of very similar priority levels.

Given a plan computed by max-sum, through our planner interfaces (see Section 5.2.2), users can specify *manual* allocations of UAVs to tasks. These manual allocations specify a task-agent pair $(i, j)$. Given this, for an agent $i$, we then define $D_i = \{j\}$. This effectively results in the deletion of all edges in the factor graph that connect the agent node $i$ with other task nodes apart from that of $j$. This, in turn, forces max-sum to only allocate agent $i$ to task $j$ (as per Equations (1) and (2)), or if two (or more) agents are required by task $j$, another agent will be chosen based on this restriction.

[10] The number of neighbours considered in the computation can be restricted based on their distance to the tasks or based on how their capabilities (sensors or flight mode) match the tasks. In this particular work, we used both to limit the number of variable nodes (agents) connected to factor nodes (tasks).

## 5.2 Interaction Design

We designed a number of fully functional, web-based user interfaces to allow a human-agent team to coordinate the UAVs. Through these interfaces, Silver operators visualise the plans suggested by max-sum and modify these plans as required. We also provide an interface for the Bronze operator to teleoperate a selected UAV. A video of the use of the system is available at `http://goo.gl/DmrJXU`. We next detail the interactions within each view.

### 5.2.1 CAMERA VIEW

The camera view provides multiple live video streams from the UAVs (an MPEG stream is available from typical UAV camera modules). In the simulation of the system, we employed Google Maps (maps.google.com) *aerial* view (see left of Figure 12 for feeds from six UAVs). The images displayed are taken at real GPS locations of the UAVs in the disaster area. Targets, as identified by the CrowdScanner, are positioned at specific points in the area to be considered, and are displayed on the aerial view whenever the UAV flies over them. The user can then click on the map and create an annotation with the matching description as perceived by the Silver or Bronze operator. Once a target has been identified, an icon describing the target is then displayed across all views to ensure immediate situational awareness across the team.

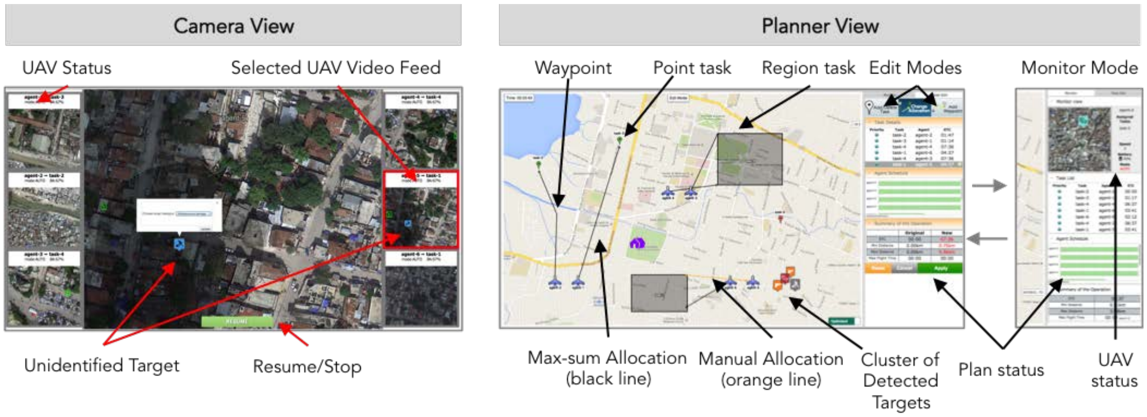


Figure 12: Silver operators' views with the camera view and the two modes of the planner view.

An important feature of the interface is the flagging system, which Silver operators can use to alert Bronze operators when specific items of interest appear in the camera view. A special (clickable) button on each camera view highlights to the Bronze operator that a specific UAV needs attention (see Section 5.2.3). Moreover, if a Bronze operator takes over control of a particular UAV, this UAV's camera view updates its status to 'tele-operated'. By so doing, we allow Silver and Bronze operators to coordinate their actions.

### 5.2.2 PLANNER VIEW

The planner view is the main tool that provides both monitoring and planning capabilities (see right of Figure 12). Operators can choose to create two types of tasks for the UAVs; *point* tasks with waypoints, which require UAVs to fly to specific points in the space and scan the area along the way, and *region* tasks that define areas that teams of UAVs can self-organise to sweep-scan (i.e., they automatically divide the area between themselves and scan their individual sections). The operator can decide, either using max-sum or manually, which UAV should go to each of these tasks. These capabilities are accessible in two modes through the tabs on the top right, namely ‘Monitor’ and ‘Task Edit’. We describe each of these modes in turn.

**Monitor Mode** This mode shows the current status of the allocation (see the right part of Figure 12). The allocation of UAVs to tasks is represented as lines with arrowheads. Region tasks are marked as grey boxes and point tasks using icons. Paths chosen by the max-sum algorithm are shown in black, while paths chosen manually by the users are shown in orange. Once a region task has been completed, the grey box turns green. A region task is deemed completed when UAVs have covered its area, and a point task is considered completed when the allocated UAV has reached that task and hovered for 5 seconds. Once a point task is completed, the task disappears from the map. The right side of the monitor displays the current allocation of agents to tasks, the expected completion times and the schedule (as a Gantt chart) of the UAVs going to the tasks.

**Task Edit Mode** This mode provides the user with a number of planning options (see right part of Figure 12) through a number of sub-modes. The user can:

1. add/delete tasks (region or point): Users can create two types of tasks: (i) region tasks — this task requires two UAVs to carry out a sweep scan of the area selected by the user, and (ii) point tasks — a point selected on the map.

2. change/adapt the allocation of tasks to agents: the allocation automatically computed by max-sum can be changed by the user by clicking on a UAV and allocating it to another task. Max-sum then adapts its allocation to fit the constraint set by the user (as per Section 5.1). For each allocation, a straight line is drawn from the selected UAV to the selected task (unless waypoints are specified).

3. add waypoints to the paths taken by the UAVs: this applies to paths chosen to point tasks, whereby users can adjust the path taken by a UAV to cover areas in more complex ways than in region tasks.

Once an allocation of UAVs to tasks has been chosen, the user can verify the completion time of the tasks using the sidebar widgets and then decide to execute the plan (i.e., fly the UAVs).

### 5.2.3 BRONZE OPERATOR VIEW

The Bronze operator view is displayed on a tablet interface. In this view (see Figure 13), the Bronze operator can select the specific UAV she may want to teleoperate or supervise more closely. Each UAV’s camera view is accessible under different tabs (left of the screen). Additionally, in this view, we provide a notification mechanism for the Silver commanders to alert the Bronze operators, whereby the tab related to a specific UAV can be made to flash when the Silver commander ‘flags’ the UAV in their screen.

The view also incorporates a simulated joystick that controls the direction and speed of the UAV (pushing further in a given direction speeds up the UAV) and a slider that regulates the altitude of the UAV. The Bronze view is designed to receive a live video feed from any drone that transmits an MPEG video stream.

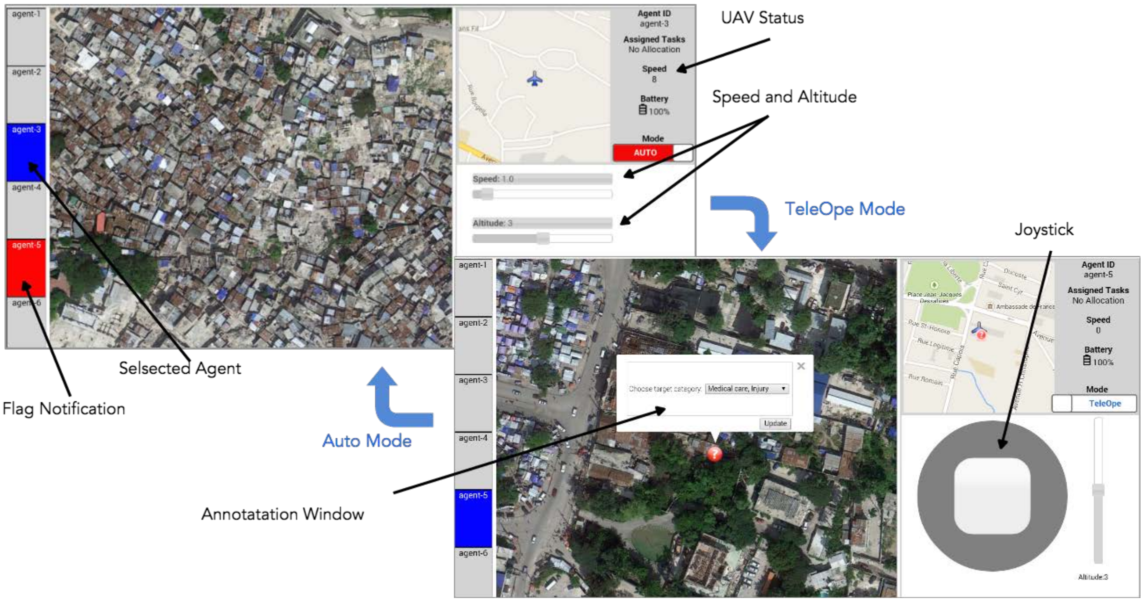


Figure 13: Tablet-based Bronze Operator Views.

### 5.3 Using the Mixed-Initiative Multi-UAV Controller

Our controller allows emergency responders to work with multiple UAVs to carry out missions in reduced operator to vehicle ratios (i.e., more UAVs than operators). Over multiple iterations, various prototypes of the system were tested by emergency responders from Rescue Global and other emergency organisations in an attempt to refine its capabilities and interactions (see Figure 14). In more detail, this involved organising workshops and walking the responders through the following steps:

1. A disaster scenario is presented to the responders. For example, they are told that an earthquake has hit a country and several casualties are stranded across the capital city and a fire has erupted in one area and is about to engulf several inhabited parts.

2. A UAV task allocation problem is specified. This involves a specification of the number, type, and position of all UAVs available. They may also be given a deadline to execute some tasks.

3. As the responders carry out their tasks, we ask them to explain how they are planning to execute their tasks.

4. At the end of the exercise, they are asked what features impeded their performance and what other features they wish they had in order to execute their tasks better. These formed the basis for further improvements to the interface.

As a result of several such exercises (and different scenarios), it was found that the key challenge was to ensure that plans offered by max-sum were understandable and adjustable. Other important features included the ability to specify regions that the UAVs would patrol in case a threat (e.g., fire or radioactive cloud) could reach some areas, and the ability to select UAVs using simple control (e.g., click and drag or using keyboard controls). This led to the design of the planner interface detailed in Section 5.2.2.

