Abstract—We introduce a new technique for coordinating teams of unmanned aerial vehicles (UAVs) when deployed to collect live aerial imagery of the scene of a disaster. We define this problem as one of task assignment where the UAVs dynamically coordinate over tasks representing the imagery collection requests. To measure the quality of the assignment of one or more UAVs to a task, we propose a novel utility function which encompasses several constraints, such as the task’s importance and the UAVs’ battery capacity so as to maximise performance. We then solve the resulting optimisation problem using a fully asynchronous and decentralised implementation of the max-sum algorithm, a well known message passing algorithm previously used only in simulated domains. Finally, we evaluate our approach both in simulation and on real hardware. First, we empirically evaluate our utility and show that it yields a better trade off between the quantity and quality of completed tasks than similar utilities that do not take all the constraints into account. Second, we deploy it on two hexacopters and assess its practical viability in the real world.

I. INTRODUCTION

Coordinating teams of unmanned aerial vehicles (UAVs) for exploration or recognition tasks in disaster scenarios is a key research challenge. Recently, both research and industry have started investigating problems where these vehicles interact with first responders to provide real-time live aerial imagery of specific sites in the area of a disaster, such as damaged buildings or flooded streets. Each first responder is provided with a personal digital assistant (PDA) that he uses to submit imagery requests in the form of tasks and to collect such imagery as soon as it becomes available. The aim of the UAV team is then to complete these tasks in an effective fashion. Thus, the team should be able to coordinate, whilst interacting with such PDAs, to jointly decide over an assignment of the tasks that maximises the number of completed tasks, whilst preserving the limited battery life of the vehicles. When all the tasks cannot be completed, the team should also be able to discriminate between these tasks so that the most important ones are preferred.

Now, coordinating UAVs whilst interacting with first-responders presents various challenges. The importance of a task varies with time. For instance, a building on fire will surely burn out as time passes, or a building full of people will be gradually evacuated. Thus, addressing this dynamism is a key requirement of the problem. The termination of a task is uncertain. For instance, some tasks might require imagery to be collected for a long interval of time (e.g., a building on fire), whereas others might require it to be collected for a short interval (e.g., a burnt out building), but the exact time cannot be known with precision beforehand. Solution techniques need therefore to address this uncertainty. UAVs have a limited battery, therefore solution techniques need to be accurate and allow the UAVs to make effective joint decisions as their predicted travel time diverges from actuality. Finally, some of the UAVs can fail but the remaining team should continue the operation nonetheless. Consequently, coordination should be performed in a decentralised fashion.

In order to meet this requirements, a variety of approaches have been produced. The majority of these techniques work by having the vehicles make collective decisions that maximise a utility function \cite{3, 9}. However, whereas most of these approaches have been shown to yield an effective performance on a variety of similar problems, such as search and track and task assignment for situational awareness \cite{2}, \cite{6}, they do not or only partially meet the requirements of our problem. This is mainly due to two reasons. First, to define the former utility, canonical information metrics such as entropy or mutual information gain are typically used \cite{4}, \cite{6}, \cite{8}. However, since these metrics allow the UAVs to collectively decide the best trajectories to take to minimise the uncertainty over some specific feature of the environment, such as the position of a target or the temperature of a building, they are not suitable for our setting because they do not address the uncertainty over the tasks completion. To understand the second reason note that a vast amount of literature exists on coordination techniques for unmanned vehicles. Examples include market-based algorithms \cite{5}, \cite{7}, \cite{10}, negotiation algorithms inspired by game theory \cite{4}, \cite{6} and constraint optimisation algorithms \cite{2}. However, these approaches fail to meet some of the requirements of our problem. Indeed, market-based and negotiation algorithms are well known to require a lot of time to converge to good solutions. Thus they fail to meet the accuracy requirement. Similarly, constraint optimisation algorithms needs to be periodically re-run from scratch to incorporate the changes in the problem. Thus, they fail to take dynamism into account.

