Can UAV and UGV Be Best Buddies?
Towards Heterogeneous Aerial-ground Cooperative Robot System for Complex Aerial Manipulation Tasks

Tamara Petrovic, Tomislav Haus, Barbara Arbanas, Matko Orsag and Stjepan Bogdan
University of Zagreb, Faculty of Electrical Engineering and Computing, Unska 3, 10000 Zagreb, Croatia

Keywords: Path Planning, Task Scheduling, UAV, UGV, Aerial Manipulation.

Abstract: This paper presents the results of our efforts to build a heterogeneous robotic system capable of executing complex disaster response and recovery tasks. We aim to explore high level task scheduling and mission planning algorithms that enable various types of robots to cooperate together, utilizing each others strengths to yield a symbiotic robotic system. In the proposed scenario, a ground vehicle and an aerial robot work together to close a valve in a disaster stricken industrial environment. To that end we use TÆMS framework in order to specify interrelationships between mission subtasks and develop an effective scheduling and coordination mechanism, inspired by Generalized Partial Global Planning. We present simulation results with two different outcomes that show cooperative capabilities of the system.

1 INTRODUCTION

In recent years we have witnessed a tremendous rise of research potential in the field of unmanned aerial vehicles (UAVs). Consequently, the worldwide UAV market grows rapidly as well. Unfortunately, mostly due to UAV's limited payload capabilities, in both research and industry, engineers focused their efforts to deploy UAVs in surveillance, reconnaissance or search and rescue mission, 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 (Jimenez-Cano et al., 2013; Lindsey et al., 2012), agriculture, urban sanitation, high-speed grasping and payload transportation (Thomas et al., 2014), (Sreenath et al., 2013) and many more (Kim et al., 2013; Scholten et al., 2013; Fumagalli et al., 2012).

In our research we are particularly interested in aerial manipulation employing a dual-arm manipulator on-board a UAV, thus completing a mobile manipulating unmanned aerial vehicle (MMUAV). In our previous work we have focused our efforts to model the influence of the dual-arm manipulator to the change of the center of mass and moment of inertia(Orsag et al., 2014), and to devise a necessary stability criteria for dexterous aerial manipulation (Korpela et al., 2013). We have been able to utilize our approach on 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).

According to the UAV classification (Korchenko and Illyash, 2013), most quadrotor aerial platforms designed so far can be categorized as small or micro UAVs. This implies that the vehicle weight is less than 5kg, is capable of flying below 150m altitude and has the range of less than 10km, capable of carrying merely surveillance equipment with limited time-of-flight capabilities. To put things into perspective, for all sense and purposes, todays UAVs cannot be deployed in complex aerial manipulation tasks which would normally require lifting heavy objects and long execution time. In order to solve this problem, we propose introducing an unmanned ground vehicle (UGV) to aid and assist the UAV in such complex manipulation scenarios. The two vehicles complete each other, thus forming a symbiotic aerial-ground robot system. In such a system, the UGV provides the UAV with a safe landing area and transports it across large distances, thus saving its battery life, and preserving it for other tasks. On the other hand, the UAV can provide additional degree of freedom for the UGV, enabling it to negotiate obstacles by simply lifting it across them. In our previous work, we have

Petrovic T., Haus T., Arbanas B., Orsag M. and Bogdan S..
Can UAV and UGV Be Best Buddies? - Towards Heterogeneous Aerial-ground Cooperative Robot System for Complex Aerial Manipulation Tasks.
DOI: 10.5220/0005548302380245
In *Proceedings of the 12th International Conference on Informatics in Control, Automation and Robotics* (ICINCO-2015), pages 238-245
ISBN: 978-989-758-122-9

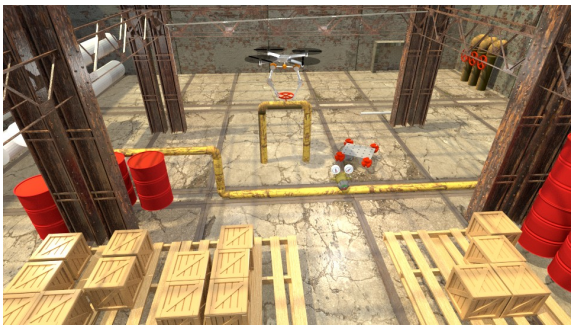

Figure 1: Heterogeneous symbiotic robotic system deployed on a mission to close a valve in a disaster stricken industrial environment.

developed a similar concept for an unmanned surface marine vehicle (USV) and UAV marsupial system that works together to recover objects floating at the sea surface (Miskovic et al., 2014). Of course, controlling a heterogeneous team of robots, like the one proposed herein, requires precise, fast and reliable high level planning and task allocation algorithm, which is the backbone of this paper.

