Autonomous Robots (2018) 42:1601–1618
https://doi.org/10.1007/s10514-018-9712-y

Decentralized planning and control for UAV–UGV cooperative teams

Barbara Arbanas[1] · Antun Ivanovic[1] · Marko Car[1] · Matko Orsag[1] · Tamara Petrovic[1] · Stjepan Bogdan[1]

Received: 15 February 2017 / Accepted: 2 February 2018 / Published online: 15 February 2018

Abstract
In this paper we study a symbiotic aerial vehicle-ground vehicle robotic team where unmanned aerial vehicles (UAVs) are used for aerial manipulation tasks, while unmanned ground vehicles (UGVs) aid and assist them. UGV can provide a UAV with a safe landing area and transport it across large distances, while UAV can provide an additional degree of freedom for the UGV, enabling it to negotiate obstacles. We propose an overall system control framework that includes high-accuracy motion planning for each individual robot and ad-hoc decentralized mission planning for complex missions. Experimental results obtained in a mockup arena for parcel transportation scenario show that the system is able to plan and execute missions in various environments and that the obtained plans result in lower energy consumption.

Keywords Unmanned aerial system · Aerial manipulation · Heterogeneous robotics systems · Decentralized planning

1 Introduction

While the tremendous advances have been made in development and control of single robots for aerial, ground and marine applications, research focus has lately shifted to multi-robot systems, where robots work (move) together to accomplish tasks that would be otherwise unachievable by a single robot. Diverse capabilities of robots in such systems are brought together in order to achieve better performance, broader space coverage, improved energy utilization, and better knowledge through data fusion.

UAV research, together with its respective market, has been growing rapidly thanks to recent technological developments. Unfortunately, mostly due to limited payload capabilities of UAVs, in both research and industry, engineers have focused their efforts to deploy them in surveillance, reconnaissance or search and rescue missions, avoiding all possible interaction with the environment. However, the ability of aerial vehicles to manipulate a target or carry objects and interact with the environment, could greatly expand the application potential of UAVs to: infrastructure inspection (Fumagalli et al. 2014), construction and assembly (Lindsey et al. 2012; Jimenez-Cano et al. 2013), agriculture, urban sanitation, high-speed grasping and payload transportation (Thomas et al. 2014; Sreenath et al. 2013) and many more (Fumagalli et al. 2012; Kim et al. 2013; Scholten et al. 2013).

Our previous research interest focused on aerial manipulation employing a dual-arm manipulator on-board a UAV to yield an unmanned aerial system (UAS). Within this research we concentrated on modeling (Orsag et al. 2014) and control (Korpela et al. 2013) for various aerial manipulation tasks, which include but are not limited to: pick and place, construction and assembly, and perching and manipulating objects (Korpela et al. 2014). However, most quadrotor aerial plat-

This is one of several papers published in *Autonomous Robots* comprising the "Special Issue on Distributed Robotics: From Fundamentals to Applications".

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s10514-018-9712-y) contains supplementary material, which is available to authorized users.

✉ Barbara Arbanas
barbara.arbanas@fer.hr

Antun Ivanovic
antun.ivanovic@fer.hr

Marko Car
marko.car@fer.hr

Matko Orsag
matko.orsag@fer.hr

Tamara Petrovic
tamara.petrovic@fer.hr

Stjepan Bogdan
stjepan.bogdan@fer.hr

[1] Laboratory for Robotics and Intelligent Control Systems (LARICS), Faculty of Electrical Engineering and Computing, Unska 3, 10000 Zagreb, Croatia

forms can be classified as small or micro UAVs, weighing less than 5 kg (Korchenko and Illyash 2013), with the range of less than 10 km, limited payload and time-of-flight capabilities. To put things into perspective, although there have been recent scientific results (Kondak et al. 2014), most commercially available UAVs cannot be deployed in complex aerial manipulation tasks which would normally require lifting heavy objects and long execution time. In order to overcome these limitations, instead of building UAVs with better capabilities, we propose introducing unmanned ground vehicles (UGVs) to aid and assist the UAS in complex aerial manipulation scenarios. In such a system of robots, the UGV provides the UAS with a safe landing area and transportation across large distances, thus saving precious battery life of the UAS.

To achieve the overall goal of executing complex mission scenarios in heterogeneous systems, we propose a decentralized manner of control. In order to do so, we need to tackle both low-level and high-level control, which are disseminated in this paper. Low-level includes mapping and localization, motion planning for both UAS and UGV, landing of UAS onto UGV and picking up lightweight UGV (L-UGV) using UAS. High-level mission planning covers decentralized ad-hoc reasoning about actions that need to be taken in order to execute the missions, and their interrelations within a given environment. For example, during the planning phase, the optimal course of actions for the execution of the mission needs to be determined: whether to use the UAS alone, or if it would be beneficial to utilize the UGVs as well. Hereby we aim to optimize a given criteria, such as mission speed and energy consumption.

Scenario To test the proposed system we imagine a scenario in which a team of robots is deployed to bring back a parcel. The parcel is placed within different cluttered environments. The assumption is that the environment 3D map is given to the planning procedure. Given any known map, the planning algorithm is able to design the system behaviour, but depending on the layout the plan is going to have different degrees of optimality. In order to have a stable localization for each agent, the environment needs to have enough distinctive features for the vision system. For example, a monochrome room is not suitable. Environment examples are industry plants, construction sites, etc. The robots are instructed to negotiate an optimal solution (either with respect to energy consumption or mission duration), containing the set of actions that enable the system to map the environment and retrieve the parcel. Execution of the proposed scenario on selected environments is shown in video material contained within the following playlist (LARICSlab 2017a).

The results presented herein are built upon our previous work (Petrovic et al. 2015) and (Arbanas et al. 2016). In (Petrovic et al. 2015) we have conducted a simulation-based analysis of the proposed system for a simple scenario, which includes a single UAS and a single L-UGV. In (Arbanas et al. 2016) we proceeded to experimentally verify an augmented version of the system in a mockup environment, on a scenario involving two Pioneer UGVs (acting as UAS carriers) and a UAV equipped with dexterous manipulator arms.

Key contributions First, we propose a novel kind of aerial-ground system with a focus on symbiotic behavior that enhances the motion capabilities of individual vehicles. To that end we have designed and constructed L-UGV suitable to work closely with UAV equipped with an aerial manipulator, specifically designed to be carried by the UAS. The two vehicles complement each other, thus forming a symbiotic aerial-ground robot system. Next, high-accuracy trajectory planning and execution algorithms are implemented and tuned for each vehicle. A vision-based localization and mapping algorithm has been implemented for replacing the motion capture system used to control the UAS and UGV. Used together, the proposed trajectory planning and localization algorithms enable our system to fly in narrow corridors (< 1.5 m), while planning trajectories longer then 12 m on-board in real-time.

From a high-level mission planning perspective, we have developed and validated a decentralized hierarchical planning method able to construct and coordinate, in real-time, feasible team plans for any given map of the environment. Planning is modular and able to cope with teams consisting of any number of UASs and UGVs. Our previous work is augmented to allow for mission representation with arbitrary number and arrangement of obstacles detected from the start point to the parcel. In this paper we describe in detail the enhanced planning procedure and show experimental results for a parcel transportation mission conducted in a mockup arena using UAS and a L-UGV.

Paper organization In Sect. 2 we analyze related state-of-the-art work in aerial-ground cooperative systems and decentralized planning. In Sect. 3 we give a brief overview of agents used in the system and their capabilities in a form of high-level behaviors. In Sects. 4 and 5 we describe the developed planning and scheduling algorithm, while in Sects. 6 and 7 we describe hardware and software design, respectively. Experimental validation is given in Sect. 8, together with analysis of energy savings achieved through schedule optimization. The conclusion is given in Sect. 9.

2 Related work

In this Section we review some of the existing work in the area of heterogeneous cooperative teams, in particular UAV-UGV teams, including methods for decentralized mission planning. The majority of the works related to UAV-UGV teams consider cooperation in terms of: collaborative sensing, data fusion and information sharing between UAV and UGV, such as collaborative mapping of the environment (Michael et al. 2012; Papachristos and Tzes 2014), generating maps using

UAV for aerial-ground team navigation (Hsieh et al. 2007), sensor information sharing (Butzke et al. 2016), and target tracking (Dias et al. 2015). Other approaches directly exploit different energy capabilities of aerial and ground vehicles, for example, by deploying micro-UAVs using UGV robots for energy preservation (Mathew et al. 2015), providing refueling stops for UAVs using UGVs (Maini and Sujit 2015). Analogous systems, which are all marsupial in nature, can as well be found in UAV–AUV systems (Miskovic et al. 2014) and UGV–UGV systems (Drenner et al. 2007; Wurm et al. 2013). Our presented system goes a step further by focusing on symbiotic motion relations between UAV and UGV, specifically, UAV equipped with aerial manipulators transports UGV over obstacles, while UGV transports UAV over larger distances. Additionally, UAV provides a map for UGV.

Multi-robot task and motion planning has been studied in the past decades from different perspectives in robotics, control and artificial intelligence research communities. In general, decentralized task-planing can be *static*, where robots use predefined plans or rules for execution (Cirillo et al. 2014), or *dynamic*, where plans are constructed during the task execution and are reactive to the state of the environment (Gerkey and Mataric 2002). Communication among robots can be *explicit*, where exchange of information is done in a dedicated, peer-to-peer manner, and *implicit*, where information from other robots is collected through sensing of the environment (Yan et al. 2013). While static approaches can handle more complex tasks, they are mostly not suitable for real-time reactive applications that require dynamic planning. Our approach is dynamic in the sense that mission plans are constructed online, when a mission is assigned, depending on the state of the environment. We utilize explicit communication over the given wireless network, however, ad-hoc networks can be used for that purpose as well (Pimentel and Campos 2003).

Mission planning consists of two problems—task decomposition (answering the question *what do we do?*), and task allocation (answering *who does what?*) (Zlot and Stentz 2006). One of the most prominent solutions for planning are auction and market based approaches (Gerkey and Mataric 2002; Stentz and Dias 1999). These mostly fit into the task allocation category, where a set of simple tasks is given, and robots use bidding mechanisms to distribute tasks amongst themselves. Application of bidding mechanisms for more complex missions, with loose coupling between tasks, is given in (Zlot and Stentz 2006), however partial ordering between tasks and tight coupling, which are the basis of our cooperative missions, are not well supported.

Probabilistic multi-robot coordination approaches based on usage of decentralized partially observable Markov decision processes have been studied as well (Dias et al. 2015; Omidshafiei et al. 2015). The advantage of this approach is its inherent suitability to uncertain environments, however, the scalability problem is making them unsuitable for real-world applications that include several robots and complex tasks. Some recent approaches focus on lowering computational complexity (Amato et al. 2015).