Against this background, we propose a novel solution to the problem of coordinating teams of UAVs for dynamic
real-time task assignment. We show how this problem can be cast into a decentralised optimisation problem and introduce a novel utility function in which several constraints, such as the task’s importance and the UAVs’ battery capacity are carefully weighted to maximise performance. In so doing, we improve over canonical metrics such as entropy or mutual information gain because we incorporate all the above mentioned properties into one single function. Next, we solve the coordination problem by using the max-sum algorithm, a message passing algorithm relying on the generalised distributive law (GDL) [1], which has been shown to yield efficient decisions on a variety of simulated problems, whilst being decentralised, accurate and robust to dynamism [11], [12]. Thus, we address here for the first time its implementation on system deployed in the real world. More specifically, our implementation requires very little memory since the platforms only need to keep track of the last messages received to make a decision or to compute new messages. Moreover, we distribute the computation between both the UAVs and the PDAs and thus the computation of the solution of our problem is not delegated only to the UAVs which are potentially the most unreliable part of the system. Next, we empirically evaluate our utility against similar ones but which do not take all the constraints into account. By so doing, we show that our utility yields a better trade-off between the quantity and quality of completed tasks. Finally, we describe our experience of deploying our techniques on two real UAVs and and show the effectiveness of max-sum in different scenarios, thus proving its practical viability.

The remainder of the paper is organised as follows: Section II formalises the details of the task assignment problem. Section III introduces our solution, by detailing the utility function and the max-sum algorithm for task assignment. Section IV presents our empirical evaluation and Section V concludes.

II. PROBLEM DEFINITION

The UAVs involved in our problem are rotorcrafts. These are chosen because they have a wide range of motion capacities (i.e. being able to take off and land vertically, hover, fly forward, backwards and laterally) that make them very suitable for collecting aerial imagery. Each first responder is provided with a PDA to submit imagery requests in the form of tasks. Each task represents a location (in geographic coordinates) for which imagery is required. To submit a task $T_i$, each first responder sets three properties: (i) priority $p_i = \{\text{normal}, \text{high}, \text{very high}\}$, representing the importance of the task (i.e. collecting imagery of an occupied building is more important than doing so for an empty one); (ii) urgency $u_i = \{\text{normal}, \text{high}, \text{very high}\}$ used to prevent tasks’ starvation and (iii) duration $d_i$, which defines the interval of time for which imagery needs to be collected. Note that a first responder does not know this duration with precision since it depends on the specific reason for which imagery is required (e.g. to search for a casualty or to check access to an area). Thus, three estimates are considered ($d_i = \{5 \text{ min}, 10 \text{ min}, 20 \text{ min}\}$). The information about each submitted task is then broadcasted by the corresponding PDA, so that the UAVs in the surrounding area that receive it add the task to the set of tasks that they can potentially attend. To complete a task, a UAV needs to fly to the specified location, station itself above it and stream live video to the PDA until the first responder indicates that the task is completed. The UAVs then jointly decide which task each vehicle should complete and, in so doing, they maximise the number of completed high priority tasks. We achieve this coordination by applying max-sum as discussed next.

III. THE COORDINATION ALGORITHM

In what follows, we describe the two main components of our approach: the function measuring the utility of each task assignment and the decentralised coordination approach based on the max-sum algorithm.

A. The Task Utility

The aim of the task utility function is to discriminate between all the UAVs that can attend the task and give a higher value to those that are best fit to complete the task given their position, the remaining time that they can remain on station (for each UAV $j$ we define its battery capacity in terms of its remaining flight time $b_j$, measured in seconds) and given the task’s properties ($p_i$, $u_i$ and $d_i$).