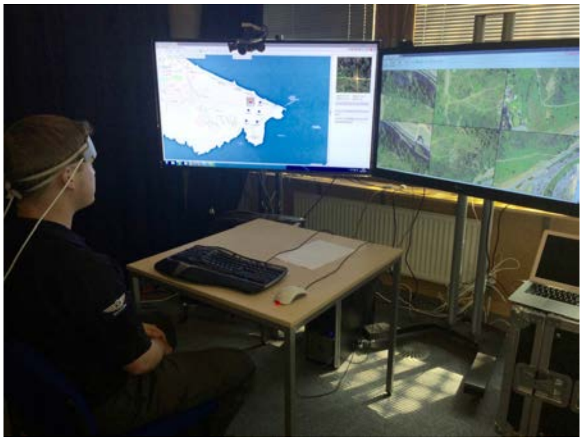


Figure 14: A silver commander from Rescue Global testing an earlier prototype of the multi-UAV controller.

Moreover, to verify the effectiveness of the various interfaces (in particular the Planner and Monitor views), we designed and executed a number of lab studies where we tested how operators would interact with the max-sum algorithm and recover from UAV drop-outs (i.e., UAVs failing or being taken back to base by the Bronze operator). The results of the study, reported by Ramchurn, Fischer, Ikuno, Wu, Flann, and Waldock (2015), showed that the interface enabled operators to outperform a purely manual task allocation, that is the HAC performed better than a purely human-driven planning approach. The HAC approach was particularly useful when drop-outs happened, whereby plans could be reconstructed (using advice from max-sum and confirmation/edits from the Silver operator) rapidly on the fly. However, in some cases, human operators were found to be over reliant on the abilities of the max-sum algorithm, echoing earlier results in this area (Smith et al., 1997).

In the next section, we describe a very similar interactional arrangement of a software agent providing plans to a Silver commander in charge of deploying Bronze FRs to undertake tasks identified and verified by the UAVs. The picture is further complicated by the fact that, as opposed to UAVs that obey all instructions, the Silver commanders have to task FRs on the ground and coordinate them. This raises new challenges for flexible autonomy.

## 6. Mixed-Initiative Task Allocation

Having confirmed the locations of key targets in the disaster area in the Mixed-Initiative UAV controller, we next consider the deployment of FRs on the ground. More specifically, in this section, we describe the system for Silver commanders to compute task allocations for Bronze FRs using the help of a planning agent. In this section, we first provide an overview of the model used by the planner agent, then describe the interaction mechanisms used to allocate tasks to the FRs. Finally, we present the results of field trials of the system and contrast the results with those of an earlier prototype.

### 6.1 The Planner Agent

We developed an algorithm for a planner agent, that computationally models the decision making of FRs in terms of the actions they take and the teams they may want to form to complete their tasks. In contrast to the UAV task allocation problem where operators (Bronze or Silver) control UAVs at will, allocating human FRs requires judging whether (a) they are fit to perform their tasks and (b) whether there are any constraints that prevent them, individually, from doing so. Hence, at any point in time during a mission, the FRs have a choice to perform a number of actions and to each action is associated a probability that they may execute it (e.g., when they are tired, they are more likely to reject tasks or when they are close to a team-mate, they may be more likely to look for a task that requires their capabilities). By modelling FRs' decision-making in this way, we can then employ decision-theoretic techniques to solve the task allocation problem.

In more detail, the algorithm receives GPS locations of targets from the Mixed-Initiative UAV Controller and the location of FRs through their mobile responder tool (see Section 6.2.2). We model the problem of allocating FRs to targets using a *Multi-Agent Markov Decision Process* (MMDP) as it involves multiple actors making decisions collaboratively under uncertainty. In what follows we only describe the MMDP model we use to solve the planning problem as the implementation details are beyond the scope of this paper (see Wu, Ramchurn, Jiang, Fischer, Rodden, & Jennings, 2015 for more details).

Formally, an MMDP is defined as tuple $\langle I, S, \{A_i\}, T, R, s^0, \gamma \rangle$, where: $I = \{1, 2, \ldots, n\}$ is the set of $n$ FRs; $S$ is a set of system states (e.g., where the FRs are positioned, their current task); $A_i$ is the action set of FR $i \in I$; $T : S \times \vec{A} \times S \rightarrow [0, 1]$ is the transition function where $\vec{A} = \times_{i \in I} A_i$ is the set of joint actions; $R : S \times \vec{A} \rightarrow \Re$ is the reward function (e.g., the level of completion of a rescue mission or the time it takes to distribute vital resources); $s^0 \in S$ is the initial state; and $\gamma \in (0, 1]$ is the discount factor. Here, an action $a_i \in A_i$ is what a FR can do in one step in a fixed amount of time so all FRs complete their actions at the same time (as is commonly assumed in other MMDP applications). If some tasks take much longer than others, FRs only need to repeat their actions several times until the task is finished. The outcome of solving an MMDP is a policy $\pi : S \rightarrow \vec{A}$ that maps states to joint actions. Starting in state $s^0$, a joint action $\vec{a}$ is selected based on policy $\pi$. Each agent executes its component $a_i$ of the joint action and the system transitions to next state $s'$ based on the transition function. This process repeats with the new state $s'$. The objective of solving an MMDP is to find a policy that maximises the discounted expected values.

This MMDP can be fed to standard solvers such as UCT (Kocsis & Szepesvári, 2006). However, this will be very inefficient due to the large search space of the model (e.g., in a simulation with 8 FRs and 17 tasks in a 50×55 grid, the total number of possible system states is more than $2 \times 10^{400}$). Hence, we decompose the decision-making process into a hierarchical planning process: at the top level, a task planning algorithm is run for the whole team to assign the best task to each FR given the current state of the world; at the lower level, given a task, a path planning algorithm is run by each FR to find the best path to the task from her current location. Furthermore, since not all states of MMDPs are relevant to the problem, we only need to consider the reachable states given the current state. Hence, we compute the policy online, starting from the current state. This reduces the computation significantly because the number of reachable states is usually much smaller than the overall state space. Next we consider how to allocate tasks and paths to FRs.

We define the team values that reflect the level of performance of FR teams in performing tasks. This is computed from the estimated rewards that the teams obtain for performing the tasks. The expected values after completing the tasks are estimated using Monte-Carlo simulations. Given the team values, we assign a task to each team by solving a *mixed integer linear program* that maximises the overall team performance given the current state, subject to the requirements of each task on the FRs.

In the path planning phase, we compute the best path for a FR to her assigned task. Since there are uncertainties in the environment and the responders' actions, we model this problem as a single-agent MDP that can be solved by *real-time dynamic programming*. By so doing, we assign tasks to FRs such that their long term effects are rewarding while the search space is reduced to a tractable size.

The output of our algorithm is a set of actions that describes which task each FR should undertake and their best paths given the tasks in the current state. This plan can then be provided on demand to Silver commanders. The challenge, however, is that such plans do not factor (i) whether the FRs are tired — if FRs are tired they may not be able to do the task allocated and may prefer to do easier (closer) tasks, and (ii) existing relationships between FRs — this can result in some FRs preferring to work together and leave others out. Crucially, it is not possible for the first agent to model all the aspects of human collaboration and perception which could mean that plans may not make sense in the real world. Against this background, we developed a novel model for agile planning, where plans sent by the planner agent to human FRs *may be rejected.* Specifically, we integrate the notion of plan rejection as one of the possible failures that may happen when actions are performed by the actors in the system. Thus, if the human FRs reject a plan, the system simply transitions to another state and returns a new plan.

We develop a new model for task allocation with rejections, $k$-RMMDPs, that is the first to explicitly capture rejections as part of the planning process. The $k$-RMMDP has the same action space as the original MMDP but has an augmented state space that additionally captures rejection events, a transition function that incorporates a rejection model, and a reward function that captures the cost of rejections. A rejection model specifies the probability of a sub-team of human FRs rejecting a joint action while the other FRs that are not in the sub-team accept it in a given state. Here, we assume that each FR has only a bounded number of chances to reject a plan in a single decision step. In other words, we only allow each FR $k$ rounds in a decision step to reject the plan and request a new plan. The upper limit $k$ is necessary to ensure an agreement can be reached by all the human FRs during a limited amount of time.

It is worth noting that our model is a variation of the MMDP where the rejection model is explicitly represented. In other words, $k$-RMMDP is a special case of MMDPs with the states, the transition function, and the reward function of considering rejections. The rejection model is necessary to specify when a plan may be rejected by some FRs during planning time. Compared to the pure MMDP representation, the main advantage of having such an explicit representation is that we are able to exploit the problem structure and develop planning algorithms to more efficiently solve our problem. Given that $k$-RMMDP is a special case of MMDPs, it can be solved by the UCT method. Although UCT can converge to the optimal solution given a sufficient number of simulations, directly applying UCT to solve k-RMMDPs turns out to be very inefficient (it can require a huge number of simulations) because the state space of k-RMMDPs is very large. Therefore, the planner can easily get trapped in local optima when all the actions are rejected by the human FRs. Hence, we develop a new approach called two-pass planning to solve the k-RMMDP.

We apply a Two-Pass Planning (TPP) process where, in the first pass, we compute the best policy for the underlying MMDP without rejections, and, in the second pass, we handle the rejections using the policy computed by the first pass. By so doing, we decouple the normal states (where humans are allocated tasks) from the rejection states (where they await new plans). Our key observation is that the Q-value function of each state-action pair can be recursively split into two components

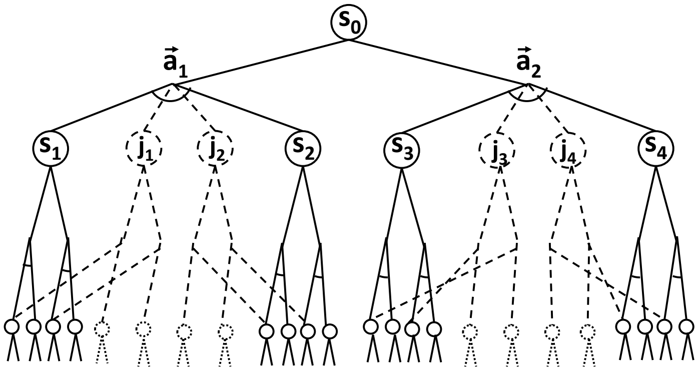


Figure 15: Search tree built by the TPP algorithm. Solid lines are generated in the first pass while the nodes with dotted lines are expanded in the second pass

given the state structure, where the first component specifies the transitions to the normal states and the second component is for the transitions to the rejection states.