Many of recent state-of-the art approaches rely on off-the-shelf automated reasoners based on, for example, Linear Temporal Logic (LTL) (Ding et al. 2011; Guo et al. 2014; Raman 2014), Answer Set Programming (ASP) (Saribatur et al. 2014). Even though these approaches show significant contributions to theoretical synthesis of correct-by-design controllers, they too often suffer from intensive computational problems, as well as the inability to quantify planner objectives. In (Wurm et al. 2013) the authors coordinate a marsupial team of robots, using an integrated temporal and cost-based planning approach for the target assignment problem, with PDDL being used as the formal specification language. Other relevant approaches include (Di Paola et al. 2015; Lemaire et al. 2004; Tang and Parker 2005).

As stated in (Yan et al. 2013), challenges in multi-robot coordination include founding multi-layered control structure that combines a high-level planning and coordination with low-level reactive execution and supervision. In this paper we follow this approach, and propose a solution for decentralized multi-robot control. In our solution, we utilize dynamic high-level task planning (decomposition and allocation) based on TÆMS (*Task Analysis, Environment Modeling and Simulation*) language (Lesser et al. 2004; Petrovic et al. 2015), while coordination and supervision is done using *Generalized Partial Global Planning* (GPGP) coordination framework (Decker and Lesser 1995). The advantage of TÆMS is the ability to easily specify complex task interrelations (both tight and loose) and to quantify the potential solutions. GPGP provides a framework for implementation of communication and coordination protocols applied to each individual robot. Further, we are combining high-level motion planning with low-level control and are proposing a method of constructing obstacle-free trajectories for each vehicle. We are using state-of-the-art motion planners that provide cost estimates to the planning module: sample-based planners for UASs and lattice-based for UGVs (Krnjak et al. 2015).

3 Unmanned ground and aerial system

In this work we consider three distinct agents with specific capabilities: a mobile unmanned aerial vehicle with a manipulator, a lightweight ground vehicle and a carrier ground vehicle. Here we describe high-level agent behaviors (actions), while the implementation is given in Sect. 7.

Table 1 List of agent behaviors

Behavior	Description
Takeoff	The UAS goes to the desired height above its current position
Land	The UAS lands below its current position
Move to desired position	This behavior is common for all agents (UAS, L-UGV and C-UGV). The agent moves to the requested position and holds that position until a new request is received
Hold position	This behavior is common for all agents. The agent holds position until a request for a different action is received
Grab L-UGV	Enables visual tracking of the L-UGV, descends and grabs the L-UGV upon successful detection
Release L-UGV	Executes preplanned dual manipulator motion to release the L-UGV and set the manipulator in the soft home position
Grab parcel	Enables visual tracking of the parcel, descends and grabs the parcel upon successful detection
Release parcel	Executes preplanned dual manipulator motion to release the parcel and set the manipulator in soft home position
Land on the L-UGV with the parcel	Enables visual tracking of the L-UGV while parcel is acquired, descends to the L-UGV, releases the parcel and further descends in order to grab the L-UGV

3.1 Unmanned aerial system

UAS is the most versatile of the robot agents in the system. It surpasses the ground vehicles with its four degrees of freedom enabling it to access every section of the environment. The UAS in our work goes beyond the well known and rather simple concept of eye-in-the-sky since it has the ability to physically interact with its surroundings: both the parcel and other agents. The analysis of the UAS capabilities has been carried out by breaking them down into several behaviors, which have been listed in Table 1.

In addition to these listed behaviors, the UAS has the ability to pinpoint and track the target, localize itself and other robots in the environment, as well as build the map of the environment (SLAM). It is a well known fact that keeping rotorcrafts airborne is energy inefficient. This, together with the power requirements of performing all the algorithm computations on-board, makes the UAS flight a very energy-expensive task. To prolong the mission duration, it is vital to utilize the ground agents to carry the UAS and thus conserve energy.

3.2 Autonomous carrier vehicle

In the considered missions we envision a carrier UGV (C-UGV), capable of transporting other agents and equipment necessary to complete the mission. This agent can be equipped with a versatile sensory apparatus and utilities, its task being to provide other robots in the system with the necessary information that allows them to correct their own measurements. Since it has considerable payload and relatively long autonomy time, the C-UGV can also serve as a charging station for the UAS and L-UGV tandem. Providing additional energy increases the duration of the mission and thus facilitates completion of multiple objectives.

The C-UGV can also be remotely operated and guided to the first obstacle in a cluttered environment using feedback from the camera and a map acquired through laser- or stereo vision-based mapping. It serves as the entry point for the operator to gain insight into the mission status. C-UGVs are not used for experiments described in this paper, but we list them here for completeness. Our previous work (Arbanas et al. 2016) shows experiments with C-UGVs, and all methods described here can be applied for missions with C-UGVs as well.

3.3 Lightweight UGV

The idea of this agent is to be light enough that the UAS can pick it up and carry it across obstacles it cannot negotiate on its own. As non-negotiable we consider all obstacles higher than a certain level, which is in this paper set to 2 cm. At the same time it needs to have a strong construction which can withstand the weight of the UAS. Such a vehicle thus needs to be stripped off of all sensors, carrying only the most necessary devices for motor control and communication.

Such a vehicle cannot navigate through the environment on its own because its position error would quickly accumulate. Therefore it relies on the UAS to provide accurate position measurements and, in a way, guide the L-UGV through a cluttered environment. The portable L-UGV is energy efficient, which makes it ideal for carrying the UAS across long distances in the mission, and thus reduce the total energy consumption within the system.

4 Mission representation

This chapter brings a formal definition of mission representation used in our work to express agents' knowledge about the missions set before them. We describe a concept of a global and a local view on the mission structure, and finally detail how this method was applied to the parcel transportation scenarios we have used as a case study.

4.1 Global mission representation

To coordinate the robot behavior, we first decompose each mission into a set of subtasks contained within a tree-like hierarchical structure using TÆMS framework. The mission tree contains action nodes that correspond to real, actionable robot behaviors, and task nodes that combine action nodes into a meaningful structure, as defined by the mission objective. Sets of actions (tasks) are defined as A (T), respectively. Each $a \in A$ can be performed by one or more robots. If we denote the set of robots as $R = \{1, \ldots, n\}$, we can specify the set of actions robot i can perform as A_i, and the set of tasks robot i can contribute to as T_i. The root task corresponds to the mission objective. Notice that redundancy is possible, hence, in general $A_i \cap A_j$ and $T_i \cap T_j$ might not be empty sets, for $i \neq j, i, j \in R$.

TÆMS framework allows for definition of simple and complex relations between tasks, as well as temporal constraints on their execution (Horling et al. 1999). Here we describe relations most relevant to our application scenarios. The most pervasive relation between tasks is a *parent-child* relation, as it fosters task decomposition. If the task $t_a \in T$ involves task (action) $t_b \in A \cup T$ in its execution, we say that task t_a is a parent of the task t_b and denote the said relation as $pc(t_a, t_b)$. Another important relation, mainly due to enforcing precedence constraints on different tasks, is the *enables* interrelationship. For instance, if we want the task $t_a \in T \cup A$ to be executed before task $t_b \in A \cup T$ starts, we can introduce an enables relation between the two as $en(t_a, t_b)$. An example of *enables* relation is given in Fig. 1. Actions $a1$ and $a2$ are related with *enables* relation ($en(a1, a2)$), since the UAS needs to first land carrying an L-UGV ($a1$), and then travel on the L-UGV to the next position ($a2$). If no precedence constraints between tasks exists, such limitations are not included into the model.

To evaluate the tasks, each $a \in A_i$ is assigned a triple (q_a, d_a, c_a), where q_a stands for action quality, d_a for duration and c_a for action cost. Action quality is set a-priori by the system designer. Each robot estimates the duration and cost of a future action based on the current state of the system (other robots and environment) and unit costs associated with the action (E_a). E_a is a parameter specific to each vehicle and can be, for example, power required to execute an action a. The outcome of each task $t \in T_i$, (q_t, d_t, c_t), is determined using the quality accumulation function $Q : T_i \rightarrow \mathbb{R}_0^3$ that describes how subtasks and subactions contribute to the quality of a parent task. Function Q can, in general, have any user-defined form.

In our scenario we use two types of Q: q_{max} and q_{seq_sumall}. The accumulation function q_{max} is used for tasks that are performed using exactly one of its subtasks (analogous to the logical XOR operator on subtasks). For such a task the triple (q_t, d_t, c_t) corresponds to the value of the single subtask that is performed, otherwise (if more than one

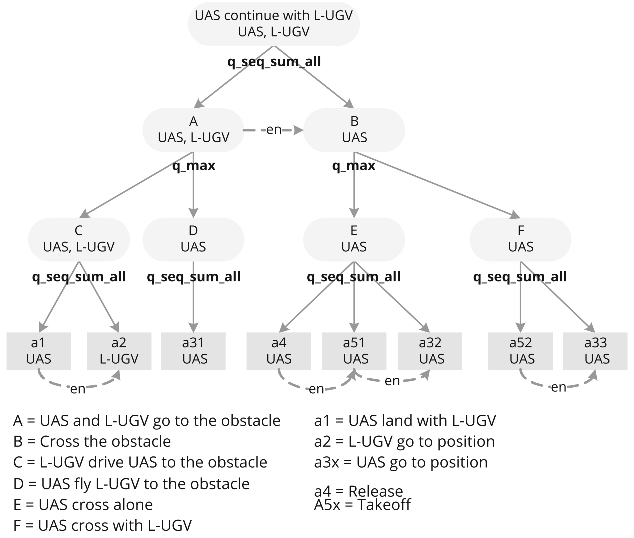

Fig. 1 Example of task structure for obstacle crossing task for UAS and L-UGV agents. Actions are represented by rectangular boxes, while round-corner boxes denote compound tasks. Full arrows illustrate task decomposition in a hierarchical structure, and soft constraints (*enables* relations) are depicted by dashed lines

subtask is performed) it is zero. The function q_{seq_sumall} is used for tasks that are performed using all of its subtasks in a predefined order (similar to the logical AND operator on subtasks, but with implicit precedence constraints). For such a task (q_t, d_t, c_t) corresponds to the sum of subtask values if all of them are performed, otherwise its value is zero. Figure 1 illustrates previously described concepts of task specification on an example of cooperative obstacle crossing task for a robot system comprised of a light-weight UGV and a UAS. To ensure better readability of the structure, all tasks in the figure are assigned shorter labels, with a corresponding description below the figure.