Formally, the utility $U_i$ of task $T_i$ is defined over the set of variables $x_i \subseteq x$, where each variable $x_j \in x_i$ represents one UAV $j$ that can attend task $T_i$ ($x = \{x_1, \ldots, x_M\}$ is then the set of all the UAVs involved in the problem). In more detail, each variable $x_j$ represents the decision of UAV $j$, that is to say, the task that it is currently attending. Formally, each variable $x_j$ takes value in the set $\mathcal{F} \subseteq \mathcal{F}$ of all the tasks that $j$ can attend ($\mathcal{F} = \{T_1, T_2, \ldots, T_N\}$ is then the set of all the submitted tasks). The use of these variables will be more clear in Section III-B.2 where we present our algorithm.

The utility $U_i$ is then defined as follows:

$$U_i(x_i) = p_i \cdot u_i \cdot t_i^{0 - t_i} \cdot \left[1 - e^{-\lambda_i(t_2-t_1)}\right]$$

(1)

where $t$ is the current time (in seconds) of the task and where $p_i$, $u_i$ and $t_i^0$ are respectively its priority, urgency and activation time (i.e. the time, in seconds, at which the task is submitted to the system).

In order to derive the utility $U_i$, the uncertainty over the termination of a task is defined as a Poisson process and is measured over the interval of time in which one or more UAVs can remain on station\footnote{$\lambda_i = \frac{1}{t_1}$ is the rate parameter of the Poisson process (i.e. each minute the probability that the task terminates is $\lambda_i$) and $d_i$ is the expected duration of the task (Section III).}. In more detail, given all the UAVs $j$ that can attend task $T_i$ (i.e. represented by all the variables in $x_i$), live imagery is going to be collected from $t_1 = \min_j t_{1_j}$ to $t_2 = \max_j t_{2_j}$. Here, $t_{1_j} = \frac{t_1}{V_j}$ is the time (in minutes) required by UAV $j$ to reach task\footnote{$l_{ij}$ is the Euclidean distance between the task and the UAV $j$ and $V_j$ is its velocity in m/s} $T_i$ and where
\( t^2_j = t + b_j \) is the remaining time (in minutes) that UAV \( j \) can remain on station. Thus, the probability \( P(t_1 \leq t^2_j \leq t_2) \) that the time at which \( T_i \) is going to terminate lies within the interval \([t_1, t_2]\) is defined as follows:

\[
P(t_1 \leq t^2_j \leq t_2) = \int_{t_1}^{t_2} \lambda_i \cdot \exp(-\lambda_i \cdot x) dx \tag{2}
\]

\[
= 1 - e^{-\lambda_i \cdot (t_2 - t_1)} \tag{3}
\]

This probability is then incorporated in the utility function defined by Equation 1.

Intuitively, the utility defined by Equation 1 measures the impact of each collective assignment \( x^j_i \in x_i \) of the UAVs that can attend task \( T_i \) to its completion. In more detail, for each of these assignments, \( t_1 \) and \( t_2 \) are calculated for all the UAVs \( j \) for which \( x^j_i = T_i \). Thus, in order to maximise the number of completed high priority tasks, the sum of their utilities is maximised. In so doing, the UAVs are able to make a variety of sophisticated decisions based on all the possible constraints of the problem. For instance, they will always choose tasks with higher priority (due to the factor \( p_i \) in Equation 1). If these have same priorities, the UAVs will always choose the task that has remained unattended for a longer interval of time (due to the factor \( 1 - \exp^{-\lambda_i \cdot (t_2 - t_1)} \) in Equation 1). In addition, multiple UAVs may attend a task if this extends the time span for which at least one UAV is on station above the task (due to the factor \( 1 - \exp^{-\lambda_i \cdot (t_2 - t_1)} \)).

B. The Task Assignment Algorithm

The algorithm presented in this section is based on an adaptation of the max-sum algorithm to solve task assignment problems [11]. We first describe how the problem is decentralised into a global utility function that is a sum of the utility of all the tasks to be attended and then we detail the algorithm.