Figure 15 graphically illustrates the search tree of our TPP algorithm where the tree nodes with solid lines are generated in the first pass while the nodes with dotted lines are expanded in the second pass. As seen from the figure, the search tree interleaves state nodes with action nodes. In the first pass, the state node $(s_0, 0)$ (the rejections count 0 is omitted in the graph) is expanded with two action nodes $\vec{a}_1$ and $\vec{a}_2$ and then transits to states $(s_1, 0), (s_2, 0)$ and $(s_3, 0), (s_4, 0)$ respectively assuming $\vec{a}_1$ and $\vec{a}_2$ are accepted by all human FRs. This process repeats for the newly expanded state nodes until the maximal search depth is reached. This is equivalent to solving the underlying MMDP with the UCT method.

In the second pass, each action node is expanded with the rejection states considering the case that the action may be rejected by some FRs. For example, two nodes of rejection states $(s_0, j_1), (s_0, j_2)$ (the system state $s_0$ is omitted in the graph) are appended to action node $\vec{a}_1$ depending on who reject $\vec{a}_1$. Then, $(s_0, j_1), (s_0, j_2)$ are expanded with new action nodes. This is equivalent to proposing new actions to the FRs after they reject $\vec{a}_1$. If the new actions are accepted, the state $(s_0, 0)$ transits to new normal states so the sub-tree with dotted lines is linked with some state nodes generated in the first pass (if such nodes do not exist, new nodes are created and expanded as in the first pass). If the new actions are also rejected, new nodes of rejection states are generated and the whole process repeats down to the maximal depth. After that, the values propagate from the leaf nodes up to nodes $(s_0, j_1), (s_0, j_2)$ and then to node $\vec{a}_1$ and node $(s_0, 0)$ with their values being updated accordingly. Finally, the best policy is computed based on the Q-values.

To apply our $k$-RMMDP model and the TPP algorithm to the real-world, the planner agent must get feedback from human FRs and allow them to reject the plan at each decision step. Hence, in the next section we develop methods for interactions with the planner agent, as well as between Silver commanders and Bronze FRs to help them converge on an effective plan.

### 6.2 Interaction Design

Following the Haiti scenario, once the UAVs have identified the targets on the ground (see Section 5), FRs with specific roles have to be allocated to these targets to further investigate, evacuate, rescue or repair, depending on the nature of the target. To exemplify how this system works, we assume there are four different types of FR roles (medics, firefighters, soldiers, and transporters), and four different types of targets (injured personnel, social unrest, infrastructural damage, and water shortage). Mimicking real-world complexity, different targets have different role requirements, e.g., to rescue an 'injured personnel' target, a medic and a transporter are required. For Silver commanders, this creates the aforementioned task allocation problem under time pressure (due to task deadlines). In order to support Silver commanders with their coordination work to allocate tasks to FRs, we developed a set of interactive tools that both integrate the agent-based task allocation and path planning algorithm in the previous section, as well as situation awareness and communication capabilities for Silver commanders and FRs.

Fundamentally, these capabilities are enabled by two applications: (a) a web-based task allocation interface to support Silver coordination of FRs; and (b) a Mobile Responder Tool for FRs to respond to task assignments and messages from Silver commanders, to find each other, and to navigate the environment to evacuate targets. The following sub-sections illustrate the ways in which we implemented a *human-in-the-loop* rationale for intelligent task allocation.

Findings from earlier field trials showed that an effective interaction strategy leaves routine task assignment to the agent, but human input is required to confirm allocations or change these on demand, and, in particular, to deal with contingencies that may arise in the disaster setting (Ramchurn et al., 2015). This mixed-initiative design rationale had also been supported in a workshop with emergency responders from Rescue Global. It was particularly highlighted that such a disaster response system is an opportunity to provide, in real-time, an up-to-date picture of the disaster zone. Crucially, they highlighted the fact that the information generated by FRs on the ground, UAVs, and the CrowdScanner, needed to be tracked and analysed continually to identify potential discrepancies in decision making, and reviewed by HQ (Silver) before any task assignments could be provided on its basis. In what follows, we describe the different elements of the mixed-initiative task allocation system, namely the task allocation interface that employs our planning agent and the mobile responder tool. Building upon this, we detail results of field trials of the system and contrast the results with those from field trials of our earlier prototype.

### 6.2.1 TASK ALLOCATION UI

The web-based task allocation interface for Silver commanders is depicted in Figure 16. The operator uses the real-time map to locate responders and targets 'on the ground', to keep track of tasks and their deadlines, to request assignments from the agent, and to inspect and confirm which of them get sent to the FRs. The task assignments may be edited 'manually', for example, to prioritise a specific target due to type or deadline.

Due to relative complexity of the 'workflow' and the resulting target states, we implemented interactive elements to make the UI effective and guide the attention of the operator as follows:

- **FR feedback**. FRs accept or reject tasks (e.g., due to local knowledge unavailable to the operator). Rejection leads to a red, flashing signal, signalling to the operator that immediate attention is required. The operator may then decide to call upon the planner agent for a new allocation based on certain constraints or manually create an assignment of tasks to FR.

- **Hover-over alignment**. Hovering the cursor over task or FR icons in the task list or in the allocation list highlights the corresponding icon on the map, so as to facilitate aligning the views.

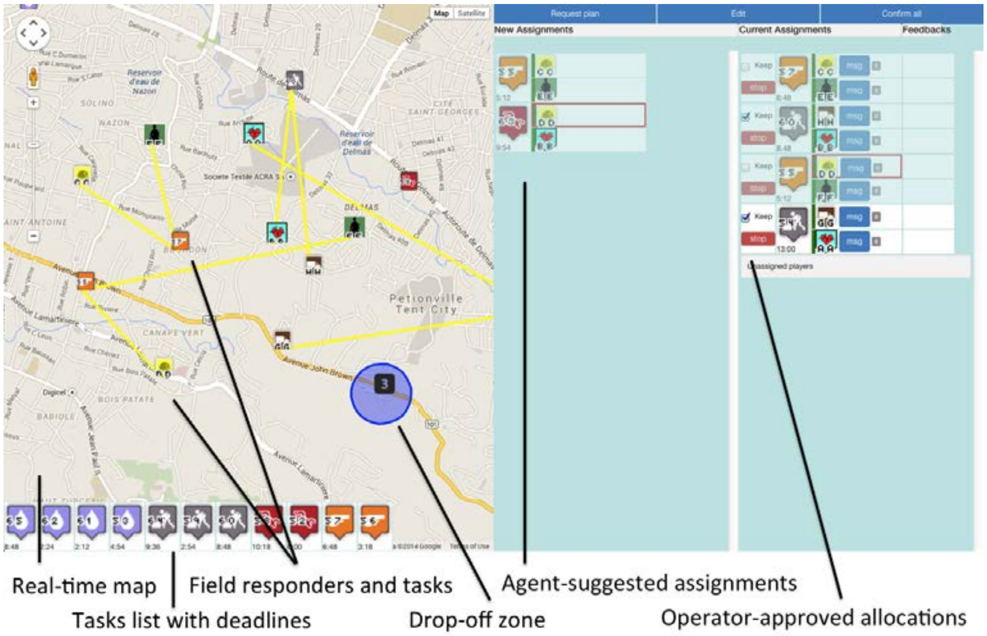

Request plan
Edit
Confirm all
New Assignments
Current Assignments
Feedbacks
Real-time map
Tasks list with deadlines
Field responders and tasks
Drop-off zone
Agent-suggested assignments
Operator-approved allocations

Figure 16: The Silver commanders’ task allocation interface.

- **Drag-and-drop editing**. Silver commanders allocate FRs to tasks by drag-and-drop. Once a task is dropped into the assignment column, the responders’ required roles are visualised to further guide the operator.

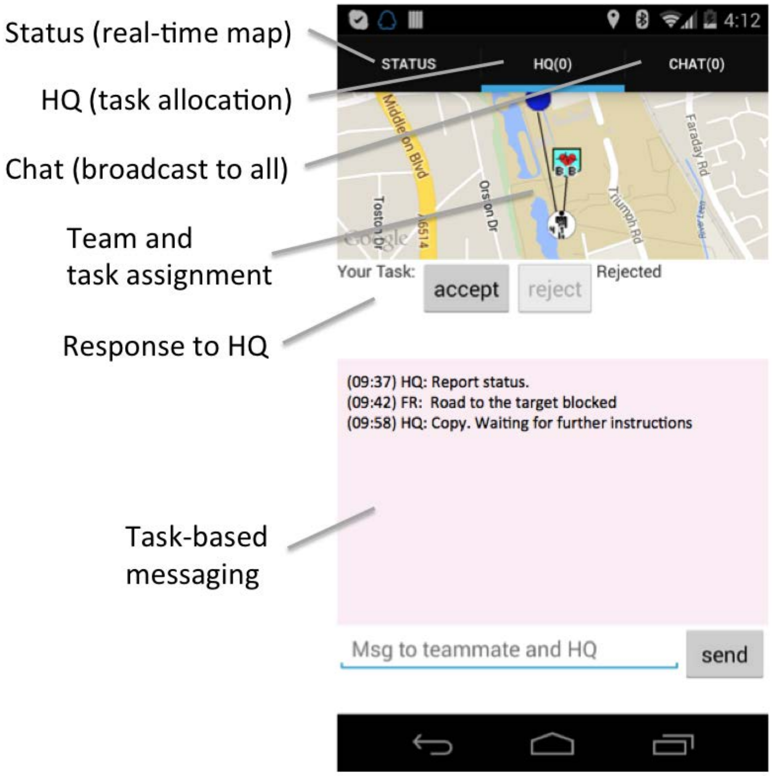
- **Task-based comms channels**. Operators have a channel for each team allocated to a task, to provide task-specific messaging (see Figure 17).

### 6.2.2 MOBILE RESPONDER TOOL

The mobile responder tool is depicted in Figure 17. It provides the Bronze FRs with the same real-time map as the Silver commanders, with some UI controls to help focus on important elements on the limited mobile screen size, such as ‘find me’ and ‘show task’. In addition, it provides task allocation information in a separate tab (shown), which users have to accept. In case they reject the allocation, they are first alerted to the task deadline in a modal (‘are you sure?’ dialogue) so as to discourage rejection.