4.2 Local mission representation

Previously described mission structure contains complete information on the way mission can be executed. Even though it is convenient to have such an overview, from agent's perspective, most of the information contained within is unnecessary and overloading. Therefore, a local view on the mission is constructed for each type of agents that can take part in mission execution. Nodes contained within it are actions agent can execute and tasks it can contribute to. In Fig. 1 labels within each node denote to which agent this node is known.

Such an approach allows for modular systems as agents are initially oblivious to who can also participate in the mission execution. The coordination procedure ensures mutual identification of the participating agents and updating of local viewpoints with minimal necessary information needed.

4.3 Application to the parcel transportation mission

The envisioned application scenario includes autonomous UASs and L-UGVs working cooperatively to transport a parcel between two locations in the environment. Inputs to the described planning procedure are a 3D occupancy map of the environment, parcel origin and destination positions, and a local view of the mission tree that describes the parcel transportation scenario (mission).

The mission structure of the described scenario can be rather complex, depending on the number of obstacles in the environment. Here we outline only the main task groups, as the mission tree is too large to be conveniently included in the paper. As Fig. 2 illustrates, at the base of the mission structure resides the *Transport parcel* task, which is further decomposed into a series of *Cross the obstacle* subtasks, followed by *Pick up parcel* and another set of *Cross the obstacle* tasks, ending with *Deliver parcel*. The displayed graph is for a general case of an environment containing N obstacles on the path from the origin point to the location of the parcel. All subtasks are temporally constrained by node to their left,

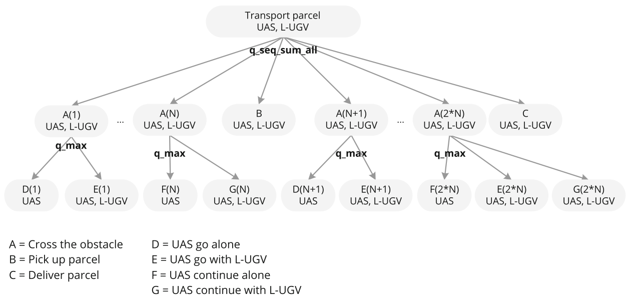

Fig. 2 Mission decomposition of parcel delivery task

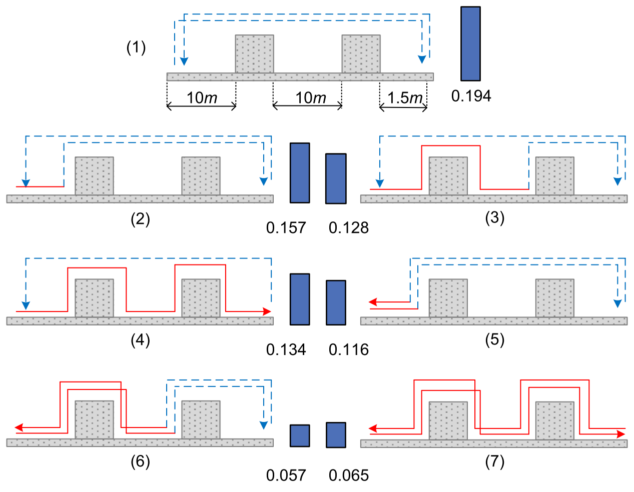

Fig. 3 Various solutions to a parcel delivery mission

thanks to the *seq_sum_all* quality accumulation function of the *Transport parcel* task.

Each *Cross the obstacle* task is split into multiple options of crossing the obstacle, and can be fulfilled only by doing one of them. The decision lies in whether the obstacle is going to be tackled jointly or the UAS is going to cross it alone. If the UAS were to cross any of the obstacles alone, all of the subsequent *UAS continue with L-UGV* tasks in the same mission would not be allowed to be executed until the UAS returns back to the same obstacle where the L-UGV is waiting. To facilitate such a behavior, we extended the mission representation by introducing another relation between tasks, the *excludes* interrelationship. This relation simply lists tasks that can't appear together in the same schedule, and such cases are discarded before schedule generation. For the instance, the graph from Fig. 2 contains following *excludes* relations: $excl(D(1), G(i)), \forall i \in (2, 2 * N))$, where N denotes the number of obstacles. Similarly, *excludes* relations are defined between other tasks that semantically don't belong to the same execution sequence.

The proposed model has proved to produce schedules with ample energy savings. The obtained solutions are given in Fig. 3, where the estimated energy expenditure of each solution is illustrated with a bar next to it, accompanied by the exact value in MJ. Figure 3 shows a symbolic representation of a mock-up arena with two obstacles between the start position (far left) and the parcel position (far right). The path segment lengths are 10, 10 and 1.5 m, starting from the left. The agent average speeds are set to 0.13 m/s, and the energy expenditure is calculated using power consumption for each agent–action pair as provided in Table 5 in Sect. 8.

The first illustration represents a base solution where the UAS executes the whole mission alone and its energy expenditure is estimated to 0.194 MJ. As we can see from Fig. 3, the solution which utilizes the L-UGV on the first two segments in both directions, but the UAS crosses the far right obstacle alone, outperforms every other schedule (illustration (6)). Since the third segment is much shorter than the previous two, it does not pay off to cross the last obstacle carrying the L-UGV. One interesting alternative is the one in illustration (3), where the UAS utilizes the L-UGV to navigate the first two segments, but returns to the starting point alone, which could be beneficial if the L-UGV needs to stay near the goal for further exploration.

5 Mission planning

In this section we describe a method used to generate plans, given a previously defined mission structure.

The previously described mission representation is the backbone of GPGP, a scheduling and coordination framework used in our work. Its main premises are domain independence and a generalized approach to multiagent coordination. During mission specification, special care needs to be taken to properly define each agent's local mission representation, as described in the previous section.

The most prominent feature of GPGP framework is each agent's oblivion regarding other agents' capabilities, which is manifested in having a local view of the task structure. Scheduling and coordination are performed in a decentralized manner, in several steps. The first step is *Update of non-local views*, where agents share their locally estimated task and action outcomes. Coordination relationships may come as a result of constraints on tasks (*enables*, *disables* interrelationships) or *parent-child* relations between tasks. The second step is *Generation of task alternatives*, where potential redundancies are being resolved and the best alternative for mission execution is selected. Given the alternative determined in this step, the *Iterative schedule construction* procedure follows, which obtains the sequence of actions for each agent with respect to temporal constraints between them.

The task alternative $s(t)$, $s(t) \subset A$, $t \in T$ is defined as an unordered set of all actions whose execution leads to the completion of task t. The alternative for the root task of the global mission representation will be denoted as s.

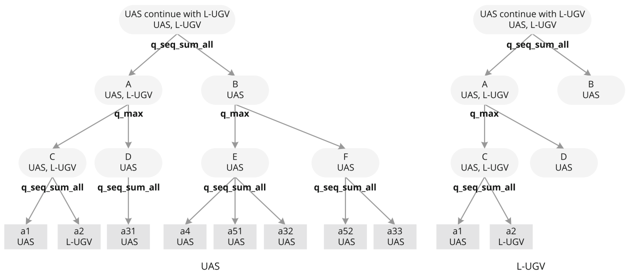

Fig. 4 Updated non-local viewpoints in the obstacle crossing task

The sizes (cardinal numbers) of the task alternative sets for various tasks in the mission plan depend on the structure of the mission task tree, and on relations between the nodes. When missions are highly constrained, the cardinal numbers are low (i.e. $O(1)$). On the other hand, for missions without any node interrelations, the combinatorial explosion can lead to a factorial size complexity of the task alternative set.

Update of non-local views

Since the knowledge of each agent is limited to a local mission representation, it needs to collect necessary information from other agents. To minimize the amount of exchanged information, agents share only estimated execution outcomes given as (q, d, c), thus encapsulating the details behind task execution. Agent i requests from agent j the outcomes of its related actions/nodes as follows:

(i) redundant tasks
$\forall t \in (A_i \cup T_i) \cap (A_j \cup T_j)$ agent i requests from agent j the values (q_t, d_t, c_t)

(ii) child tasks
$\forall t_a \in T_i, \forall t_b \in (T_j \cup A_j)$ s.t. $\exists pc(t_a, t_b)$ agent i requests from agent j the values $(q_{t_b}, d_{t_b}, c_{t_b})$

An example of updating non-local views in the scenario from Fig. 1 has been illustrated in Fig. 4.

Determination of the best task alternatives

At the beginning of the second step, each agent has complete information on the outcome estimates for tasks in its local mission representation. The process of task alternative generation begins at the action nodes of the local TÆMS tree and builds up recursively, finally ending at the root of the tree.

Naturally, each task has many different ways of being realized so the task alternative generation procedure uses a method of focusing the solution search by pruning the worst partial results in each step of the process, thus making the problem tractable. Furthermore, since agents at this point share the same outcome estimates, the same alternative set for common tasks is going to be obtained by each agent. This means that all possible redundancies in task execution have also been resolved during task alternative selection. For example, all the obtained task alternatives for

Table 2 Generated task alternatives for the obstacle crossing task

Agent	Task alternatives
UAS	$C : \{a1, a2\}$, $D : \{a31\}$, $E : \{a4, a51, a32\}$, $F : \{a52, a33\}$
	$A : \{a1, a2\}, \{a31\}$, $B : \{a4, a51, a32\}, \{a52, a33\}$
	UAS continue with L-UGV : $\{a1, a2, a4, a51, a32\}$
	$\{a1, a2, a52, a33\}, \{a31, a4, a51, a32\}, \{a31, a52, a33\}$
L-UGV	$C : \{a1, a2\}$
	$A : \{a1, a2\}, \{D\}$
	UAS continue with L-UGV : $\{a1, a2, B\}, \{D, B\}$

each agent in the scenario from Fig. 1 are given in Table 2. For example, task B (UAV is holding an L-UGV and needs to cross an obstacle) has two alternatives: $\{a4, a51, a32\}$ (release the L-UGV, takeoff and fly across the obstacle alone) and $\{a52, a33\}$ (takeoff and fly across the obstacle with L-UGV).

The score for each task alternative is devised as $sc(q_t, d_t, c_t)$, where q_t, d_t, c_t are calculated from the expected subtask (q, d, c) values. At the end, a single root task alternative s with the best estimated score is chosen to be scheduled. Our simplified objective function (score) is defined as

$$sc(q_t, d_t, c_t) = \alpha q_t - \beta d_t - \gamma c_t, \alpha, \beta, \gamma \in \mathbb{R}, \quad (1)$$

where $\alpha + \beta + \gamma = 1$.