1) Decentralised Optimisation Problem: Tasks can be submitted anytime. Thus, every second, the utility of a joint assignment of the tasks to the UAVs is calculated as the sum of the utilities of all the tasks that have currently been submitted

\[
U(x) = \sum_{j=1}^{\left|\mathcal{J}\right|} U_j(x_j). \tag{4}
\]

The optimal assignment is then an allocation of the tasks \( T_i \in \mathcal{J} \) that are currently submitted for each variable \( x_j \in x \) representing UAV \( j \), such that \( U \) is maximised:

\[
x^* = \arg \max_{x} U(x) \tag{5}
\]

In order to employ the max-sum algorithm, our problem needs to be encoded into a special graph called a factor graph, an undirected bipartite graph in which vertices represent variables and utilities, and edges the dependencies between them. Within our setting, each UAV \( j \) controls the variable node \( x_j \), whereas each PDA controls the set of utilities \( U_i \) corresponding to the tasks that it submitted. Figure 1 shows an example of a factor graph resulting from this allocation. The figure shows two UAVs \( (UAV_1 \text{ and } UAV_2) \) controlling two variables \( x_1 \) and \( x_2 \) and two PDAs: PDA\(_1\) controls two tasks \( T_1 \) and \( T_2 \) (and the corresponding utilities \( U_1 \) and \( U_2 \)) while PDA\(_2\) controls one task \( T_3 \) (and its utility

\[
U_3. \tag{6}
\]

Note that, given the dynamic nature of the problem, the topology of the factor graph varies continuously: function nodes corresponding to completed tasks disappear whereas new functions appear corresponding to new tasks. Similarly, variable nodes corresponding to UAVs out of battery life disappear whereas new variable nodes corresponding to new UAVs appear.

2) The Task Assignment Algorithm: The max-sum algorithm provides an efficient local message-passing procedure to compute the dependency of the global constraint function \( U \) on each of the variables \( x_j \) simultaneously. The function describing the dependency of \( U(x) \) on each variable \( x_j \) is referred to as the marginal function \( z_j(x_j) \) and is defined as follows:

\[
z_j(x_j) = \max_{x \setminus \{x_j\}} U(x) \tag{5}
\]

The marginal function calculated by Equation 5 determines, for each task \( T_i \) in the domain \( \mathcal{J} \) of each variable \( x_j \), the maximum value that \( U(x) \) can attain when \( x_j = T_i \). Each UAV can then calculate its best assignment by finding the solution of Equation 5

\[
x^j_i = \arg \max_{x_j} z_j(x_j). \tag{3}
\]

Max-sum computes these marginal functions by propagating messages between the functions \( U_i \) and the variables \( x_j \) of the factor graph. Each message conveys the maximum aggregate utility possible over the two components of the factor graph formed by removing the edge between \( U_i \) and \( x_j \) for each value \( T_i \) of the domain \( \mathcal{J} \) of \( x_j \). In other words, the size of each message is linear in the domain of variable \( x_j \). Thus, in settings where the number of tasks is very large, the size of these messages can increase arbitrarily and computing them can become overly expensive. To address this shortcoming, max-sum has been modified to reduce the size and the complexity of computing the messages within each node. The key idea is that since each function node \( U_i \) represents a task \( T_i \), any message passed between \( U_i \) and \( x_j \) only needs to convey the maximum aggregate utility, as discussed above, for \( x_j \) being assigned to \( T_i \) and for \( x_j \) being assigned to some other task (i.e. each message becomes a bi-valued array). Therefore, whereas in standard max-sum the messages are defined as functions of the variables, in the modified version, each

\[\text{The computation of the messages is then reduced from } O(|\mathcal{J}|^|\mathcal{A}(U)|) \text{ to } O(2^{|\mathcal{A}(U)|}) \text{ in the modified version (refer to [11] for more details).}\]
message is defined over an indicator function $\mathcal{S}_j(x_j)$ such that $\mathcal{S}_j(x_j) = 1$ when $x_j = T_j$ and $\mathcal{S}_j(x_j) = 0$ when $x_j \neq T_j$.