Mission planning in robotic teams has been intensively studied in different research communities. In (Wurm et al., 2013) authors use PDDL specification language and integrate temporal planning approach with cost-based planner for target assignment problem, and apply their approach to marsupial team of robots. In (Raman, 2014) authors use Linear Temporal Logic (LTL) for specification of high-level goals and provide a framework for synthesis of reactive controllers. More examples of LTL-based approach are disseminated in (Ding et al., 2011) and (Saha et al., 2014), where it is utilized for safe decentralized collision avoidance. Although this approach allows one to synthesize provably correct controllers, it often suffers from intensive-computational problems, and the inability to quantify planner objectives. In this paper we consider a specific application domain with two autonomous vehicles of different motion capabilities required to cooperate in a common obstacle-dense environment to achieve a high-level mission goal. For mission specification we use TÆMS framework, which allows us to specify complex relations between mission subtasks and to quantitatively express the contribution of each subtask to the overall mission plan. We develop an effective scheduling and coordination mechanism, which is inspired by Generalized Partial Global Planning (GPGP) (Lesser et al., 2004) framework. The mechanism yields a framework for mission planning algorithms as well as the implementation of communication and coordination protocols applied to each individual vehicle.

The paper is structured as follows. In Section 2 we describe the problem of distributed mission planning for the UAV-UGV system. In Section 3 we present TÆMS specification framework and in Section 4 we describe developed coordination algorithm. Section 5 shows simulation results, and Section 6 gives a brief conclusion.

2 PROBLEM DESCRIPTION

One potential use case scenario for the proposed system is shown in Fig. 1, where a MMUAV is deployed in order to close a valve in a disaster stricken industrial environment. There are three prerequisites needed to complete such a complex assignment: 1) a reliable and stable control system for aerial manipulation, 2) successful self-localization and mapping in three dimensional space and 3) fast planning and task allocation algorithms, which enable collaboration between ground and aerial vehicles. So far, we have successfully tackled the first prerequisite, and demonstrated performance capabilities of our dual arm unmanned aerial system in a mockup laboratory environments on multiple benchmark tasks: pick and place (Fig. 2(b)); construction and assembly (Fig.2(c)); and perch and manipulate objects (Fig.2(a)). In recent years, the robotics community concentrated a lot of effort into solving the second prerequisite, showing very promising results, and therefore allowing our research to focus on the high level planning algorithms that are disseminated in this paper.

2.1 Preliminaries

With aerial manipulation capabilities, the set of skills of a MMUAV grows from simple 4DOF positioning and surveillance to real environment interaction. The ability of a UAV to pick up a UGV demonstrates the symbiotic nature of the proposed heterogeneous robot system, allowing the UGV to fly over obstacles which it normally would not be able to negotiate. On the other hand, in order to prolong the battery life, the UAV can land and perch onto the UGV, relying on the ground vehicle to carry it as close as possible to the target. To define a high-level mission of the system, we first abstract the capabilities of the MMUAV and UGV into the set of behavior primitives:

2.2 MMUAV Behavior Primitives

- *Motion Primitive*
  Motion primitive $M_{UAV}(p_i, p_j)$ consists of planning and execution of an obstacle-free path p from

(a) Valve turning (b) Pick and place (c) Construction

Figure 2: Aerial manipulation benchmark tasks.

point $p_i \in SE(3)$ to $p_j \in SE(3)$ in a given obstacle cluttered 3D environment. This motion primitive is executed whenever theUAV is flying through the environment; this includes flying, landing on the UGV or valve, take-off, turning the valve, as well as carrying the UGV over obstacles. The cost of motion primitive execution, $C(M_{UAV}(p_i, p_j)) \in \mathbb{R}_0^+$, is defined as $C(M_{UAV}(p_i, p_j)) = length(p) \cdot E_{UAV}$, where E_{UAV} corresponds to UAV energy consumed per unit of distance. The value of E_{UAV} depends on the UAV characteristics, and whether the UAV is carrying the UGV or not.