## 6.3 Evaluation

Here we detail the evaluation of the Mixed-Initiative Task Allocation Interface. We detail such an evaluation of this HAC to clearly exemplify the methodology by which such systems should be evaluated to draw out new insights into how algorithms should be designed (evidenced by the results of the first prototype) and how interactions should be designed for teams of humans and agents to achieve their goals efficiently and effectively. Specifically, we demonstrate an “in the wild” evaluation of the system with humans, performing tasks in the real world, under the guidance of an agent that operates in real-time. This contrasts with previous work that focused on simulations



(09:37) HQ: Report status.
(09:42) FR: Road to the target blocked
(09:58) HQ: Copy. Waiting for further instructions

Figure 17: The Mobile Responder Tool.

of such systems, which can be repeated millions of times but miss the *critical qualitative insights* that can only be obtained with real users.

Specifically, targets (as identified by UAVs) were created on a virtual map that could be viewed by participants playing the role of the HQ and those playing the FRs on their mobile responder tool. In order to uncover the interactional issues that arise in interleaving human and agent decision-making at the HQ level (i.e., Silver commanders) and at the FR level (i.e., Bronze FRs), we tested two versions of the system. In (the earlier) version 1, the planner agent instructs the FRs directly without involvement of HQ and uses REPLAN — a baseline approach for the underlying MMDP where it tries replanning when a plan is rejected (Ramchurn et al., 2015). HQ is only provided with a map-based interface and communication channel to observe the game play and monitor the planner's allocations to the FRs. In order to intervene in the allocations, HQ has to make use of the broadcasting communication channel; hence, we refer to this design as a *human-on-the-loop* arrangement to reflect the fact that the Silver commander only intervenes if she identifies the need for it (as opposed to an 'in-the-loop' scheme which always requires human involvement in making choices).

In version 2 (as implemented in the final HAC-ER system), we adopted the mixed- initiative task allocation interfaces described in the previous sections that implement a *human-in-the-loop* arrangement and our novel TPP planning algorithm (as described in Section 6.1). We evaluate whether FRs are more likely to accept and complete task assignments computed by the planner agent if the plans account for possible rejections and are first analysed, modified, and validated by the Silver commanders (*human-in-the-loop*), than when sent directly to the FRs by the planner (Ramchurn et al., 2015).[11]

In the remaining sub-sections, we first describe the procedure used in the experiments and then detail the results. We then discuss these results and their implications for the design of HACs for mixed-initiative task allocation.

### 6.3.1 PROCEDURE

To evaluate the system, we ran field trials of the Mixed-Initiative Task allocation interface as a mixed-reality game where tasks are simulated in the virtual world and participants act as players in a disaster response game where a radioactive cloud is spreading over the simulated disaster zone. A more detailed description of the game is given by Ramchurn et al. (2015).

For both versions 1 and 2, we ran two field trials in order to evaluate the performance of the teams. For the four trials, a total of 40 participants were recruited. All participants were reimbursed with 15 GBP for the trial duration of 1.5 hours. For each game session, there were 2 HQ players and 8 FRs. The majority of participants were University students, some were staff. The HQ players were recruited among researchers at the University of Nottingham, as a short training session (about half an hour) with the HQ interfaces was required to learn the complexity of the task assignment interface.

The size of the game area on the local university campus was 400 by 400 meters, without heavy traffic. The terrain of the game area included grassland, a lake, buildings, roads, and footpaths and lawns. There were two drop off zones and 20 targets in each trial. A pilot study showed that this was a challenging, yet not overwhelming number of targets to collect in a 30-minute game session. There were four targets for each of the four target types. The pattern of cloud movement and expansion was the same for all trials.

### 6.3.2 RESULTS

A previous paper has unpacked the social implications of agent-based planning support in version 1 in greater detail (Jiang, Fischer, Greenhalgh, Ramchurn, Wu, Jennings, & Rodden, 2014). Here, we focus on key metrics that provide a means to compare compliance (task acceptance) and team performance (task completion) between versions 1 and 2.

Figure 18 shows how task assignments were acted upon in version 1. Specifically, 51 assignments were created by the planner and sent to FRs. 24 were accepted, while 11 were rejected or did not receive a response, i.e., only one or none of the two involved players responded. Out of the accepted tasks, 15 were completed successfully. An additional 8 tasks were completed that had not received a response (2 of which without agent instruction).

Figure 19 shows how task assignments were acted upon in version 2. Due to the mixed-initiative assignment interface, the planning agent created a total of 45 task assignments with an additional 5 assignments created manually by HQ. HQ approved a total of 39 assignments out of 45. FRs accepted most of the HQ-approved assignments (30 out of 39). Only 1 assignment was rejected by FRs and 8 assignments did not receive a response. During task execution, occasional HQ inter-

[11] In pilot studies with only version 1 of the system but using either REPLAN or TPP in the planner agent, it was found that participants rejected plans many more times using REPLAN than for TPP (8 times for REPLAN and none for TPP). Moreover, it was found that, with REPLAN, the participants only completed 75% of the tasks in 19 minutes while with TPP, they completed 80% of the tasks in 17 minutes (i.e., more tasks faster than REPLAN). Practically, none of the participants had depleted any of their health points in the run with TPP while, with REPLAN, the average health of FRs was 80%.

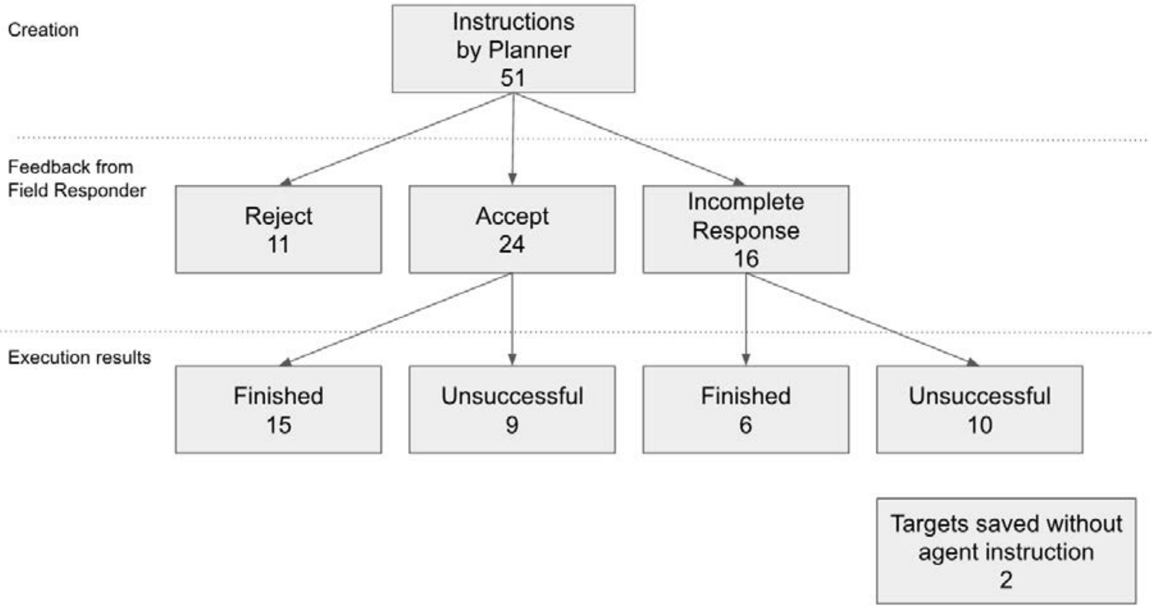


Figure 18: How task assignments were acted upon in version 1.

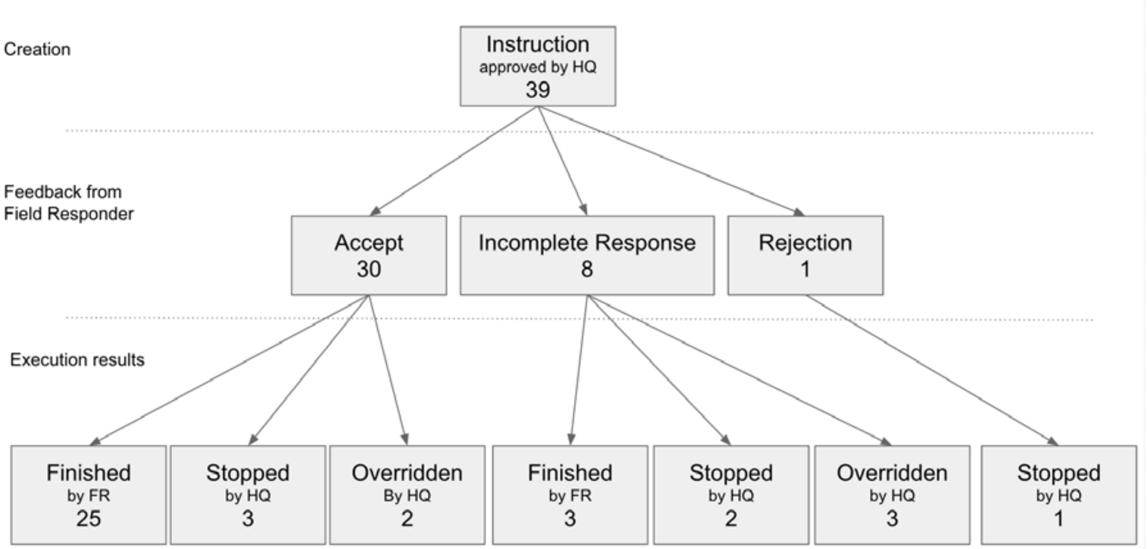


Figure 19: How task assignments were acted upon in version 2.

| | Success rate | Failure rate | Acceptance rate |
|---|---|---|---|
| Version 1 | 21/51 (42%) | 30/51 (58%) | 24/51(47%) |
| Version 2 | 28/39 (72%) | 11/39 (28%) | 30/39(77%) |

Table 1: Result overview.

ventions (i.e., directly contacting FRs over chat) resulted in 5 task cancellations and 5 assignments being overridden.

Table 1 shows key metrics for both versions. Compared to version 1, the task assignments in version 2 trials have relatively higher acceptance rate. 30 out of 39 (77%) assignments are accepted by the field players, while only 21 out of 51 (47%) assignments are accepted in version 1. An independent samples t-test indicated that acceptance rate was significantly higher for version 2 (M = 0.77, SD = 0.43), than for version 1 (M = 0.47, SD = 0.5), t(87) = 3.04, p = 0.003. Levene's test indicated unequal variances (F = 19.45, $p < 0.001$), so degrees of freedom were adjusted from 88 to 87.