The specifics of the messages follows:

From variable $x_j$ to function $U_i$:

$$q_{j \rightarrow i}(\mathcal{S}_j(x_j)) = \begin{cases} 
\max_{x_i \in \text{Adj}(j) \setminus \{i\}} \left( r_{k \rightarrow j}(1) + \sum_{w \in \text{Adj}(j) \setminus \{k\}; r_{w \rightarrow j}(0)} \right) & (x_j \neq T_j) \\
\sum_{k \in \text{Adj}(j) \setminus \{i\}} r_{k \rightarrow j}(0) & (x_j = T_j)
\end{cases}$$

where $\text{Adj}(j)$ represents the set of indices of the functions adjacent to variable $x_j$. Procedure 1 describes $\text{computeVariableMessage}(x_j, U_i, R)$, the algorithm used to compute a message from variable $x_j$ to function $U_i$.

Procedure 1 $\text{computeVariableMessage}(x_j, U_i, R)$: The procedure that UAV $j$ uses for computing a message from variable $x_j$ to function $U_i$.

**Input:** $x_j$: the sending variable; $R$: the set of all the messages received by $x_j$ since the last time a message was computed; $U_i$: the destination function.

**Output:** $q_{j \rightarrow i}(\mathcal{S}_j(x_j))$

1: $q_{j \rightarrow i}(0) = -\infty$
2: $q_{j \rightarrow i}(1) = 0$
3: for all $r_{k \rightarrow j} \in R$; $k \neq i$ do
4: $\sigma = r_{k \rightarrow j}(1)$
5: for all $r_{w \rightarrow j} \in R$; $w \neq k$, $i$ do
6: $\sigma = \sigma + r_{w \rightarrow j}(0)$
7: end for
8: $q_{j \rightarrow i}(0) = \max(q_{j \rightarrow i}(0), \sigma)$
9: $q_{j \rightarrow i}(1) = q_{j \rightarrow i}(1) + r_{k \rightarrow j}(0)$
10: end for
11: return $q_{j \rightarrow i}(\mathcal{S}_j(x_j))$

From function $U_i$ to variable $x_j$:

$$r_{i \rightarrow j}(\mathcal{S}_j(x_j)) = \max_{x_i \setminus \{x_j\}} \left( U_i(x_i) + \sum_{k \in \text{Adj}(i) \setminus \{j\}} q_{k \rightarrow i}(\mathcal{S}_k(x_k)) \right)$$

where $\text{Adj}(i)$ represents the set of indices of the variables adjacent to function $U_i$. The algorithm $\text{computeFunctionMessage}(x_j, U_i, Q)$, shown in Procedure 2, describes the computation of $r_{i \rightarrow j}(\mathcal{S}_j(x_j))$.

Within our setting, messages are exchanged continuously over the factor graph. At any time, each UAV $j$ can then compute its best assignment $x_j^*$ as the sum of the messages flowing into $x_j$: $x_j^* = \arg \max x_j \sum_{i \in \text{Adj}(j)} r_{i \rightarrow j}(\mathcal{S}_j(x_j))$. This sum represents the solution to the maximisation problem defined by Equation 4 with respect to $x_j$. Thus, it represents the solution to the marginal function defined by Equation 5. However, here the factor graph contains cycles which changes the values of the solutions recovered by the algorithm. Indeed, max-sum is provably optimal, that is, guaranteed to converge to the optimal solution, only when the factor graph is acyclic. Nonetheless, the algorithm is often applied to cyclic factor graphs anyway and strong empirical evidence has shown that max-sum performs well in a variety of problems such as monitoring spatial phenomena [12] and patrolling [13]. Within these settings, the aggregation of the messages flowing into each variable only represents an approximate solution to the maximisation problem defined by Equation 4 with respect to $x_j$ (i.e. it represents an approximation of the marginal function defined in Equation 5). The procedure $\text{makeDecision}(x_j, R)$ detailed by Procedure 3 computes the maximum of the marginal of the utility function $U$ defined in Equation 5.