- *Manipulation Primitives*
  Manipulation primitives consist of planning and executing a particular UAV action using solely robotic hands attached to the UAV. The manipulation primitives used in our application are $P(p_i)$ - perch on the UGV at position p_i, $R(p_i)$ - release the UGV and $G(p_i)$ - grip valve. Each manipulation primitive is associated with a cost $C(^*) \in \mathbb{R}_0^+$.

2.3 UGV Behavior Primitives

- *Motion Primitive*
  Motion primitive $M_{UGV}(p_i, p_j)$ is analogous to the UAV motion primitive. It consists of planning and execution of an obstacle-free path p from point $p_i \in SE(2)$ to $p_j \in SE(2)$, where SE(2) state space corresponds to the ground (floor) of the given 3D environment. This motion primitive is executed whenever the UGV is driving through the environment using its own power and, therefore, does not include the UGV being carried over an obstacle by the UAV. The cost of motion primitive execution, $C(M_{UGV}(p_i, p_j)) \in \mathbb{R}_0^+$, is defined as $C(M_{UGV}(p_i, p_j)) = length(p) \cdot E_{UGV}$, where E_{UGV} corresponds to UGV energy spent per unit of distance. The value of E_{UGV} depends on the UGV characteristics.

2.4 Problem Definition

Even though ideas and methods presented in this paper apply to a wider range of applications and tasks, we elaborate our approach using a specific high-level mission. This mission requires the UAV to approach and turn a valve, whose position in the given 3D environment is known prior to execution. In practice this would require a lightweight UAV system to enter the area and build a map of the environment, which can be later used as a starting point for the proposed robotic system.

- *Mission Schedule*
  Let B_{UAV} (B_{UGV}) denote the set of the UAV (UGV) behaviors described in the previous section extended with zero-cost idle behaviors $I_{UAV}(I_{UGV})$. Let $p_0^{UAV} \in SE(3)$ ($(p_0^{UGV}) \in SE(2)$) be the initial positions of vehicles in the environment, and p_v the valve position. The high-level mission schedule consists of ordered sequences of UAV and UGV behaviors including expected start and end times, $(b_{UAV}^1, t_1, t_2), (b_{UAV}^2, t_2, t_3), \ldots, (b_{UAV}^n, t_{n-1}, t_n)$ and $(b_{UGV}^1, t_1', t_2'), (b_{UGV}^2, t_2', t_3'), \ldots, (b_{UGV}^n, t_{n-1}', t_n')$, such that a sequential execution of individual behaviors results in the execution of the high-level mission goal. Derived mission schedule serves as a coordination framework during missions execution, even though actual execution times may differ from the initial plan. Mission schedule is feasible if UAV (UGV) sequence has no discontinuities in space and time, and if the behaviors that are active at a certain time instance are achievable in the real system.

- *Decentralized Mission Planning Problem*
  Given the map of an environment, motion primitives and a high-level mission determine a decentralized method, which runs on each vehicle separately, each responsible for the construction of individual schedules to achieve the overall mission plan. Furthermore, determine a coordination protocol that ensures a robust mission execution.

Example 1
Consider an autonomous system with a UAV and a UGV, operating in an environment given in Fig. 3. The high-level goal is to approach and turn the valve V located at position p_V. Each robot has a knowledge of the environment map, and plans and executes an obstacle-free path. Fig. 3 shows that the mission can be executed in several different ways. First, the UAV can complete this task alone by executing the following behavior: position above the valve ($M_{UAV}(p_{UGV}, p_V)$), grip the valve ($G(p_V)$) and turn the valve ($T(p_V)$). Other options include cooperation with the UGV. One possible solution is that the UAV lands on the UGV ($M_{UAV}(p_{UAV}, p_{UGV})$, travels on it towards the valve (I_{UAV} and $M_{UGV}(p_{UGV}, p_v)$), then grips the valve ($G(p_V)$) and turns it ($T(p_V)$). Depend-