Schedule construction

Given a root task alternative s, the goal of a scheduling algorithm in general is to build a schedule $S = ((a_1, t_1^s, t_1^f), \ldots, (a_n, t_n^s, t_n^f)), a_k \in s$, where t_k^s (t_k^f) denotes the time instance at which the action starts (finishes). Moreover, since each agent is restricted to a local view of the mission, each agent constructs a local schedule S_i where actions of other agents are taken into consideration. The overall schedule S is a superposition of individual agents' schedules.

At this point, each agent knows a set of actions it needs to perform and the interrelations with tasks of other agents. The problem itself is a form of a job-shop. Scheduling is done in an iterative manner according to Algorithm 1.

As given in Algorithm 1, each agent first constructs an initial schedule and sends its *commitments* to other interested agents. Commitments are the base concept in multiagent schedule construction, as each agent has a capability of committing to other agents to execute a particular action by a certain time. Logically, commitments stem from precedence constraints put upon tasks. Formally, every commitment is defined as $comm(t_a \in T_i \cup A_i, end\_time(t_a), t_b \in T_j \cup A_j)$, $i \neq j, i, j \in R$ and is stored in the commitment base of the

```
Data: alternative $s_i$, agent index $i$, mission tree, set of
      neighboring agents $Neigh$
Result: coordinated local schedule $S_i$

$S_i'$ = schedule generated from alternative ($s_i$)
forall the $k \in Neigh \cup \{i\}$ do
    $Compl_k$ = tasks completed by agent $k$ while executing $S_i'$
end
while True do
    forall the $j \in Neigh$ do
        forall the $t_a \in Compl_i$, $t_b \in Compl_j$, $en(t_a, t_b)$ do
            $Comm_j = Comm_j \cup \{comm(t_a, end\_time(t_a), t_b)\}$
        end
        forall the $disables(t_a, t_b)$, $t_b \in Compl_i$, $t_a \in Compl_j$ do
            $Comm_j = Comm_j \cup \{comm(t_b, end\_time(t_b), t_a)\}$
        end
        send $Comm_j$ to agent $j$;
    end
    if $|Comm_j| = 0$, $\forall j \in Neigh$ AND no new commitments
    received then
        $S_i = S_i'$
        break;
    end
    $S_i'$ = schedule generated from alternative ($s_i$)
end
```

Algorithm 1: Scheduling procedure for agent i

agent. This commits agent i to execute task t_a by the time $end\_time(t_a)$ and t_b denotes the constrained task of agent j. In every next iteration, the agent fits its schedule to the received commitments of other agents and its own unmet commitments. The procedure is repeated until all the commitments are met, that is, until cardinal number of the set of agent commitments towards each other agent j, denoted as $|Comm_j|$, is equal to zero.

For large-scale problems, scheduling can be done, for example, by employing a genetic algorithm, with schedule itself acting as a chromosome, population base being comprised of schedules that comply with the provided mission structure, and a fitness function that takes total duration and a number of broken soft constraints into account. For the application scenario considered in this paper, we used a priority based heuristic.

The schedule construction process is done for the previous example as illustrated in Table 3.

Table 3 An example of applying the scheduling procedure in the obstacle crossing scenario from Fig. 1

k	Schedules and commitments
1	UAS: $((a1, 0, 3), (a52, 3, 6), (a33, 6, 8))$
	L-UGV: $((a2, 0, 10))$
	Comm(UAS): $(a1, 3, a2)$
	Comm(L-UGV): $(a2, 10, B)$
2	UAS: $((a1, 0, 3), (slack, 3, 10), (a52, 10, 13), (a33, 13, 15))$
	L-UGV: $((a2, 3, 13))$
	Comm(UAS): $\emptyset$
	Comm(L-UGV): $(a2, 13, B)$
3	UAS: $((a1, 0, 3), (slack, 3, 13), (a52, 13, 16), (a33, 16, 18))$
	L-UGV: $((a2, 3, 13))$
	Comm(UAS): $\emptyset$
	Comm(L-UGV): $\emptyset$

The procedure is carried out in multiple iterations k of the outer *while* loop

6 System design

Before we proceed with analysis of experimental results of the proposed mission planner, in this section we describe the actual hardware and software implementation of the agents, and the capability that they provide to the system as a whole.

6.1 UAS hardware design

UAS aerial robot is built around the *AscTec NEO* hexacopter equipped with an *Intel NUC* onboard computer and a *Skybotix VI-Sensor* for stereo vision-based localization. Such a high-performance onboard computer is needed for computationally complex image processing in stereo vision and trajectory planning algorithms. Furthermore, a dual-arm manipulator with two degrees of freedom is mounted on the UAS to perform the necessary pick and place tasks. The manipulators are designed using *Dynamixel AX-12* servo motors, with the tool at the end of both arms designed to pick up the L-UGV and the parcel.

6.2 L-UGV hardware design

L-UGV was constructed using carbon fiber and fiberglass materials, providing strength and making the vehicle lightweight at the same time. The shape was specifically tailored not to affect the air flow produced by the UAS during pickup. At the same time, the design focused on increasing the possibility of a successful grab. We have tested several designs of similar weight ($\approx$ 450 g), where the design c) in Fig. 5 showed the least impact on the UAS thrust necessary to pick the L-UGV up.

The vehicle carries the bare minimum of sensors, using only motor mounted encoders and sensors available on the UAS to navigate its way across the layout. L-UGV is driven with four FAULHABER micro motors series 1331 with a two-channel IE2-400 magnetic encoder and powered with a small (350 mAh; 7.4 V) LiPo battery. Also, four IR LEDs which are used for finding and tracking the L-UGV are mounted on its top.

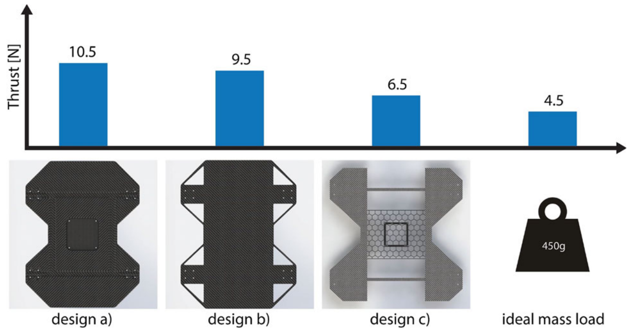

Fig. 5 Considered L-UGV designs and the amount of thrust needed to lift them. For reference, the amount of thrust needed to lift an ideal mass load of 450 g is displayed on the right

A low-level control and sensor-collecting electronic board was designed, built and tested. The board deals only with low level differential drive speed control and power management, as well as communication through an XBee module. Low-level control consists of four angular speed control loops, one loop per motor, and an inverse kinematic model, as shown in Fig. 6. High-level computations are performed on an external CPU (on UAS or a C-UGV) and computed reference values for angular $\omega(t)$ and linear $v(t)$ speed are forwarded to the L-UGV through XBee. A standard inverse kinematic model for a differential drive was used:

$$\omega_L(t) = \frac{v(t)}{R} + \frac{B\omega(t)}{2R},$$
$$\omega_R(t) = \frac{v(t)}{R} - \frac{B\omega(t)}{2R} \qquad (2)$$

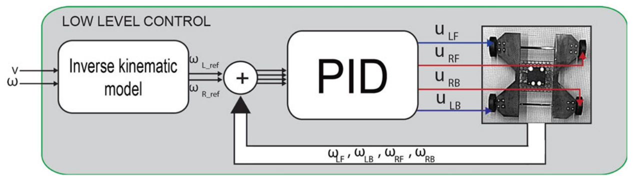

Fig. 6 L-UGV low-level control structure. ω_{L_ref} and ω_{R_ref} are angular speed references for left and right wheels, and $u_{LF}, u_{RF}, u_{LB}, u_{RB}$ are voltage references for the left front, right front, left back and right back motor, respectively

where $\omega_L(t)$ and $\omega_R(t)$ are angular speeds of the left and right wheels, B is the L-UGV width and R is the radius of the wheel.

To navigate the L-UGV through environment, path planning algorithms are run on a parent vehicle using its sensors, obtained maps, and information on the current position of the L-UGV. UAS acts as a parent vehicle in our envisioned system. Path planning algorithm that is used for L-UGV is described in (Krnjak et al. 2015). The proposed approach calculates an obstacle-free path from the current to the desired final pose, based on the use of a state lattice constructed from a set of feasible paths built around the kinematic model of the vehicle. This algorithm can be divided into the state lattice construction process and the search for a path in the resulting constructed lattice. Collision detection takes into account the size of the vehicle, as described in the following sections.

7 Software design

7.1 UAS obstacle-free trajectory

Generation of an obstacle-free trajectory is divided in three stages: path planning in the environment map, initial polynomial trajectory generation, and ensuring that the trajectory remains obstacle free. The collision free path is acquired through RRT* algorithm implemented within OMPL (*Open Motion Planning Library*) (Şucan et al. 2012) in a map represented using *OctoMap* (Hornung et al. 2013), where the UAS is considered to be a box of fixed dimensions. The output of the RRT* algorithm is a set of m waypoints $\boldsymbol{q}_i \in Q$, $\boldsymbol{q}_i \in \mathbb{R}^{4\times 1}, i \in (1, 2, \ldots, m)$, where $\boldsymbol{q}_i = \left[\boldsymbol{p}_i^T \ \psi_i\right]^T$, $\boldsymbol{p}_i = \left[x_i \ y_i \ z_i\right]^T$ where $\boldsymbol{p}_i$ denotes position and ψ_i denotes yaw angle.

Let $\theta(t)$ denote the trajectory position polynomial, k the order of the polynomial and l the derivative order. In this work we consider derivative orders $l \in [1, 4]$. We use 6th ($k = 6$) order polynomials for the first and the last segment, and 5th ($k = 5$) order polynomials for segments in between. The general form of the position polynomials and their derivatives is:

$$\theta^{(l)}(t) = \begin{cases} \sum_{i=0}^{k} a_k t^k, & l = 0 \\ \sum_{i=0}^{k-l} \left[\prod_{j=0}^{l-1}(k-j)\right] a_k t^{k-l}, & l \in [1, 4] \end{cases} \tag{3}$$

The trajectory is optimized until at least one constraint is met on at least one segment. We define a constraints vector $\boldsymbol{\chi}$ that contains the maximal desired speed v_{max}, acceleration a_{max}, angular speed ω_{max} and angular acceleration α_{max}:

$$\boldsymbol{\chi} = \left[v_{max} \ a_{max} \ \omega_{max} \ \alpha_{max}\right]^T \tag{4}$$

The unknown velocities and accelerations are computed by equalizing 3rd derivatives of adjacent segments in a joint waypoint as well as 4th derivatives. Trajectory duration depends on trajectory optimization process.