In addition, an independent samples t-test shows that the completion rate of tasks in version 2 (M = .72, SD = 0.46) is also significantly higher than that in version 1 (M = 0.42, SD = 0.5), t(85) = 3.04, p = 0.003. Again, Levene's test indicated unequal variances (F = 6.5, p = 0.012), so degrees of freedom were adjusted from 88 to 85.

In summary, the results show significant improvements from version 1 to version 2 in the key evaluation metrics of acceptance and completion of task assignments.

### 6.3.3 DISCUSSION

The results show that task acceptance and completion has been significantly improved. The communication between HQ and the FRs was largely unproblematic in the final version (i.e., version 2), and most targets were successfully evacuated according to plan. We can attribute these results to a number of improvements based on what we learnt from trials of an early prototype of the system (Jiang et al., 2014). Specifically, the current task allocation is shown as a graphical overlay on the mobile map, not just as a textual instruction given by the planning agent in version 1. This seems to significantly reduce the FR's confusion about their current target and team-mate and where to find them.

Further, the HQ players in the first version were observed to struggle to intervene in the planning process, and we believe the performance differences are linked to the distinction between an *in-the-loop* arrangement (version 2) and an *on-the-loop* arrangement in the first version. In the latter arrangement, the only way for HQ to intervene in the planning is to send unstructured text messages on the broadcast channel. HQ's ability to intervene to support the coordination of FRs has been greatly enhanced by the mixed-initiative task allocation interface introduced in the *in-the-loop* arrangement (see Figure 16). Our evaluation has not only shown that task allocations computed by the planner are more likely to be accepted by FRs when there is a human in the loop that confirms or modifies each allocation according to the situation at hand, but also that this arrangement leads to a better task completion rate. More broadly, the move towards a stronger *in-the-loop* arrangement highlights the need for interfaces that provide means for humans to moderate and intervene in agent-based planning in order to respond to situational contingencies. This echoes results seen in deployments of the Mixed-Initiative Multi-UAV controller as described in the previous section.

| PROV terms | Examples in HAC-ER |
|---|---|
| Entity | Reports from the crowd, UAV targets, tasks for FRs, task allocation plans (by the Planner Agent), instructions in a plan |
| Activity | Aggregating crowd reports into targets for UAVs, annotating target types (UAV operators), accepting an instruction (FRs), carrying out an assigned task (FRs) |
| Agent | Crowd reporters, CrowdScanner, UAV operators, Planner Agent, FRs |

Table 2: Examples of provenance entities, activities, and agents in HAC-ER.

Now, having described the main task allocation and situational awareness components of HAC-ER, we next turn to the information infrastructure that underpins it. In particular, to ensure that all the components of the system can rely on timely and relevant information, we next introduce the Provenance-tracking system that ensures the system is accountable.

## 7. Tracking Information and Decisions in HAC-ER

As discussed in Section 3, HAC-ER consists of loosely-coupled components that involve collectives of humans and agents (see Figure 3, which depicts the flows of information between the components). Since making mistakes in our domain could potentially put lives at risk, tracking the provenance of information fed into decision making is a critical requirement. In this section, we describe how provenance of information in HAC-ER is tracked (Section 7.1) and used to improve awareness of changes across its components (Section 7.2).

### 7.1 Tracking Provenance

The World Wide Web Consortium defines provenance as "a record that describes the people, institutions, entities, and activities involved in producing, influencing, or delivering a piece of data or a thing" (Moreau & Missier, 2013). In this context, an Entity is something known to hold for a period of time. It had not existed before it wasGeneratedBy a specific Activity, e.g. a process, an action, or a task; and it ceased to exist once it wasInvalidatedBy an activity. In addition, an Agent is something that bears some form of responsibility for an activity taking place or for the existence of an entity. Note that the concept of a *provenance agent* is not limited to just software agents in HACs, but also includes people and institutions. See Table 2 for some examples in HAC-ER.

In our system, when a piece of information is produced by one of the components, the component records the inputs it used in the production of that piece of information and the agent(s) and/or human(s) that were involved. Figure 20 shows an example of provenance recorded by the UAV controller component. In the example, the entity `uav/target/9.2` represents a target identified by the UAV Controller component in HAC-ER. It was generated by a "UAV Verification" activity carried out by the UAV Silver commander who verified the target `cs/target/9.1` reported by the CrowdScanner and marked it as an "Infrastructure Damage". Therefore, the UAV controller recorded that the verified target `uav/target/9.2` wasDerivedFrom `cs/target/9.1` and wasAttributedTo `uav_silver_commander`. Examining this provenance, either when the entity `uav/target/9.2` is used during an operation, or in a much later audit when the operation has finished, allows us to track back to the origin of the information and to answer questions such as "who was responsible for this information" (via the wasAttributedTo relations) and "on which other

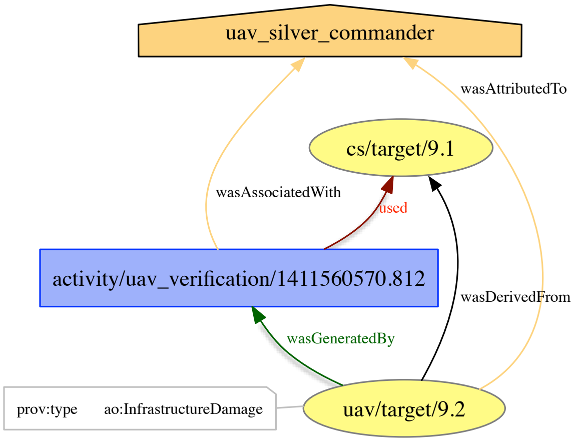


Figure 20: A provenance graph in HAC-ER: The provenance of a target (uav/target/9.2) annotated by the UAV Bronze FR, in which ellipses are entities, boxes are activities, and houses are agents.

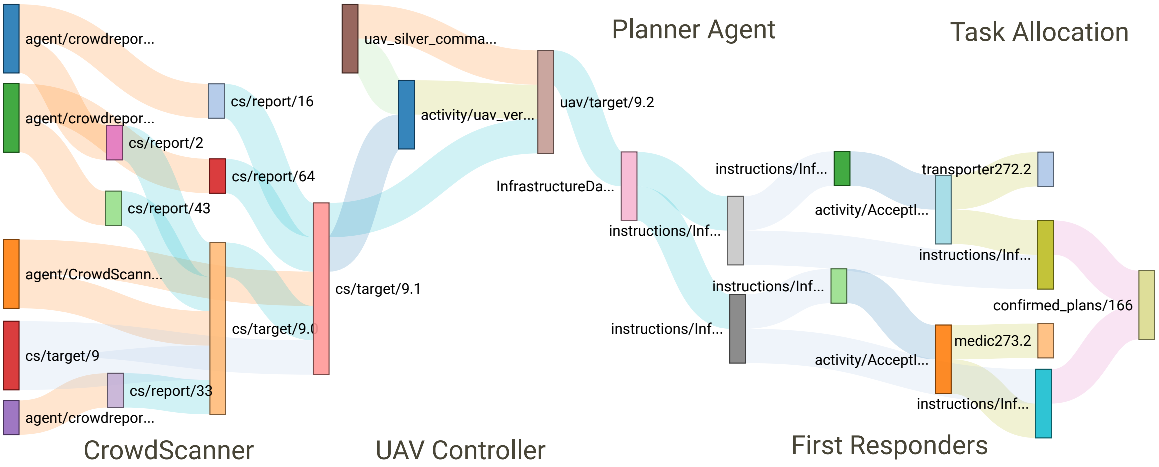


Figure 21: The Sankey diagram from the provenance of the entity confirmed_plans/166 showing the influences flowing (from left to right) to it, which was a plan that was originally generated by the Planner Agent and then later accepted by two FRs, transporter272 and medic273. The colours of the flows depict the type of influences, e.g. derivation (light blue), attribution (orange), generation (green), etc. The grey text labels were added to highlight the components in which the influences took place.

information it depended” (via the wasDerivedFrom). The capability to answer the latter is of particular importance during an operation as information in our context is often uncertain and changing over time. This is demonstrated later in Section 7.2.

Provenance information recorded in HAC-ER is stored in a purpose-built repository for provenance, called ProvStore (Huynh & Moreau, 2014). Individual HAC-ER components (i.e., the

CrowdScanner, UAV Controller, and Task Allocation) record the provenance of information and data generated in each of their activities and report it to ProvStore once the activity completes. The provenance of any entity can then be retrieved from ProvStore whenever it is required. Figure 21, for instance, shows the Sankey diagram generated from the provenance of `confirmed_plan/166`[12] that was queried from ProvStore. As can be seen, the result goes back all the way to the original crowd reports (and their reporters) aggregated by the CrowdScanner (some entities are omitted due to space constraints). It demonstrates the involvement of *all* the components in HAC-ER at various stages, leading to the eventual production of the `confirmed_plan/166`. Should any information in the chain of influences shown in Figure 21 later be discovered as unreliable or incorrect, it would be possible to assess its potential impact by querying from ProvStore to find all of its *dependants* — information that was directly or indirectly derived from it (via wasDerivedFrom relations). This is indeed what the provenance agent monitors in order to ensure that responders on the ground and their commanders are aware of potentially adverse changes in dynamic situations that are typical in disaster areas. In the next section, we describe how this is achieved.

### 7.2 Monitoring Decisions

In an ongoing operation, available information is typically uncertain and/or incomplete (Villaveces, 2011). In HAC-ER, for instance, targets identified from aggregating crowd reports are inherently uncertain (see Section 4); new (and more trustworthy) reports from FRs, for example, can invalidate targets that are already assigned to UAVs (e.g., a new report saying that a riot has been contained or food stocks have been found). UAV operators can also make mistakes when creating targets for UAVs and later, modify them. However, during the course of an operation, incorrect information may have already been propagated to the planner agent, resulting in assignments for FRs. Those assignments may have subsequently been confirmed by a Silver commander before the incorrect information is discovered. In order to be in control of such dynamic situations and to effectively manage such changes, the provenance agent tracks significant information changes at ProvStore that may have impacts on decisions already made. It works as follows:

- **Listening to information invalidation**: ProvStore provides an interface for external services to register to be notified when an event of interest occurs. Whenever prior information is invalidated, decisions and actions must be revised immediately. Therefore, the provenance agent asks ProvStore to notify it with any invalidation of existing information, represented as entities, asserted by one of HAC-ER's components.