Procedure 2 $\text{computeFunctionMessage}(x_j, U_i, Q)$: The procedure that a PDA uses for calculating a message from function $U_i$ to variable $x_j$.

**Input:** $x_j$: the destination variable; $Q$: the set of all the messages received by $U_i$ since the last time a message was computed; $U_i$: the sending function.

**Output:** $r_{i \rightarrow j}(\mathcal{S}_j(x_j))$

1: $r_{i \rightarrow j}(\mathcal{S}_j(x_j)) = [-\infty, -\infty]$
2: for all $x_j$ do
3: $\sigma = U_i(x_i)$
4: for all $q_{k \rightarrow j} \in Q$, $x_k \in x_i$; $k \neq j$ do
5: $\sigma = \sigma + q_{k \rightarrow j}(\mathcal{S}_k(x_k))$
6: end for
7: $r_{i \rightarrow j}(\mathcal{S}_j(x_j)) = \max(r_{i \rightarrow j}(\mathcal{S}_j(x_j)), \sigma)$
8: end for
9: return $r_{i \rightarrow j}(\mathcal{S}_j(x_j))$

In order to apply max-sum to our problem, we follow the factor graph representation and implement procedures 1 and 2 within each UAV and Procedure 3 within each PDA. In so doing, the computation is completely decentralised as requested by our requirements. To achieve asynchronicity, each platform activates these procedures periodically. The memory consumption is also very efficient. Indeed, both procedures 1 and 2 require only the last messages received to compute the new messages and Procedure 3 needs again these last messages to make a decision.

IV. EXPERIMENTAL EVALUATION

We run experiments in order to assess the behaviour of our algorithm. Specifically, simulations are used to measure the
performance of the coordination algorithm in a controlled repeatable setting. Hardware tests are then used to evaluate its performance in the real world.

A. Simulations

An empirical evaluation of our utility function is necessary to verify whether, by considering all the constraints of the problems, our utility does not limit the capacity of the coordination mechanism to yield a good performance in terms of the number of completed high priority tasks when compared with less constrained utilities. To achieve this, we benchmark our utility (the Complete utility $C$) against four other utilities, each considering an increasing number of constraints: (i) $U_1^1 = p_i \cdot u_i^{-t_i}$ (ignores battery and duration $IB$&$D$) that removes the constraint related to the duration ($\lambda_i$) and to the UAVs properties (i.e. remaining battery capacity $t_2$ and distance $t_1$); (ii) $U_2^2 = 1 - e^{t_i - h_i}$ (ignore priority and urgency $IP$&$U$) that removes the priority $p_i$, the urgency and the activation time constraint $u_i^{t_i - h_i}$; (iv) $U_3^3 = 1 - e^{t_{\max} - h_i}$ (ignore battery $IB$) that again removes priority and duration, but also the battery capacity ($t_{\max}$ is equal to the initial amount of each UAV’s battery capacity); finally, as a lower bound we benchmark all the former approaches against a local greedy algorithm ($LG$) where each UAV selects which task to attend depending on the utility (v) $U_4^4 = p_i \cdot u_i \cdot t_i \cdot e^{-t_{\max} - h_i}$ that gives a higher value to the closest high priority tasks.