After the generation step is finished, the initial trajectory is fit within the radial bound ρ. We denote straight line path for each segment with $\lambda_i(t)$, considering that i-th straight line path segment is defined with waypoints $\boldsymbol{q}_i$ and $\boldsymbol{q}_{i+1}$, and the trajectory on each segment with $\theta_i(t)$. To measure maximal distance between piecewise straight line path and the generated trajectory, we employ the Hausdorff distance on each segment of the trajectory:

$$h(\theta, \lambda) = \max_{t\in(0,1)} \min_{\tau\in(0,1)} \|\theta(t) - \lambda(\tau)\| \leq \rho \tag{5}$$

In our case, the Hausdorff distance can be analytically obtained as the distance of a point on the trajectory segment from the line defined with two waypoints:

$$\begin{aligned} \boldsymbol{w}_i(t) &= \theta(t) - \boldsymbol{q}_i \\ \boldsymbol{v}_i &= \frac{\boldsymbol{q}_{i+1} - \boldsymbol{q}_i}{||\boldsymbol{q}_{i+1} - \boldsymbol{q}_i||} \\ h(\theta, \lambda) &= \max_{t\in(0,1)} ||\boldsymbol{w}_i(t) - (\boldsymbol{w}_i(t) \cdot \boldsymbol{v}_i) \cdot \boldsymbol{v}_i|| \end{aligned} \tag{6}$$

If the maximum distance exceeds the threshold ρ, a new point is added at the maximum breach. The second part of bounding the trajectory is through collision checks. If there is a point where the trajectory collides with the environment, a new point is added in the same fashion as for radial bounding. These two steps are an iterative process which finishes when there are no collisions with the environment and the trajectory is radially bounded.

7.2 UAS environment exploration

Since the environment map is not known at the beginning of the mission, we devised an exploration strategy capable of building the map from an unknown environment. The main idea of the exploration task is enabling a pilot to perform this task remotely without crashing into obstacles. The pilot uses available information from the UAS, i.e. the front camera picture, current position, environment map, etc., and a gamepad controller to explore the unknown area. To assist the driver, we employ a potential field based algorithm for obstacle avoidance on occupied space. In our algorithm, the potential field is bounded by the maximum and minimum distance from the UAS.

During exploration, both the 3D map and the projected map are built by mapping obstacles and walls. The mission goal is also acquired in this phase, for example, the point of the inspection, parcel pickup position, or the valve turn position. In the delivery application scenario, the parcel with a known QR code is detected using the UAS onboard camera.

7.3 UAS acquiring waypoints of interest

In order to plan the mission, it is necessary to determine waypoints of interest, which are used as inputs to planning. Waypoints of interest can be, for example, ideal rendezvous points, wall openings, etc. For our scenario waypoints of interest are the spots before and after each obstacle that can be used for UAS landing and takeoff when carrying the L-UGV. They need to be as close to the obstacle as possible, but far enough to ensure a safe UAS flight.

Since the environment map is acquired after the exploration phase and is a priori unknown, a strategy to determine points of interest had to be devised. First, the RRT* path is planned for the UAS from the start to the goal point in order to make sure the UAS can carry out the mission on its own. The start point is considered to be above the landing spot determined at the end of the exploration phase. The goal point is directly above the parcel. Afterwards, the planned path is projected on the ground and the projected map is used in the following steps of the procedure. The algorithm goes along the projected path and searches for obstacles. Upon hitting an obstacle, an initial point of interest is created. This point is then moved backwards along the projected RRT* path until it becomes feasible for the L-UGV and is saved as a takeoff point. The algorithm continues from the initial point of interest and moves along the projected path until the point is no longer colliding with the obstacle. The first point that becomes feasible for the L-UGV is saved as a land point. This procedure is being repeated until it reached the goal point. Since the goal point is directly above the parcel, it can be treated as an obstacle after projecting the path onto a 2D map, which prevents the L-UGV from reaching that point. Thus, we compute the final reachable point for the L-UGV in the same fashion as acquiring the takeoff points.

7.4 UAS visual tracking of L-UGV

To find and track both the parcel and the L-UGV, we designed two algorithms, one based on AR marker tracking (Siltanen 2012) and the other based on tracking the IR LEDs placed on top of the L-UGV. In order to successfully find and pick up the L-UGV, an infrared LED tracking algorithm is designed.

There are four key stages in obtaining the position of the L-UGV: finding the L-UGV on image and creating a ROI (region of interest), finding blobs, removing outliers, and solving the Perspective-4-Point (P4P) problem. *Blobs* are sets of white pixels on image surrounded by black ones. Image sharpening and lens distortion corrections are applied at this point. For further enhancement of the image, we use the Laplacian operator, a second-order image filter (Gonzalez and Woods 2006). The Laplacian is implemented as a convolution between the image and the appropriate kernel. After sharpening the image, closed contours on the image are compared with the known L-UGV contour. Contour matching is performed using a method based on Hu invariants (Bradski and Kaehler 2008). If a contour match is found, a ROI is created around that contour. Restricting the search to the discovered ROI simplifies finding the right combination of blobs, which makes the procedure easier on the CPU. The right combination of blobs is a projection of IR LEDs on the image plane. Let us denote the position of IR LEDs in L-UGV coordinate system with:

$$\boldsymbol{S}_{IR} = \left[\boldsymbol{X}_1^T \ \boldsymbol{X}_2^T \ \boldsymbol{X}_3^T \ \boldsymbol{X}_4^T\right]^T \tag{7}$$

and their projection on the image plane with:

$$\boldsymbol{S}'_{IR} = \left[\boldsymbol{X}_1^{\prime T} \ \boldsymbol{X}_2^{\prime T} \ \boldsymbol{X}_3^{T\prime} \ \boldsymbol{X}_4^{\prime T}\right]^T \tag{8}$$

where $\boldsymbol{X}'_i = \left[x'_i \ y'_i\right]^T$, and $\boldsymbol{X}_i = \left[x_i \ y_i \ z_i\right]^T$, $i \in [1, 4]$, as shown on Fig. 7. The next step is to find set of blob positions $\boldsymbol{S}'_B$ whose area is smaller than ϵ_B. To solve the P4P problem, we have to find the right combination of blobs $\boldsymbol{S}'_{IR}$ from set $\boldsymbol{S}'_B$ in way that $\boldsymbol{X}'_i$ represents the projection of $\boldsymbol{X}_i$.

To do so, the points $\boldsymbol{S}_{IR}$ are sorted in the following way. $\boldsymbol{S}_{IR}$ is transformed to a coordinate system with the origin in the geometric center of the blobs:

$$\boldsymbol{P} = \frac{1}{4}\sum_{i=1}^{4} \boldsymbol{X}_i \tag{9}$$

$$\boldsymbol{S}_{IR}^P = \boldsymbol{S}_{IR} - \boldsymbol{P}. \tag{10}$$

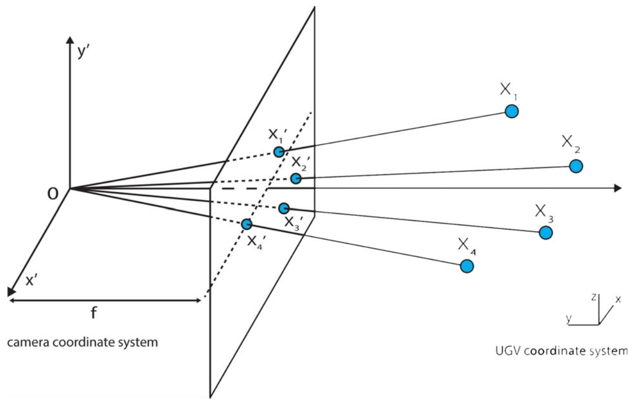

Fig. 7 The camera pose problem illustration. Reference points X_i in the world space, corresponding to four IR LEDs in our problem, are projected to points X_i' in the camera image plane

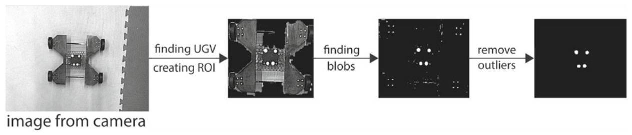

Fig. 8 IR tracking algorithm steps

Next, the angle ϕ_i is calculated as follows and the points $\boldsymbol{S}_{IR}$ are sorted by ϕ_i in a descending order.

$$\phi_i = \arctan \frac{y_i^P}{x_i^P} \qquad (11)$$

where y_i^P and x_i^P are the transformed coordinates of the i-th IR LED from $\boldsymbol{S}_{IR}^P$. To find $\boldsymbol{S}'_{IR}$, every possible combination, $\binom{|\boldsymbol{S}'_B|}{4}$ combinations in total, needs to be sorted as described above and the blob positions compared with the sorted points from $\boldsymbol{S}_{IR}$, using a method based on Hu invariants. Once the match is found, the P4P problem can be solved. The method that we use is an iterative P4P method based on Levenberg–Marquardt optimization (Moré 1978) and can be found in (Bradski and Kaehler 2008). The algorithm steps are illustrated in Fig. 8. In the end, the position of the L-UGV in the camera coordinate system is transformed to the global coordinate system.

To reduce the computational complexity of the algorithm, the position of the ROI can be estimated without searching for the L-UGV contour in every iteration, using the relative UAS motion information between iterations as shown in Algorithm 2. The described algorithm gives an average root-mean-square error of 0.0203 m.

```
Data: image, $\boldsymbol{X}_{UAS}(k)$, $\boldsymbol{X}_{UAS}(k-1)$, $\boldsymbol{S}_{IR}$, camera matrix $C$
Result: $\boldsymbol{X}_{L-UGV}$
// Getting the shift in camera position
$\Delta\boldsymbol{X}'_{UAS} = C \cdot (\boldsymbol{X}_{UAS}(k) - \boldsymbol{X}_{UAS}(k-1))$
// Scaling factor for the ROI
$r_{SF} = z_{UAS}(k)/z_{UAS}(k-1)$
image = laplacianFilter(image)
while true do
    if goFindUGV then
        ROI = findUGV(image)
        $\boldsymbol{S}'_B$ = findBlobs(ROI)
        $\boldsymbol{S}'_{IR}$ = removeOutliers($\boldsymbol{S}'_B$, $\boldsymbol{S}_{IR}$)
        $\boldsymbol{X}_{L-UGV}$ = solvePnP($\boldsymbol{S}_{IR}$, $\boldsymbol{S}'_{IR}$)
        goFindUGV = false
        break
    end
    else
        $ROI = ROI + \Delta X'_{UAS}$
        $ROI.width = ROI.width \cdot r_{SF}$
        $ROI.height = ROI.height \cdot r_{SF}$
        $\boldsymbol{S}'_B$ = findBlobs(ROI)
        $\boldsymbol{S}'_{IR}$ = removeOutliers($\boldsymbol{S}'_B$, $\boldsymbol{S}_{IR}$)
        if $\boldsymbol{S}'_{IR}$ then
            $\boldsymbol{X}_{L-UGV}$ = solvePnP($\boldsymbol{S}_{IR}$, $\boldsymbol{S}'_{IR}$)
            break
        end
        else
            goFindUGV = true
        end
    end
end
```

Algorithm 2: L-UGV tracking algorithm

8 Experimental results

For localization we use stereo visual odometry based on feature selection and tracking (SOFT) (Cvisic and Petrovic 2015). The algorithm uses the *VI-Sensor*, which provides images from two cameras as well as IMU measurements. To be able to correct position a SLAM algorithm is used (SOFT-SLAM) (Cvisic et al. 2017). This algorithm also provides a 3D map of the environment as an *OctoMap*, in which the path and the trajectory are planned for the UAS. However, in order to successfully guide the ground agents through the environment, the 3D map is projected onto ground.