- **Identifying potential impacts**: Upon receipt of a notification of invalidation, the provenance agent identifies all the dependants of the invalidated entity by querying ProvStore for the transitive closure over the wasDerivedFrom relation ending at the entity, i.e. all entities that can reach the invalidated entity via the wasDerivedFrom relation. For example, Figure 22 shows a chain of information derivations in HAC-ER. In this figure, the dependants of `uav/target/9.2` are the three entities on its right as it can be reached either directly or indirectly from those. If any of the dependants are confirmed decisions, e.g. entity `confirmed_plan/166`, the provenance agent will notify their owners in the next step.

- **Notifying affected stakeholders**: If some information or a decision was arrived based on some input(s) that are now invalidated, its "creator" may need to re-evaluate their decision.

[12] A "confirmed" plan is a task allocation plan that has been confirmed by a Silver commander and the two involved FRs.

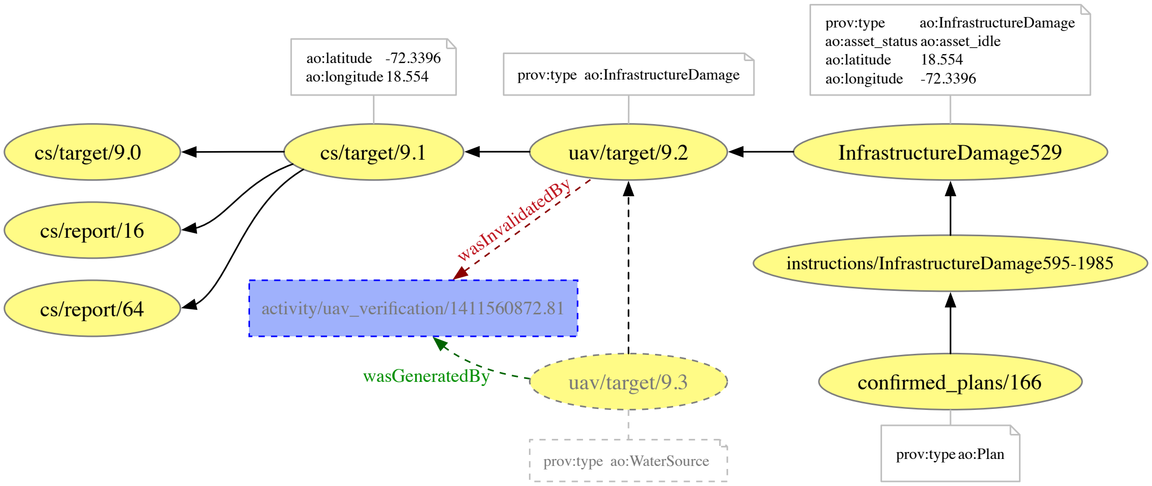


Figure 22: A chain of derived information tracked across HAC-ER with the wasDerivedFrom relations between the entities depicted by the black edges. At a later point (showing in dotted lines), uav/target/9.2 was revised and replaced by uav/target/9.3. Such events are monitored by the provenance agent to ensure that their potential impact on the dependants of the invalidated information is assessed appropriately.

Hence, in such cases, the Provenance Agent will send a notification to the affected stakeholders, informing them of the fact that one of their assumptions has just become invalid. At the same time, the notification also provides the list of entities that are potentially affected by this, along with the provenance information that has all the details for further investigation. As an example, the entity uav/target/9.2 (in Figure 22), may have been mistakenly identified as an “Infrastructure Damage”. The error was later discovered and corrected, resulting in a new version of the target (uav/target/9.3) and the original version being invalidated. The dependants of uav/target/9.2 were identified by the Provenance Agent as in the previous step, and a notification was sent to the commander who had confirmed the task allocation specified in confirmed_plan/166.

With the information and decisions in HAC-ER monitored by the provenance agent in tandem with ProvStore, our tests demonstrated that the system maintains situation awareness of the participants in a timely manner. Whenever the information used for task planning is invalidated, Silver commanders were notified of the changes in less than a minute after they occurred. More importantly, the generic provenance tracking and monitoring mechanisms implemented here are flexible and do not depend on this particular domain. These mechanisms are also designed to be scalable in that they only consider specific paths in the provenance graph to report on changes efficiently. Future changes in HAC-ER’s individual components, or even the addition of new components, will not affect the existing operation of the provenance agent as long as all the components report the provenance of their operation to ProvStore as previously described in Section 7.1.

## 8. Overall Evaluation of HAC-ER

The evaluation of a prototype such as HAC-ER is a challenging task as it involves a number of components, each with their individual complexity and the range of potential scenarios in which they could be tested. Hence, while these components could be individually validated through field trials or lab studies (as shown in individual sections or cited papers), the whole system was evaluated through a series of focus groups that involved disaster response experts, academics, and volunteers from disaster response charities, with a focus on qualitative results (i.e., common themes emerging from interview questions or from observed behaviours). The focus groups involved running the participants through a presentation that used the Haiti earthquake as the running example to highlight the challenges that emergency responders faced. The participants were asked to experience the rescue mission alongside researchers who described how the controls of the different interfaces would gather data from the Crowdscanner, deploy multiple UAVs and FRs on the ground, and also visualise the provenance information gathered throughout the mission.

The key results of such activities are to highlight potential design guidelines and capture the open questions the participants raised. Throughout the session, which typically lasted 30 to 45 minutes, and at the end of each session the participants were asked their opinion on each element of the system as well as how they felt about the overall experience. While most of the participants were enthusiastic about the capabilities offered by the system, they raised interesting issues and offered different perspectives on the applicability of the system. In particular, the key emerging insights from these studies as follows:

- Information gathering in disaster response: typical disaster response in the aftermath of earthquakes takes place over a matter of days rather than hours as expected within HAC-ER. Such operations require that multiple parties come to an agreement as to what information they will share and on what occasions they will share it or use it. For example, using the Crowdscanner to gather data from partner organisations (e.g., other emergency responders, charities, local government) was found to be particularly useful, but less so when it involved information which could be easily manipulated. While the system allows the users to assign a prior level of trust to such information, it was not clear how such values would be selected based on the source of the data. This could be addressed by using datasets from previous disaster situations, for which reports have been verified, to identify appropriate values for hyper-parameters that control the prior level of trust for each type of data source. Furthermore, as information is passed from the Crowdscanner to other parts of the system, participants found the system lacking in terms of being able to visualise the Bayesian heatmap on the same interface that allowed them to coordinate their UAVs or FRs. Having this information as layers on the same screen would allow them to see how the situation unfolded and how their information gathering efforts were helping to shape the Bayesian Heatmap. In general, participants appreciated the fact that the different information gathering components could be run in different locations and by multiple teams, not necessarily from the same organisation. However, this tended to be in tension with the need to restrict access to information to different types of users and also with the ability to integrate different tools.

- Coordinating FRs and UAVs: participants found that while the Mixed-Initiative Task Allocation system works well to track responders on the ground, the task allocation method it provided assumed a reliance on mobile connectivity and mobile phone interfaces. Experts did not like this over-reliance on mobile connectivity even if they did have access to such tools. Instead, they would have preferred it if the system allowed multi-modal communication methods (e.g., voice, text, etc.) and not necessarily rely on the features of a smart phone. Indeed, in other work (Fischer, Reeves, Rodden, Reece, Ramchurn, & Jones, 2015), we have explored how responders sometimes use paper maps and walkie-talkies or satellite phones to communicate. In addition to this, it was also suggested that UAVs should ideally be directly coordinated with the FRs so that the information they gather allows the FRs to adapt their mission according to unfolding threats and newly discovered targets. Hence, rather than going through the loop described in Section 3, the system could involve a number of smaller decision-making loops involving different parts of the system. Building upon such insights, we have started to investigate algorithms to coordinate human and UAV tasks (Wu, Ramchurn, & Chen, 2016) but more work is needed to develop interaction mechanisms and algorithms to support joint human and robotic activities (as opposed to humans only supervising UAVs or humans supervising other humans with agent-based support as in HAC-ER).

- Design and evaluation methodology: while employing methodologies from Design Ethnography and HCI in our studies, it became clear that the trial of advanced prototypes such as HAC-ER with domain experts required adjusting of our study techniques and design. In particular, it was important to adjust the way scenarios are presented and how participants are trained to engage with such systems. For example, on a number of occasions, participants expected that agents would have more advanced capabilities than they actually had, such as the ability to recognise casualties automatically or optimally coordinate in all circumstances. Hence, in such cases, it is important to establish detailed training material to clarify what capabilities are actually provided by the technology. Moreover, experts tended to apply their own judgement to the operation of the system based on their existing work practices (e.g., prioritise casualties over everything else or record every step they take on a paper map to have a resilient record of events). In trials with non-experts, we observed that the gamification of the use of HAC-ER (e.g., winning points for finding casualties) was much more important than for the domain experts. These observations led to the refinement of the system to allow for greater customisation of the planning tools as well as the development of several other prototypes to cater for their needs. Thus, out of the work on HAC-ER, other tools were developed such as the Augmented Bird Table (a tool to digitally record features and events on paper maps) (Fischer et al., 2015) and CrowdAR (a crowdsourcing-based real-time computer vision system for UAV imagery) (Salisbury, Stein, & Ramchurn, 2015).

- A metaphor for new types of organisations: many participants, typically with expertise beyond emergency response, were keen to point out that HAC-ER's usefulness was in the way it interleaved human and machine intelligence in a number of connected components which made it greater than the sum of its parts. In particular, it was seen as a metaphor for a distributed organisation whose individual functional elements are closely coordinated, with their interdependencies monitored in real-time. The visualisations and agent-based approaches in HAC-ER demonstrated how different parts of an organisation could be monitored or managed, partially autonomously. As we learnt through the development of HAC-ER, implementing such advanced solutions, however, requires a significant effort in running participatory design workshops and trials, involving practitioners and technology experts working closely to develop solutions iteratively. Moreover, a balance needs to be struck between being purely

> technology-focused and task-focused to allow for solutions that provide new capabilities and improve the organisation's performance while keeping the learning curve manageable.[13]

In general, the development and evaluation of HAC-ER led to a number of advances in a number of research areas including machine learning, HCI, multi-agent systems but also in improving the work practices of information gathering organisations and field responders. Such results would not have been possible without having a long-running (5 years) project that brought together experts from different disciplines, ready to go beyond their usual research topics to adopt new approaches and work together to address key challenges (see a case study video of the project at `http://goo.gl/mRTVmS`). HAC-ER, in this sense, can be regarded as an exemplar of how to bring together multi-disciplinary research to bear upon real-world problems while, at the same time, generating significant research results (Jennings, Moreau, Nicholson, Ramchurn, Roberts, Rodden, Rogers, & Westley, 2016).