We run 200 simulations, considering a set of fixed independent variables whose values are chosen so as to generate settings likely to happen in the real world. In more detail, we consider teams of 10 UAVs and 30 tasks, their initial position was randomly chosen within an area of 1500 x 1000m. For each UAV $j$: $b_j \in [500x, 1000x]$, $V_j = 5$ m/s and $j$’s communication range is set to 400m. For each task $T_i$: $l_i \in [0, 1000]$; $p_i = 1$ or $p_i = 1000$; $d_i \approx 5$min or $d_i \approx 20$min (note that we only consider two values to emphasize the impact of the priority and the duration on each task’s utility); finally, since both priority and urgency linearly influence the utility, we set the urgency ($u_i = 1.001$). The completion of a task is uncertain and drawn from a Poisson distribution. In each experiment we measure the number of completed tasks over the total number of tasks and the number of completed high priority tasks over the number of completed tasks. Finally, we measure the influence of each utility on the battery capacity of each UAV by recording the average number of time that each UAV runs out of battery capacity while hovering above a task.

Results are shown in Figures 2(b) 2(a) and 2(c). The error bars in the figures represent the standard error of the mean. Figure 2(a) and 2(b) confirm that our utility yields a better trade off between the quality of tasks completed and their quantity. To understand this note that Figure 2(a) shows that the utility that yields the highest average number of completed tasks ratio is $IP$&$U$ (55% of the tasks are completed). This is expected since it represents the less constrained utility, who does not discriminate between tasks and thus allow a higher number of them to be completed. Our utility and $IB$ are more constrained, therefore they complete respectively 50% and 45% of the available tasks. The two remaining utilities ($IB$&$D$ and $LG$) behave greedily and therefore they perform much worse than the other ones.

Now, Figure 2(b) shows that our utility is the one completing in average the highest number of high priority tasks (75% of the completed tasks have a high priority). Indeed, by using $IP$&$U$ only 55% of the completed tasks have a high priority. Thus the use of our utility the UAVs yields 5% less completed tasks than the use of $IP$&$U$. Thus these results show that our utility yields a better trade off between the quality and the quantity of the tasks completed. The other benchmarks perform as expected, by using $IB$, the second most constrained, 65% of the completed tasks have a high priority, whereas the remaining two ($IB$&$D$ and $LG$) utilities yields roughly 50% of high priority completed tasks. This surprisingly high rate is explained considering that the assignment of the priorities is based on a uniform probability (i.e. 50% of the tasks have a high priority and 50% have a low one). Figure 2(c) shows that by using our utility only 4 out of 10 UAVs run out of battery while hovering above a task, whereas by using the other utilities the number varies from 5 to 7. Running out of battery while hovering is a clear symptom of the fact the assignment has been done in a non effective fashion since the battery life of the UAV has not been taken into account. In this sense, the poor performance.
of the utilities not considering the battery life such as LG, IB&D and IB is to be expected, whereas the performance of the two remaining utilities is roughly the same since they both take the UAVs battery into account.

B. Hardware Tests

The aims of running hardware tests is to verify the performance of the coordination mechanism when deployed in the real world. Hence, these tests do not focus on important challenges of real flight tests, such as collision avoidance (the UAVs are flown at separate altitudes, 20m and 40m) and take off and landing (delegated to a human operator) but on validating the behaviour of the coordination algorithm when confronted with the dynamism and the heterogeneity of situations that can happen in the real world.

We demonstrate our system using two commercial off-the-shelf Mikrokopter hexacopter multi-rotor rotary wing UAVs (Figure 3) and two ground-based PDAs. These were deployed on two desktop PCs (Intel Core 2 Duo 3.0GHz, 3.2Gb RAM) connected via ethernet. Figure 4 illustrates the way we built our system. As shown in the figure, a software module containing the algorithm described in Section III is implemented for both UAVs and PDAs. Each decision made by a UAV is transmitted to the corresponding hexacopter wirelessly. Each UAV is provided with a flight control system that provides both attitude stabilisation as well as GPS waypoint-based guidance system to control its motion—it follows a sequence of waypoints representing locations to reach—while holding a pre-determined altitude. A pair of 900MHz radio modems are used to establish a wireless command and control datalink between the ground control software and the UAVs. Each UAV is finally equipped with a downward pointing video camera to capture imagery of the targets (white squares laid on the ground).