To achieve easier interaction with the mission planner, a series of behaviors for the UAS are developed, as described in Table 1. The described behaviors had to be tuned to our experimental setup. The dimensions of both the parcel and the L-UGV had to be taken into account for pick-up and release behaviors. Landing on the L-UGV while carrying a parcel is a two-stage behavior. In the first stage, the UAS descends above the L-UGV and prepares to release the parcel. Due to the ground effect and other possible disturbances, the UAS has to wait for the right moment to release the parcel. When the drop-off point proximity conditions are satisfied, the UAS releases the parcel, quickly descends further down, and grabs the L-UGV. The action finishes with landing.

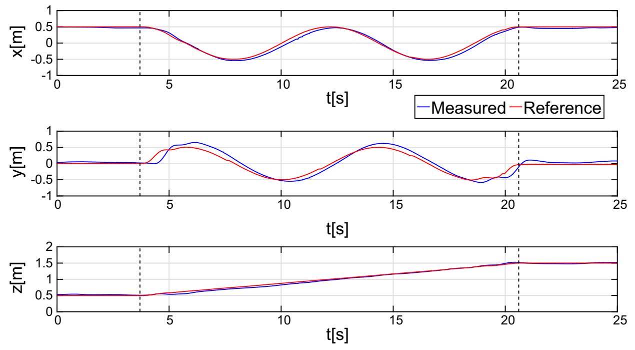

Fig. 9 Generated helical trajectory interpolated with high order polynomials. In this example we set helix radius as $r = 0.5\,\mathrm{m}$ and slope as $b = 0.06\,\mathrm{m/s}$, from which we extracted a set of waypoints

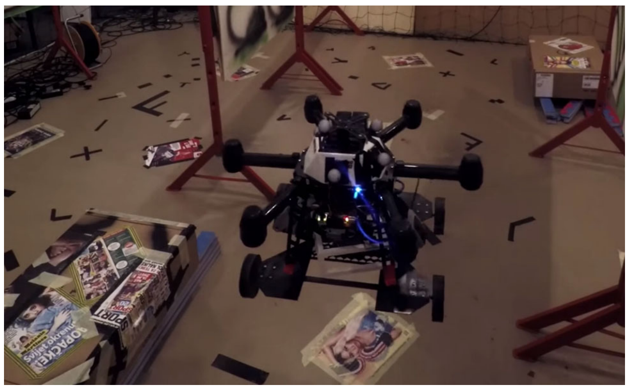

Fig. 10 *AscTec NEO* hexacopter equipped with a dual arm manipulator along with a specially designed lightweight UGV

8.1 UAS trajectory tracking

The trajectory tracking error was analyzed on a helical trajectory example. We predefined a set of waypoints describing a helix and interpolated them with the polynomial trajectory described in Sect. 7.1. The generated trajectory, as well as the UAS position, are shown in Fig. 9. In this case, we set the vector of trajectory speed and acceleration constraints $\boldsymbol{\chi}$ to $\left[1.0\,\mathrm{m/s}\ 2.0\mathrm{m/s}^2\ 1.0\,\mathrm{rad/s}\ 3.0\,\mathrm{rad/s}^2\right]^T$, which yielded the trajectory execution time of 16.8 s. To track the UAS we used *Optitrack* motion capture system and, at this point, an obstacle free space. The average measured RMS error is 0.1526 m.

To be able to autonomously plan and execute a trajectory in a cluttered environment, some precautions had to be made. Since we were flying in narrow corridors, we had to keep a safety distance from the walls, thus in RRT* path planning we consider the UAS to be a box with dimensions $\left[d_x\ d_y\ d_z\right] = \left[1.2\ 1.2\ 0.6\right]$ m. This provides a safety margin of 0.2 m from each side, since the diameter of the UAS is $d_{UAS} = 0.8\,\mathrm{m}$. However, the trajectory can deviate from a piecewise straight line RRT* path as described in Sect. 7, so we are constraining the trajectory within a radial bound $\rho = 0.1\,\mathrm{m}$. There is also no place for aggressive maneuvers in narrow corridors, which can be regulated using constraint vector described in Eq. (4). To safely execute trajectories in narrow corridors we chose conservative set of constraints: $\boldsymbol{\chi} = \left[0.4\,\mathrm{m/s}\ 0.8\,\mathrm{m/s}^2\ 1.0\,\mathrm{rad/s}\ 3.0\,\mathrm{rad/s}^2\right]^T$.

8.2 Mission execution

During experimental verification, we performed several different missions to analyze the planning method efficiency. To complete the missions, we employed an L-UGV—UAS team described in the previous sections and depicted in the Fig. 10. As a baseline mission for energy consumption analysis, we completed the first parcel delivery mission using only the UAS. At the beginning of each mission, we perform exploration of an unknown area in order to map the environment and localize areas of interest. The objective of all missions is to reach the target obtained in the exploration part and execute the necessary actions, e.g. inspect area, pick up parcel, etc.

Table 4 System classification with regard to mass

		C1	C2	C3	C4
UAS	2.7 kg	X	X	X	X
L-UGV	0.4 kg		X		X
parcel	0.1 kg			X	X
		2.7 kg	3.1 kg	2.8 kg	3.2 kg

On the first environment setup we performed three missions of the parcel delivery task, which differ only in employed agents and task decomposition. These experiments were meant to illustrate energy conservation capabilities of our system. Next, we experimented with different environment configurations, where we demonstrate the robustness of the proposed method to variable environments.

We calculate expended energy by integrating the power needed to complete each action over the time taken to complete them. Each action requires different amount of power, due to various combinations of the overall mass, as presented in Table 4.

All the tasks that the system needs to execute are comprised of the three basic behaviors: Takeoff (A1), Land (A2), and Move to desired position—Fly (A3), Drive (A4). Based on the characteristics of the UAV (i.e. total battery capacity $Q_{p,uav} = 8000\,\mathrm{mAh}$, voltage $V_{avg,uav} = 15.8\,\mathrm{V}$ and flight time $t_f \approx 10\,\mathrm{min}$), together with the measured values of the L-UGV, we calculate the power requirements for each action-class pair provided in Table 5.

Table 5 Power needed to execute each action with regard to overall mass

	C1	C2	C3	C4
A1	579 W	711 W	616 W	731 W
A2	566 W	696 W	602 W	729 W
A3	570 W	701 W	606 W	744 W
A4	N/A	N/A	3.3 W	3.3 W

Table 6 Energy expended in different experimental setups

	Energy (J)	Energy saving (%)
Mission 1	542.150	Baseline
Mission 2	343.118	36.7
Mission 3	101.098	81.4

8.2.1 Energy conservation capability

The first environment configuration is provided in Fig. 11. The task of robotic system was to navigate the maze, pick up the parcel at the end of it and deliver it to the starting position. The optimization criteria was to minimize overall energy consumption.

Mission 1

The baseline for first three experiments is a mission with only UAS as an agent. In this mission we obtain the baseline energy consumption, which we aim to reduce in other missions using L-UGV as an auxiliary agent. After exploration, the UAS executes a trajectory to position itself above the parcel, upon which it picks it up and executes a return trajectory.

Mission 2

In this mission we employ the L-UGV. At the beginning of the mission, the UAS rests on the L-UGV, holding onto it. As it is shown in Sect. 4, the optimal schedule with regard to energy conservation is the one which utilizes L-UGV in negotiating the first two sections. UAS releases the L-UGV before the second obstacle and continues the mission alone, as it did in Mission 1. As shown in Table 6, the energy saving utilizing L-UGV is noticeable and amounts to 36.7%.

Mission 3

To further utilize energy saving capabilities of the system, we enabled employment of L-UGV during the return to the start position with the parcel. To be able to do so, we had to provide an action of landing on the L-UGV while carrying the parcel. This modification, although seemingly similar to *Land on UGV* (without parcel) task, showed to be complex to implement due to the UAS that needs to, at the same time, safely release the parcel from a close distance, land on the parcel and grab L-UGV for safe transportation. The resulting schedule resembles the previous one, differing in L-UGV utilization in both directions, as illustrated in Sect. 4. With this schedule, we were able to enhance the energy conservation by another 44.7% of baseline energy, totaling savings to 81.4%.