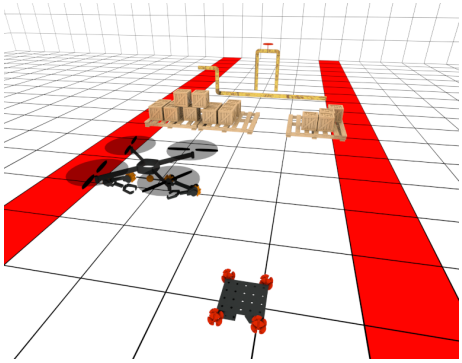

Figure 3: Figure shows the obstacle environment used during mission planner, which is the derivative of the full mockup environment from Fig. 1. The cluttered environment consists of box-like obstacles, a pipeline and the goal valve, with red lines representing map limits.

ing on the environment, one possible solution could be to land on an UGV and relying on it to carry it over obstacles. Given the assumption that each robot has only a partial knowledge of the mission, our goal is to develop a method that would, in a decentralized manner, determine one of such solutions.

3 SPECIFICATION FRAMEWORK: TÆMS

Specifying mission and its tasks in a hierarchical manner is a common approach in the coordination of a group of agents. This approach requires the existence of a hierarchical task network, where each task is broken down into a number of subtasks required to complete it. Since the agents are heterogeneous, each has a number of specific skills that allow the agent to perform a particular subtask of the task network. Moreover, each vehicle is aware of only a subset of tasks in the hierarchy, namely, tasks it can perform alone, and direct predecessors/successors of these tasks. The challenge is then to determine which part of the task network will be executed by which agent in time. This whole process takes place without the presence of central intelligence management, which places special emphasis on the interaction between agents through the exchange of common objectives and action plans.

The language for specifying agents skills and tasks, which we adopt in this paper, is known as TÆMS (Task Analysis, Environment Modeling and Simulation Language) language (Chen and Decker, 2004). TÆMS is a language designed to represent hierarchical task networks that contain complex relations between individual tasks. As we will demonstrate in the following sections, the most distinctive characteristic of TÆMS is that it allows the mission designer to quantitatively specify the capabilities of each individual agent and the correlation between different agents' tasks. The TÆMS hierarchy tree consists of nodes, which represent tasks and actions involved in the achievement of a common goal. Root level node corresponds to the high-level system goal.

- *Action Nodes*

  At the bottom of the hierarchy are action nodes (marked as rectangles). These nodes are unique since only they correspond to actionable tasks of an agent, while the purpose of task nodes is to organize action nodes into a meaningful structure.

  The specification of an action node contains information relevant to the action execution. Each action node is described quantitatively with its *duration* and *cost*. Duration is the time required to execute an action and is used for scheduling individual actions. All values are a-priori expectations for a specific action execution.

  In the studied UAV-UGV system action nodes correspond to behavior primitives. Cost of motion primitives is estimated on-line as length of the path obtained using a path planner, and depends on the current vehicle positions and the environment. When estimating the cost of the UAV carrying the UGV over an obstacle based on the length of the path to the *first* obstacle on UGV's path. The remain of the path to the valve is estimated as an average of UAV and UGV path lengths.

- *Task Nodes*

  While action nodes represent actions that an agent can perform in reality, task is an abstract action that agent performs as a series of atomic actions. The primary purpose of a task is to combine its subtasks in a structure that provides information on how the subtasks can be combined to accomplish the task alone. This parent-child relation is modeled using *quality accumulation function*,

- *Quality Accumulation Function*

  QAF defines how the quality of subtask performance affects the overall quality of the parent task and is reciprocal to the cost of the task. TÆMS structure defines many possible parent-child relations: $(q_{min}, q_{sum}, q_{seq\_min})$. We here describe the ones used for UAV-UGV system.

  - q_{max} - equivalent to the logical operator OR. The overall quality of the task is equal to the maximum quality of its subtasks. Therefore, to accomplish the task it is possible to perform only one subtasks with quality greater than zero.

– q_{sum_all} - equivalent to the logical operator AND, and equals to the sum of subtask qualities if all subtasks can be performed, otherwise its value is zero.

- *Interrelationships*

  QAF describes parent-child relations, while relation between tasks are defined in TÆMS as a separate set of interrelationships. Interrelationship that is used in our application is *enable*, which demands that one task is finished before another is started.