## 9. Conclusions

In this paper, we presented a prototype disaster management system based on Human-Agent Collectives. The individual components of HAC-ER demonstrate how humans and agents can coalesce in flexible social relationships to manage information and decisions, in order to address specific challenges faced in gathering a high level of situational awareness from crowds and UAVs, and in allocating tasks to first responders on the ground. HAC-ER has been field tested with more than 100 users so far and validated by real-world emergency responders.

These trials show system performance is improved when greater control over autonomy is given to users (e.g., in the multi-UAV and the task allocation systems in particular). Furthermore, emergency responders found that tracking the provenance of information from the Crowdscanner significantly improves their confidence in information coming from crowds. To achieve this we needed to develop new AI-based solutions. Specifically, we developed a Bayesian heatmap technique that uses Gaussian processes and IBCC to aggregate crowd reports and present them in a meaningful form to FRs. Moreover, we developed algorithms and interaction techniques to integrate human input in decentralised coordination algorithms as well as a novel $k$-RMMDP model and TPP algorithms for decision-making under uncertainty.

To deploy HAC-ER in a real-world disaster scenario requires mainly practical improvements to be made. For example, crowdsourcing platforms would need to open up their data APIs for the Crowdscanner to tap into to allow for online updating of the source data (reports) and the Bayesian heatmap. In turn, for the multi-UAV system to be deployed, aviation authorities would need to allow multiple UAVs to be flown in the same air space, while for the mixed-initiative task allocation system to be deployed, we would require a robust, deployable, mobile data infrastructure to be in place. As a first step, future work will look at deploying HAC-ER in disaster response training exercises and evaluating our interaction mechanisms and agents *in situ*.

In general, HAC-ER is a showcase for highly multi-disciplinary scientific endeavours. It involved the coming together of experts from machine learning, HCI, Design Ethnography, and agent-based computing as well as volunteers and domain experts. Moreover, work in HAC-ER has opened up new avenues of research, particularly in the study of trust in autonomy or autonomy within cyber-

[13] As we noted while working on HAC-ER, developing new technology for emergency response planning that requires significant training time is not particularly favoured by emergency responders (Fischer et al., 2015). This is particularly problematic when such technology adds an overhead to their activities and the benefits from using such technology are not immediate at the point of use or only visible to some of the actors in the organisation.

physical systems that involve autonomous “things” (e.g., cars, thermostats, homes, etc.). The development of the platform was undertaken over the course of the five-year lifetime of the ORCHID project, with several components iteratively developed and tested, and eventually integrated within a common framework. The success of this process was built upon both a thirst to go beyond typically siloed research areas in order to break new ground at the intersection of these disciplines, but more importantly, the acceptance of others’, potentially conflicting, research methods. Thus, while it was easier for those with an AI background to relate in terms of the methodologies employed for the evaluation of their algorithms (e.g., statistically significance through large numbers of trials, or validation through comparison with multiple alternative algorithms), it was much more difficult for them to relate to qualitative methods typically employed in an HCI or Design Ethnography that seek to understand the finer reasons for why human systems succeed or fail, sometimes using quantitative methods as well as qualitative ones. This led to debates among the team about what constitutes a “scientifically valid result” for systems involving humans and agents. However, such barriers were overcome through the leadership of more experienced investigators and by developing trusted relationships among the research partners over multiple joint activities (e.g., joint fieldwork, co-developed platforms and algorithms, hackathons, showcase events, student events). Out of this collaborative effort, new methodologies were evolved to allow researchers to evaluate HACs and generate valid results for different research communities (e.g., envisionment workshops to expose users to potential futures or simulated machine intelligence to replicate the functionality of agents, or even using crowdsourcing platforms to integrate human intelligence into AI-driven systems).

## Acknowledgments

Note that this paper builds upon our previous work (Ramchurn, Simpson, Fischer, Huynh, Ikuno, Reece, Jiang, Wu, Flann, Moreau, Rodden, & Jennings, 2015) which won the Best paper award on the Innovative Applications Track at AAMAS 2015. It extends the latter to include results of experiments run with individual components of HAC-ER and provide more details on the background to our work. This work was done as part of the ORCHID programme funded by EPSRC (Grant EP/I011587/1). We also wish to thank RescueGlobal for their valuable feedback and for participating in our trials. Finally, we wish to thank all the anonymous reviewers whose feedback was invaluable in bringing out the best parts of this work to light.

## References

Altay, N., & Labonte, M. (2014). Challenges in humanitarian information management and exchange: evidence from haiti. *Disasters*, *38*(s1), S50–S72.

Amador, S., Okamoto, S., & Zivan, R. (2014). Dynamic multi-agent task allocation with spatial and temporal constraints. In *Proceedings of the 28th Conference on Artificial Intelligence (AAAI)*, pp. 1384–1390.

Baraniuk, C. (2015). Earthquake artificial intelligence knows where damage is worst. *New Scientist*, *3041*.

Barto, A. G., Bradtke, S. J., & Singh, S. P. (1995). Learning to act using real-time dynamic programming. *Artificial Intelligence*, *72*(1), 81–138.

Bellman, R. (1957). *Dynamic Programming*. Prentice Hall.

Bertuccelli, L., Beckers, W., & Cummings, M. (2010). Developing operator models for UAV search scheduling. In *AIAA Guidance, Navigation, and Control Conference*.

Boutilier, C. (1996). Planning, learning and coordination in multi-agent decision processes. In *Proc. of TARK 1996*, pp. 195–210.

Bradshaw, J. M., Feltovich, P., & Johnson, M. (2011). Human-Agent Interaction. In Boy, G. (Ed.), *Handbook of HumanMachine Interaction*, chap. 13, pp. 293–302. Ashgate.

Chen, S., Wu, F., Shen, L., Chen, J., & Ramchurn, S. D. (2015). Decentralized patrolling under constraints in dynamic environments. *IEEE Transactions on Cybernetics (IEEE SMC)*, *46*(12).

Cheney, J., Chiticariu, L., & Tan, W.-C. (2009). Provenance in Databases: Why, How, and Where. *Foundations and Trends in Databases*, *1*(4), 379–474.

Corbane, C., Saito, K., Dell'Oro, L., Bjorgo, E., Gill, S. P., Emmanuel Piard, B., Huyck, C. K., Kemper, T., Lemoine, G., Spence, R. J., et al. (2011). A comprehensive analysis of building damage in the 12 january 2010 mw7 haiti earthquake using high-resolution satelliteand aerial imagery. *Photogrammetric Engineering & Remote Sensing*, *77*(10), 997–1009.

Crowley, J., & Chan, J. (2010). 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.

Cummings, M. L., Brzezinski, A. S., & Lee, J. D. (2007). The impact of intelligent aiding for multiple unmanned aerial vehicle schedule management. *IEEE Intelligent Systems*, *22*(2), 52–59.

Cummings, M. L., Clare, A., & Hart, C. (2010). The role of human-automation consensus in multiple unmanned vehicle scheduling. *Human Factors*, *52*(1), 17–27.

Delle Fave, F. M., Rogers, A., Xu, Z., Sukkarieh, S., & Jennings, N. R. (2012). Deploying the max-sum algorithm for decentralised coordination and task allocation of unmanned aerial vehicles for live aerial imagery collection. In *Proc. of ICRA*, pp. 469–476.

Delle Fave, F. M., Shieh, E., Jain, M., Jiang, A. X., Rosoff, H., Tambe, M., & Sullivan, J. P. (2014). Efficient solutions for joint activity based security games: fast algorithms, results and a field experiment on a transit system. *Autonomous Agents and Multi-Agent Systems*, *29*(5), 787–820.

Dubash, R. M., ling Yen, I., & Bastani, F. B. (1992). Fault tolerant process planning and control. In *Proc. of the 16th Annual International Computer Software and Applications Conference*, pp. 188–193.

Fischer, J. E., Reeves, S., Rodden, T., Reece, S., Ramchurn, S. D., & Jones, D. (2015). Building a birds eye view: Collaborative work in disaster response. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pp. 4103–4112, New York, NY, USA. ACM.

Fox, M., Gerevini, A., Long, D., & Serina, I. (2006). Plan stability: Replanning versus plan repair. In *Proc. of ICAPS 2006*, Vol. 6, pp. 212–221.

Goodrich, M. A., McLain, T. W., Anderson, J. D., Sun, J., & Crandall, J. W. (2007). Managing autonomy in robot teams: observations from four experiments. In *Proc. of HRI*, pp. 25–32. ACM.

Grant, T. (2005). Unifying planning and control using an OODA-based architecture. In *Proceedings of the 2005 annual research conference of the South African institute of computer scientists and information technologists on IT research in developing countries*, pp. 159–170. South African Institute for Computer Scientists and Information Technologists.

Grant, T., & Kooter, B. (2005). Comparing OODA & other models as operational view C2 architecture. In *The 10th International Command and Control Research and Technology Symposium*.

Groth, P., & Moreau, L. (2013). PROV-Overview: An overview of the PROV family of documents. W3C Working Group Note, World Wide Web Consortium.

Gutierrez, P., Meseguer, P., & Yeoh, W. (2011). Generalizing adopt and bnb-adopt. In *Proc. of IJCAI*, pp. 554–559.

Hansen, E. A., & Zilberstein, S. (2001). LAO*: A heuristic search algorithm that finds solutions with loops. *Artificial Intelligence*, *129*, 35–62.

Hiatt, L. M., Zimmermanz, T. L., Smithz, S. F., & Simmons, R. (2009). Strengthening schedules through uncertainty analysis. In *Proc. of IJCAI 2009*.

Howard, R. A. (1960). *Dynamic Programming and Markov Processes*. MIT Press.

Huynh, T. D., Ebden, M., Venanzi, M., Ramchurn, S., Roberts, S., & Moreau, L. (2013). Interpretation of crowdsourced activities using provenance network analysis. In *The First AAAI Conference on Human Computation and Crowdsourcing*, pp. 78–85, Palm Springs, CA, USA.

Huynh, T. D., & Moreau, L. (2014). ProvStore: A public provenance repository. In *5th International Provenance and Annotation Workshop (IPAW'14)*, Cologne, Germany.

Imran, M., Castillo, C., Lucas, J., Meier, P., & Vieweg, S. (2014). Aidr: Artificial intelligence for disaster response. In *Proceedings of the companion publication of the 23rd international conference on World wide web companion*, pp. 159–162. International World Wide Web Conferences Steering Committee.