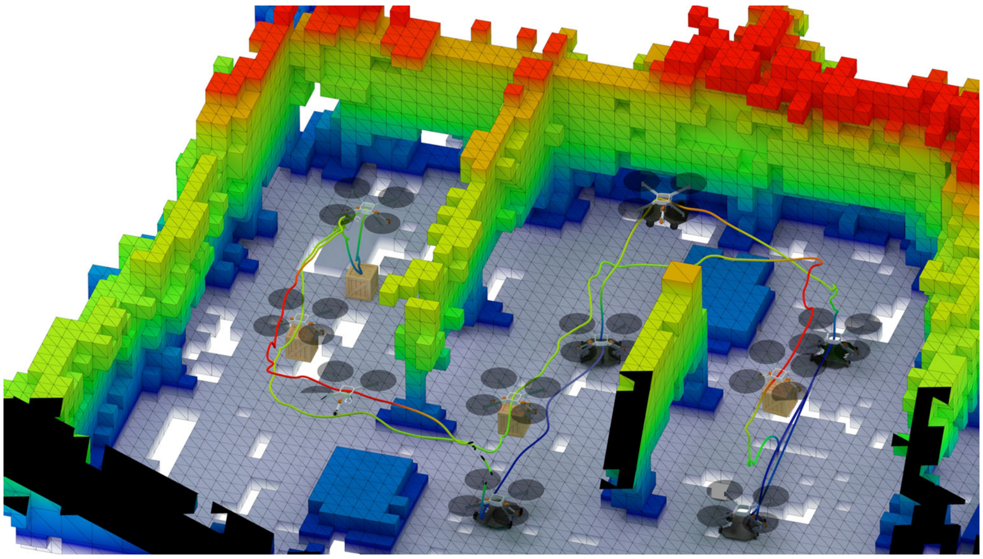

Fig. 11 Experimental result for the parcel delivery task with the L-UGV being utilized in only one direction (Mission 2). Starting position for both agents is located in the lower right corner of the map, while the parcel position is in the upper left corner. L-UGV drives the UAS to the first obstacle, UAS takes off with the L-UGV, crosses the obstacle and lands after it. L-UGV continues to drive with UAS to the next obstacle, after which the UAS takes off without the L-UGV, and continues on its own to pick up and deliver the parcel

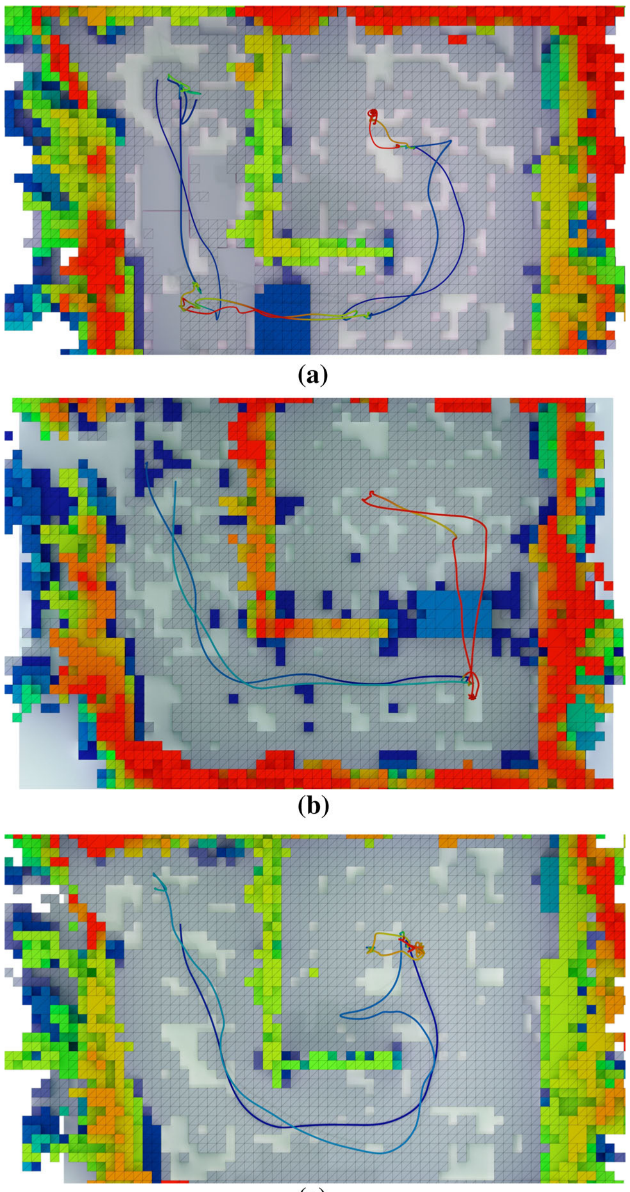

Fig. 12 Experimental results for parcel delivery and inspection tasks on variable environments. Starting position for both UAV and L-UGV is in the upper left corner of the map, while the goal position is on the right side of the map, across the central wall. **a** Mission 4—L-UGV drives the UAS to the obstacle, UAS takes off with L-UGV, flies over it and lands behind it. L-UGV continues to drive with UAS to the goal position where it releases from the L-UGV, takes off and performs the inspection task. After that UAS lands on L-UGV and they continue in the same fashion to return to the starting position. **b** Mission 5—L-UGV drives the UAS to the obstacle, at which UAS releases from the L-UGV, takes off and continues to the goal position on its own. UAS performs the inspection task, flies back to the L-UGV and lands on it. L-UGV drives with the UAS back to the start position. **c** Mission 6—L-UGV drives the UAS to the position near the parcel. UAS takes off, positions itself above the parcel, grabs it and returns to the L-UGV. UAS lands on the L-UGV while holding onto the parcel and L-UGV drives with parcel and UAS to the start position

8.3 Robustness to variable environment

To demonstrate effectiveness of our method in variable environments, we devised three additional environment setups, as shown in Fig. 12. Missions 4 and 5 are structurally the same as Missions 1 and 2, but they omit the parcel delivery. They instead perform inspection (exploration) of the area of interest. Finally, in Mission 6, we completed the same objective as in Mission 3.

All of the software used in this paper is available in the repository (LARICSlab 2017b) which contains ROS packages for planning and low-level control, maps and ROS bags obtained during experiments.

9 Conclusion

In this paper we have proposed and tested a decentralized task planning and coordination framework for systems of multiple robots (UAVs and UGVs) with different capabilities. Inputs to the planning procedure are the map of the environment and a hierarchical decomposition of system mission, both of which are obtained in the initial (exploration) phase of mission execution. Outputs of the sampling-based path planners are used for costs estimation in the planing procedure, as well as input for determination of feasible, obstacle-free trajectories.

The approach was tested experimentally using an UAS built around an *AscTec NEO* hexacopter equipped with a dual arm manipulator along with a specially designed lightweight UGV, dubbed L-UGV. In order to localize we deployed a stereo visual odometry algorithm based on feature selection and tracking. The algorithm uses a *VI-Sensor* which provides images from two cameras as well as IMU measurements. Together with the proposed trajectory planning and control it enables our system to fly in cluttered environments with narrow corridors (< 1.5 m).

Presented results show efficiency of the planning method with regard to energy consumption in mission execution, saving up to 80% of energy consumed during benchmark execution. We also present robustness of the method with respect to different environmental setup, and different system configuration. In future work we aim to test the approach for more complex missions and environments, such as systems of multiple UAVs and multiple UGVs with focus on interchanging team behavior. We are going to consider broadening of UAV-UGV mission spectrum to include more power-intensive tasks, for example, using a power-tethered link connecting UGV and UAV (Zikou et al. 2015). Moreover, we aim to extend the algorithms to include a human agent into mission planning and execution.

Acknowledgements This paper was supported by the EU-FP7-ICT project European Robotics Challenges (EuRoC), Grant Agreement No. 608849.

References

Amato, C., Konidaris, G., Cruz, G., Maynor, C. A., How, J. P., & Kaelbling, L. P. (2015). Planning for decentralized control of multiple robots under uncertainty. In: *IEEE International Conference on Robotics and Automation (ICRA)* (pp. 1241–1248).

Arbanas, B., Ivanovic, A., Car, M., Haus, T., Orsag, M., Petrovic, T., & Bogdan, S. (2016) Aerial-ground robotic system for autonomous delivery tasks. In: *IEEE International Conference on Robotics and Automation (ICRA)* (pp. 5463–5468).

Bradski, D. G. R., & Kaehler, A. (2008). *Learning Opencv* (1st ed.). Sebastopol: O'Reilly Media Inc.

Butzke, J., Gochev, K., Holden, B., Jung, E. J., & Likhachev, M. (2016). Planning for a ground-air robotic system with collaborative localization. In: *IEEE International conference on robotics and automation (ICRA)* (pp 284–291).

Cirillo, M., Pecora, F., Andreasson, H., Uras, T., & Koenig, S. (2014). Integrated motion planning and coordination for industrial vehicles. In: *Proceedings of the 24th international conference on international conference on automated planning and scheduling* (pp. 463–471).

Cvisic, I., & Petrovic, I. (2015). Stereo odometry based on careful feature selection and tracking. In: *European conference on mobile robots (ECMR)* (pp. 1–6).

Cvisic, I., Cesic, J., Markovic, I., & Petrovic, I. (2017). Soft-SLAM: Computationally efficient stereo visual SLAM for autonomous UAVS. *Journal of Field Robotics*.

Decker, K. S., & Lesser, V. R. (1995). Designing a family of coordination algorithms. In *Proceedings of the 1st international conference on multi-agent systems (ICMAS-95)* (pp. 73–80).

Di Paola, D., Gasparri, A., Naso, D., & Lewis, F. L. (2015). Decentralized dynamic task planning for heterogeneous robotic networks. *Autonomous Robots*, *38*(1), 31–48.

Dias, A., Capitan, J., Merino, L., Almeida, J., Lima, P., & Silva, E. (2015). Decentralized target tracking based on multi-robot cooperative triangulation. In *IEEE International Conference on Robotics and Automation (ICRA)* (pp. 3449–3455). IEEE.

Ding, X. C., Kloetzer, M., Chen, Y., & Belta, C. (2011). Automatic deployment of robotic teams. *IEEE Robotics Automation Magazine*, *18*(3), 75–86.

Drenner, A., Janssen, M., Carlson, C., & Papanikolopoulos, N. (2007). Design, control, and simulation of marsupial systems for extending operational lifetime. In *European Control Conference (ECC)* (pp. 3146–3152).

Fumagalli, M., Naldi, R., Macchelli, A., Carloni, R., Stramigioli, S., & Marconi, L. (2012). Modeling and control of a flying robot for contact inspection. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 3532–3537).

Fumagalli, M., Naldi, R., Macchelli, A., Forte, F., Keemink, A., Stramigioli, S., et al. (2014). Developing an aerial manipulator prototype: Physical interaction with the environment. *IEEE Robotics Automation Magazine*, *21*(3), 41–50.

Gerkey, B. P., & Mataric, M. J. (2002). Sold!: Auction methods for multirobot coordination. *IEEE Transactions on Robotics and Automation*, *18*(5), 758–768.

Gonzalez, R. C., & Woods, R. E. (2006). *Digital image processing* (3rd ed.). Upper Saddle River, NJ: Prentice-Hall Inc.

Guo, M., Tumova, J., & Dimarogonas, D.V. (2014). Cooperative decentralized multi-agent control under local LTL tasks and connectivity constraints. In: *53rd IEEE conference on decision and control* (pp. 75–80).

Horling, B., Lesser, V., Vincent, R., Wagner, T., Raja, A., Zhang, S., Decker, K., & Garvey, A. (1999). *The TAEMS white paper*.

Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., & Burgard, W. (2013). OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, *34*, 189–206.