Our tests were run at a test facility outside of Sydney, Australia. A video summarising the tests can be found at http://vimeo.com/34800379. In the video (see Figure 5 for a snapshot), windows A and B show the hexacopters, window C shows the computation over the factor graph over which max-sum is running and window D shows the path of the UAVs. We conducted three tests:

Flight 1 – Homogeneous Tasks: Two identical tasks (T₁ and T₂ in Figure 6(a), both have normal priority and urgency, 5 min duration) are simultaneously submitted to the UAVs (UAV₁ and UAV₂ in the figure). The aim of this test is to assess the behaviour of the coordination mechanism in response to a canonical coordination scenario. In this setting, the maximum of each task’s utility is obtained when the task is assigned to the closest UAV (this is due to the exponential factor in Equation 1). Initially, the two UAVs coordinate by sharing max-sum messages with the PDAs. The coordinated decision that maximises the sum of the tasks’ utilities \( U(x) \) is then the one in which each UAV is assigned a single task. Indeed, this is what we observed during our test 6(b) confirming the correctness of our system. Figure 6(c) finally shows the UAVs hovering above their corresponding tasks.

Flight 2 – Sequential arrival of Tasks: Two different tasks (T₁ and T₂ in the figures, T₁ has a normal priority, while T₂ has a high priority, both have normal urgency and 5 min duration) are submitted to the UAVs (UAV₁ and UAV₂ in the figures). T₂ is submitted 40s after T₁. The aim of this to test is to assess the behaviour of the mechanism in the presence of heterogeneous properties and dynamism. Initially, only T₁ is present and the maximum of its utility is obtained when it is assigned to both the UAVs (due to the exponential factor in Equation 1). Note that the utility of assigning the task to both the UAVs cannot be lower than the utility of assigning it to only one of them (i.e. as shown by Equation 1). For this reason, in this case, the UAVs both go to the same task (Figure 6(d)). As soon as T₂ appears, the setting becomes the same as per flight 1. Thus, the maximum of each task’s utility is obtained when the task is assigned to the closest UAV. Thus, the UAVs revise their decisions and UAV₂ goes to complete it (Figure 6(e)). Once T₁ is completed, the setting becomes the same as the beginning of Flight 2. Thus, again the UAVs revise their decisions and are both assigned to the remaining task (Figure 6(f)). Three coordinated decisions then maximise the sum of the tasks utilities \( U(x) \). Initially, the best decision is the one in which both the UAVs are assigned to the only available task. Then, the best decision becomes the one in which each UAV is assigned a single task. Again, this is what we observed during our test. Finally, the best decision is the one in which they are again assigned to the remaining task.
Future work will focus on achieving on board control for the UAVs and on scaling the complexity of our flight tests so as to verify how the systems behave in more realistic situations where the number of submitted tasks is much larger than the number of UAVs.

REFERENCES


V. CONCLUSIONS AND FUTURE WORK

In this paper we introduced a novel coordination approach for teams of UAVs to provide live aerial imagery to the first responders at the scene of a disaster. We casted the problem as one of task assignment in which the UAVs dynamically coordinate over tasks representing the imagery collection requests in a decentralised fashion. To measure the quality of the assignment of one or more UAVs to a task, we derived a novel utility function which incorporate all the constraints of the problem into one single function. We then solved the problem using a fully asynchronous and decentralised implementation of the max-sum algorithm. Next, we empirically showed that our utility yields a better trade off between the quantity and quality of completed tasks than utilities that do not take all the constraints into account. Finally, we deployed our coordination approach on two hexacopters and demonstrated its practical viability on three different scenarios, thus demonstrating that the max-sum is a very good candidate to be used for coordinating unmanned aerial vehicles in real operations.
Fig. 6. A sequence of snapshots depicting the behaviour of the two UAVs in the flight tests.