Jennings, N. R., Moreau, L., Nicholson, D., Ramchurn, S., Roberts, S., Rodden, T., Rogers, A., & Westley, A. (2016). ORCHID: The science of human-agent collectives — final report. Tech. rep., University of Southampton,UK. available at http://goo.gl/eZplS0 on 10/09/2016.

Jennings, N. R., Moreau, L., Nicholson, D., Ramchurn, S. D., Roberts, S., Rodden, T., & Rogers, A. (2014). On human-agent collectives. *Communications of the ACM*, *57*(12), 33–42.

Jensen, R. M., Veloso, M. M., & Bryant, R. E. (2004). Fault tolerant planning: Toward probabilistic uncertainty models in symbolic nondeterministic planning. In *Proc. of ICAPS 2004*, pp. 335–344.

Jiang, W., Fischer, J., Greenhalgh, C., Ramchurn, S., Wu, F., Jennings, N., & Rodden, T. (2014). Social implications of agent-based planning support for human teams. In *Collaboration Technologies and Systems (CTS), 2014 International Conference on*, pp. 310–317.

Kamar, E., Hacker, S., & Horvitz, E. (2012). Combining human and machine intelligence in large-scale crowdsourcing. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pp. 467–474. International Foundation for Autonomous Agents and Multiagent Systems.

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

Kocsis, L., & Szepesvári, C. (2006). Bandit based monte-carlo planning. In *Proc. of ECML 2006*, pp. 282–293.

Koes, M., Nourbakhsh, I., & Sycara, K. (2006). Constraint optimization coordination architecture for search and rescue robotics. In *Proc. of ICRA 2006*, pp. 3977–3982.

Lenox, T. L., Payne, T., Hahn, S., Lewis, M., & Sycara, K. (2000). Agent-Based Aiding for Individual and Team Planning Tasks. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, *44*(1), 65–68.

Lewis, M., Sycara, K., & Scerri, P. (2009). Scaling up wide-area-search-munition teams. *IEEE Intelligent Systems*, *24*(3), 10–13.

Macarthur, K., Stranders, R., Ramchurn, S., & Jennings, N. (2011). A distributed anytime algorithm for dynamic task allocation in multi-agent systems. In *Proceedings of the 25th National Conference on Artificial Intelligence (AAAI)*, pp. 701–706. AAAI Press.

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

Meier, P. (2014). *Digital Humanitarians: How Big Data Is Changing the Face of Humanitarian Response*. CRC Press.

Meuleau, N., & Smith, D. E. (2003). Optimal limited contingency planning. In *Proc. of UAI 2003*, pp. 417–426.

Miller, C. A., & Parasuraman, R. (2007). Designing for flexible interaction between humans and automation: Delegation interfaces for supervisory control. *Human Factors*, *49*(1), 57–75.

Moreau, L. (2015). Aggregation by provenance types: A technique for summarising provenance graphs. In *Graphs as Models 2015*, pp. 129–144, London, UK. Electronic Proceedings in Theoretical Computer Science.

Moreau, L., & Missier, P. (2013). PROV-DM: The PROV data model. Tech. rep., World Wide Web Consortium. W3C Recommendation.

Morrow, N., Mock, N., Papendieck, A., & Kocmich, N. (2011). Independent Evaluation of the Ushahidi Haiti Project. *Development Information Systems International*, *8*, 2011.

Musliner, D. J., Durfee, E. H., Wu, J., Dolgov, D. A., Goldman, R. P., & Boddy, M. S. (2006). Coordinated plan management using multiagent mdps.. In *AAAI Spring Symposium: Distributed Plan and Schedule Management*, pp. 73–80.

Nair, R., & Tambe, M. (2005). Hybrid bdi-pomdp framework for multiagent teaming.. *Journal of Artificial Intelligence Research (JAIR)*, *23*, 367–420.

Naja, I., Moreau, L., & Rogers, A. (2010). Provenance of decisions in emergency response environments. In McGuinness, D. L., Michaelis, J. R., & Moreau, L. (Eds.), *Provenance and Annotation of Data and Processes*, Vol. 6378 of *Lecture Notes in Computer Science*, pp. 221–230. Springer Berlin Heidelberg.

Nancy, Cooke, G. J., & Pederson Harry, B. B. (2006). 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.

Pineda, L., Lu, Y., Zilberstein, S., & Goldman, C. V. (2013). Fault-tolerant planning under uncertainty. In *Proc. of IJCAI 2013*, pp. 2350–2356, Beijing, China.

Puterman, M. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons.

Ramchurn, S. D., Farinelli, A., Macarthur, K. S., & Jennings, N. R. (2010). Decentralized coordination in robocup rescue. *The Computer Journal*, *53*(9), 1447–1461.

Ramchurn, S. D., Fischer, J. E., Ikuno, Y., Wu, F., Flann, J., & Waldock, A. (2015). A study of human-agent collaboration for multi-uav. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, IJCAI'15, pp. 1184–1192. AAAI Press.

Ramchurn, S. D., Huynh, T. D., Venanzi, M., & Shi, B. (2013). Collabmap: crowdsourcing maps for emergency planning. In *Proceedings of the 5th Annual ACM Web Science Conference*, pp. 326–335. ACM.

Ramchurn, S. D., Simpson, E., Fischer, J. E., Huynh, T. D., Ikuno, Y., Reece, S., Jiang, W., Wu, F., Flann, J., Moreau, L., Rodden, T., & Jennings, N. R. (2015). HAC-ER: a disaster response system based on human-agent collectives. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multi-agent Systems*, pp. 533–541. International Foundation for Autonomous Agents and Multi-agent Systems.

Ramchurn, S. D., Wu, F., Jiang, W., Fischer, J. E., Reece, S., Greenhalgh, C., Rodden, T., Jennings, N. R., & Roberts, S. (2014). AtomicOrchid: Human-agent collectives to the rescue (demostration). In *Proceedings of the 13th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pp. 1693–1694, Paris, France.

Ramchurn, S. D., Wu, F., Jiang, W., Fischer, J. E., Reece, S., Roberts, S., Rodden, T., Greenhalgh, C., & Jennings, N. R. (2015). Human–agent collaboration for disaster response. *Autonomous Agents and Multi-Agent Systems*, *30*(1), 82–111.

Reece, S., & Roberts, S. (2010). An introduction to gaussian processes for the kalman filter expert. In *2010 13th International Conference on Information Fusion*, pp. 1–9.

Reece, S., Roberts, S., Nicholson, D., & Lloyd, C. (2011). Determining intent using hard/soft data and gaussian process classifiers. In *Information Fusion (FUSION), 2011 Proceedings of the 14th International Conference on*, pp. 1–8. IEEE.

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

Salisbury, E., Stein, S., & Ramchurn, S. (2015). Crowdar: augmenting live video with a real-time crowd. In *HCOMP 2015: Third AAAI Conference on Human Computation and Crowdsourcing*, pp. 169–177.

Scerri, P., Tambe, M., & Pynadath, D. V. (2002). Towards adjustable autonomy for the real-world. *Journal of Artificial Intelligence Research*, *17*(1), 171–228.

Scerri, P., Pynadath, D., Johnson, L., Rosenbloom, P., Si, M., Schurr, N., & Tambe, M. (2003). 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, pp. 433–440, New York, NY, USA. ACM.

Schurr, N., Marecki, J., Lewis, J. P., Tambe, M., & Scerri, P. (2005). The defacto system: Training tool for incident commanders. In *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI)*, pp. 1555–1562.

Sheridan, T. B., & Verplank, W. L. (1978). Human and computer control of undersea tele-operators. Tech. rep., DTIC.

Silverman, B. G. (1992). Human-computer collaboration. *Hum.-Comput. Interact.*, *7*(2), 165–196.

Simpson, E., Roberts, S., Psorakis, I., & Smith, A. (2013). Dynamic Bayesian combination of multiple imperfect classifiers. *Intelligent Systems Reference Library series*, *Decision Making with Imperfect Decision Makers*.

Simpson, E., Roberts, S. J., Smith, A., & Lintott, C. (2011). Bayesian combination of multiple, imperfect classifiers. In *In proceedings of the 2nd International Workshop on Decision Making with Multiple Imperfect Decision Makers*.

Simpson, E. D., Venanzi, M., Reece, S., Kohli, P., Guiver, J., Roberts, S. J., & Jennings, N. R. (2015). Language understanding in the wild: Combining crowdsourcing and machine learning. In *Proceedings of the 24th International Conference on World Wide Web*, pp. 992–1002. International World Wide Web Conferences Steering Committee.

Smith, P., McCoy, C., & Layton, C. (1997). Brittleness in the design of cooperative problem-solving systems: the effects on user performance. *IEEE Trans. on Systems, Man and Cybernetics: Part A*, *27*(3), 360–371.

Sukthankar, G., Sycara, K., Giampapa, J. A., & Burnett, C. (2009). Communications for agent-based human team support. In *Handbook of Research on Multi-agent Systems: Semantics and Dynamics of Organizational Models*, pp. 285–313. IGI Global.

Tambe, M. (2011). *Security and Game Theory: Algorithms, Deployed Systems, Lessons Learned* (1st edition). Cambridge University Press, New York, NY, USA.

Villaveces, J. (2011). Disaster response 2.0. *Forced Migration Review*, *38*, 7–9.

Wagner, T., Phelps, J., Guralnik, V., & VanRiper, R. (2004). An application view of coordinators: Coordination managers for first responders. In *Proc. of AAAI 2004*, pp. 908–915.

Weitzner, D. J., Abelson, H., Berners-Lee, T., Feigenbaum, J., Hendler, J., & Sussman, G. J. (2008). Information accountability. *Communications of the ACM*, *51*(6), 82–87.

Wilkins, D. E., Lee, T. J., & Berry, P. (2003). Interactive execution monitoring of agent teams. *Journal of Artificial Intelligence Research (JAIR)*, *18*, 217–261.

Wu, F., & Jennings, N. R. (2014). Regret-based multi-agent coordination with uncertain task rewards. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI)*, pp. 1492–1499, Quebec City, Canada.

Wu, F., Ramchurn, S. D., & Chen, X. (2016). Coordinating human-UAV teams in disaster response. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 524–530.

Wu, F., Ramchurn, S. D., Jiang, W., Fischer, J., Rodden, T., & Jennings, N. R. (2015). Agile planning for real-world disaster response. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 132–138.

Wu, F., Zilberstein, S., & Chen, X. (2010). Trial-based dynamic programming for multi-agent planning. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI)*, pp. 908–914, Atlanta, United States.