Hsieh, M. A., Cowley, A., Keller, J. F., Chaimowicz, L., Grocholsky, B., Kumar, V., et al. (2007). Adaptive teams of autonomous aerial and ground robots for situational awareness. *Journal of Field Robotics*, *24*(11–12), 991–1014.

Jimenez-Cano, A., Martin, J., Heredia, G., Ollero, A., & Cano, R. (2013). Control of an aerial robot with multi-link arm for assembly tasks. In: IEEE international conference on robotics and automation (ICRA) (pp. 4916–4921).

Kim, S., Choi, S., & Kim, H.J. (2013). Aerial manipulation using a quadrotor with a two dof robotic arm. In *IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 4990–4995).

Kondak, K., Huber, F., Schwarzbach, M., Laiacker, M., Sommer, D., Bejar, M., & Ollero, A. (2014). Aerial manipulation robot composed of an autonomous helicopter and a 7 degrees of freedom industrial manipulator. In *IEEE international conference on robotics and automation (ICRA)* (pp. 2107–2112). IEEE.

Korchenko, A., & Illyash, O. (2013). The generalized classification of unmanned air vehicles. In: *Actual problems of unmanned air vehicles developments proceedings (APUAVD)* (pp. 28–34). IEEE.

Korpela, C., Orsag, M., Pekala, M., & Oh, P. (2013). Dynamic stability of a mobile manipulating unmanned aerial vehicle. In *IEEE ICRA* (pp. 4922–4927).

Korpela, C., Orsag, M., & Oh, P. (2014). Towards valve turning using a dual-arm aerial manipulator. In *IEEE/RSJ international conference on intelligent robots and systems (IROS 2014)* (pp. 3411–3416). IEEE.

Krnjak, A., Draganjac, I., Bogdan, S., Petrovic, T., Miklic, D., & Kovacic, Z. (2015). Decentralized control of free ranging AGVS in warehouse environments. In *IEEE international conference on robotics and automation (ICRA)* (pp. 2034–2041).

LARICSlab. (2017a). https://goo.gl/0J1hmK.

LARICSlab. (2017b). https://goo.gl/cZMrHQ.

Lemaire, T., Alami, R., & Lacroix, S. (2004). A distributed tasks allocation scheme in multi-uav context. In: Proceedings of the ICRA'04 IEEE international conference on robotics and automation, 2004 (Vol. 4, pp 3622–3627).

Lesser, V., Decker, K., Wagner, T., Carver, N., Garvey, A., Horling, B., et al. (2004). Evolution of the GPGP/TÆMS domain-independent coordination framework. *Autonomous Agents and Multi-Agent Systems*, *9*(1–2), 87–143.

Lindsey, Q., Mellinger, D., & Kumar, V. (2012). Construction with quadrotor teams. *Autonomous Robots*, *33*(3), 323–336.

Maini, P., & Sujit, P. (2015). On cooperation between a fuel constrained UAV and a refueling UGV for large scale mapping applications. In International conference on unmanned aircraft systems (ICUAS) (pp. 1370–1377). IEEE.

Mathew, N., Smith, S. L., & Waslander, S. L. (2015). Planning paths for package delivery in heterogeneous multirobot teams. *IEEE Transactions on Automation Science and Engineering*, *12*(4), 1298–1308.

Michael, N., Shen, S., Mohta, K., Mulgaonkar, Y., Kumar, V., Nagatani, K., et al. (2012). Collaborative mapping of an earthquake-damaged building via ground and aerial robots. *Journal of Field Robotics*, *29*(5), 832–841.

Miskovic, N., Bogdan, S., Nad, E., Mandic, F., Orsag, M., & Haus, T. (2014). Unmanned marsupial sea-air system for object recovery. In 22nd Mediterranean Conference of Control and Automation (MED) (pp. 740–745).

Moré, J. J. (1978). The Levenberg-Marquardt algorithm: Implementation and theory. In G. A. Watson (Ed.), *Numerical analysis*. Lecture notes in mathematics (Vol. 630, pp. 105–116). Berlin, Heidelberg: Springer.

Omidshafiei, S., Agha-Mohammadi, A. A., Amato, C., & How, J. P. (2015). Decentralized control of partially observable markov decision processes using belief space macro-actions. In *IEEE International Conference on Robotics and Automation (ICRA)* (pp. 5962–5969).

Orsag, M., Korpela, C., Bogdan, S., & Oh, P. (2014). Hybrid adaptive control for aerial manipulation. *Journal of Intelligent and Robotic Systems*, *73*(1–4), 693–707.

Papachristos, C., Tzes, A. (2014). The power-tethered UAV-UGV team: A collaborative strategy for navigation in partially-mapped environments. In *Mediterranean conference on control and automation* (pp. 1153–1158).

Petrovic, T., Haus, T., Arbanas, B., Orsag, M., & Bogdan, S. (2015). Can UAV and UGV be best buddies? Towards heterogeneous aerial-ground cooperative robot system for complex aerial manipulation tasks. In: 12th International conference on informatics in control, automation and robotics (ICINCO) (Vol. 01, pp. 238–245).

Pimentel, B.S., Campos, M.F.M. (2003). Cooperative communication in ad hoc networked mobile robots. In *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS 2003) (Cat. No. 03CH37453)* (Vol. 3, pp. 2876–2881).

Raman, V. (2014). Reactive switching protocols for multi-robot high-level tasks. In *IEEE/RSJ international conference on intelligent robots and systems (IROS 2014)* (pp. 336–341).

Saribatur, Z., Erdem, E., & Patoglu, V. (2014). Cognitive factories with multiple teams of heterogeneous robots: Hybrid reasoning for optimal feasible global plans. In *IEEE/RSJ International conference on intelligent robots and systems (IROS 2014)* (pp. 2923–2930).

Scholten, J., Fumagalli, M., Stramigioli, S., & Carloni, R. (2013). Interaction control of an uav endowed with a manipulator. In *IEEE International conference on robotics and automation (ICRA)* (pp. 4910–4915)

Siltanen, S. (2012). *Theory and applications of marker-based augmented reality*. VTT Science.

Sreenath, K., Michael, N., Kumar, V. (2013). Trajectory generation and control of a quadrotor with a cable-suspended load-a differentially-flat hybrid system. In *IEEE ICRA* (pp. 4888–4895). IEEE.

Stentz, A., & Dias, M. B. (1999). *A free market architecture for coordinating multiple robots*. DTIC Document: Tech. rep.

Şucan, I.A., Moll, M., Kavraki, L. E. (2012). The open motion planning library. *IEEE Robotics & Automation Magazine, 19*(4), 72–82. http://ompl.kavrakilab.org.

Tang, F., Parker, L.E. (2005). Asymtre: Automated synthesis of multi-robot task solutions through software reconfiguration. In *Proceedings of the IEEE International Conference on Robotics and Automation* (pp. 1501–1508).

Thomas, J., Loianno, G., Sreenath, K., Kumar, V. (2014). Toward image based visual servoing for aerial grasping and perching. In *IEEE ICRA* (pp. 2113–2118).

Wurm, K., Dornhege, C., Nebel, B., Burgard, W., & Stachniss, C. (2013). Coordinating heterogeneous teams of robots using temporal symbolic planning. *Autonomous Robots*, *34*(4), 277–294.

Yan, Z., Jouandeau, N., & Cherif, A. A. (2013). A survey and analysis of multi-robot coordination. *International Journal of Advanced Robotic Systems*, *10*(12), 399.

Zikou, L., Papachristos, C., Alexis, K., & Tzes, A. (2015). Inspection operations using an aerial robot powered-over-tether by a ground vehicle. In *International symposium on visual computing* (pp. 455–465).

Zlot, R., & Stentz, A. (2006). Market-based multirobot coordination for complex tasks. *The International Journal of Robotics Research*, *25*(1), 73–101.

Barbara Arbanas received the B.Sc. and M.Sc. degrees in computer science from the University of Zagreb, Faculty of Electrical Engineering, Zagreb, Croatia in 2015 and 2013, respectively. She is currently employed at the Faculty of Electrical Engineering and Computing, University of Zagreb, as a research associate and a Ph.D. student. Her interests include multi-robot coordination and planning, distributed artificial intelligence, scheduling and optimization. She is currently involved in EU funded projects subCULTron and EuRoC.

Antun Ivanovic received his M.S.E.E. in 2015 and B.S.E.E. in 2013 at the University of Zagreb, Croatia. He is currently employed at the Faculty of Electrical Engineering and Computing, University of Zagreb, as a research associate and a Ph.D. student. His main areas of interest are aerial robotics, motion planning and autonomous systems. In his undergraduate years he received University of Zagreb Rector award for the best student projects in 2014.

Marko Car received his M.S.E.E. in 2015 and B.S.E.E. in 2013 at the University of Zagreb, Croatia. He is currently employed at the Faculty of Electrical Engineering and Computing, University of Zagreb, as a research associate and a Ph.D. student. His main areas of interest are aerial robotics, control theory and autonomous systems. In his undergraduate years he received University of Zagreb Rector award for the best student projects in 2014.

Matko Orsag received his Ph.D. in 2015, M.S.E.E. in 2010 and B.S.E.E. in 2008 at the University of Zagreb, Croatia. He is currently employed at the Faculty of Electrical Engineering and Computing, University of Zagreb, as an assistant professor where his teaching duties include both undergraduate and graduate programmes. His main areas of interest are autonomous systems, robotics and intelligent control systems. During his undergraduate years he received several student awards, as well as a 2008. bronze plaque "Josip Loncar" for overall outstanding academic achievement. He also received the University of Zagreb Rector Award for the best student projects in 2006. He is a coauthor of multiple conference and journal papers.

Tamara Petrovic received the M.Sc. and Ph.D. degrees in electrical engineering from the University of Zagreb, Faculty of Electrical Engineering, Zagreb, Croatia in 2007 and 2014, respectively. Since 2008 she has been a Research and Teaching Assistant with the Laboratory for Robotics and Intelligent Control Systems (LARICS). She participated in several EU funded projects (EC-SAFEMOBILE, subCULTron, ASSISIbf). Her areas of research are discrete event systems, multi-agent control and coordination.

Stjepan Bogdan received his Ph.D.E.E. in 1999, M.S.E.E. in 1993 and B.S.E.E. in 1990 at the University of Zagreb, Croatia. Currently he is an associate professor at the Faculty of Electrical Engineering and Computing, University of Zagreb. His main areas of interest are discrete event systems, intelligent control systems, and autonomous systems. He is a coauthor of three books and numerous papers published in journals and proceedings. He serves as associate editor of IEEE Transactions of Automation Science and Engineering, Journal of Intelligent and Robotic Systems, Transactions of the Institute of Measurement and Control and Journal of Control Theory and Applications.