4 COORDINATION AND SCHEDULING

The coordination algorithm run on each robot is divided into several stages. Agents run their respective algorithms independently, but we ensure, using appropriate communication protocols, that agents are always in the same stage of the coordination process. The presented procedure can be used for more than two independent agents. In the text that follows we describe each stage of the coordination process separately:

1. **Init**

   Coordination is started when high-level goal 'Open valve' is broadcasted. Agents involved in execution of 'Open valve' task, respond by broadcasting tasks contained in their view of TÆMS tree. Next, a DTC (*Design to criteria*) scheduler is called, which assesses agents actions, and creates, for each task in agent's TÆMS structure, all alternatives for task execution and their qualities. An alternative is a set of actions whose execution may lead to execution of 'Open-valve' task. The best alternative is chosen for further procedure.

2. **Evaluate Non-local Tasks.**

   Each agent needs to identify tasks of other agents that affect its own performance. These are parent-child coordination relationships (*PC*), determined as: $PC_a = \{(x_2,b)|x_1 \in T_a, x_2 \in T_b, x_2 \textit{ is child of } x_1\}$, where a,b individual agents and T_a, T_b agents' task sets.

   For each element of PC_a, agent a requests and receives from agent b quality of child task. If more than one agent can execute a particular non-local task, mean value of the outcome is taken into account. Initial timed schedule is then created. An example of initial schedule is: ((Position above UGV, 0, 0), (Land on UGV, 0, 2), (UAV stand still, 2, 2), (Perch, 2, 3), (Position above UGV, 3, 4), (Land on UGV, 4, 6), (Release, 6, 7), (Q stand still, 7, 7), (Position above UGV, 7, 8), (Grip valve, 8, 18), (Turn left, 18, 33)).

3. **Resolve Redundant Tasks.** At this stage we solve redundancy issues, if such exist. Redundancy is a situation where a task can be executed using more than one agent, and is scheduled for execution in the initial schedule. In that case, one of the involved agents is chosen as a referee, who communicates with other agents and chooses the best one. Other agents discard redundant tasks from their alternative. Described UAV-UGV system does not have redundant tasks.

4. **Update Schedule.** At this stage, each action is assigned to a dedicated agent, and individual schedules are updated accordingly. Afterwards, each agent identifies hard interrelationships: tasks (x_1) that enable tasks (x_2) of other agents. Agent makes a commitment to the other agent to perform its task until time T, estimated from its initial schedule.

5. **Create Final Schedule** At this stage, agent constructs a timed schedule using information about its own commitments, as well as commitments of other agents to it. If a schedule that satisfies commits cannot be found, agent updated its commitments again, and schedules its plan again. This procedure is repeated until a feasible schedule is found. Resulting schedule for the previous example for UAV is: ((Position above UGV, 0, 0), (Land on UGV, 0, 2), (UAV stand still, 2, 2), (slack, 2, 17), (Perch, 17, 18), (Position above UGV, 18, 19), (Land on UGV, 19, 21), (Release, 21, 22), (UAV stand still, 22, 22), (Position above UGV, 22, 23), (Grip valve, 23, 33), (Turn left, 33, 48)), and for UGV: ((UGV stand still, 0, 0), (slack, 0, 2), (UGV drive to position, 2, 17), (UGV stand still, 17, 17), (slack, 17, 22), (UGV drive to location, 22, 23)). Denote that slack behavior corresponds to the execution of a task of the other agent.

6. **Coordination and Execution** Agents start executing tasks based on their schedules. If any of the agents cannot achieve its commitments in given time tolerance, or executes it earlier, schedules of each agent is adjusted accordingly.

It should be noted here that the solution obtained using this approach always in form of a sequence of UAV/UGV behaviors (actions) where each action occurs only ones. More complex solutions, which include diverse sequences of vehicle behaviors and reactive behavior, can be achieved by repeating the

scheduling procedure depending on the state of the environment.

5 SIMULATION RESULTS

5.1 Simulation Testbed Description

The chosen use case scenario has been implemented in the *Gazebo* simulator using its *Robot Operating System* (ROS) interface. We developed all algorithms in the Python programming language and we utilized ROS as a communication middleware. One should note that ROS multimaster extensions, such as *FKIE Multimaster*, support decentralized implementation of the algorithms, which is a necessary option for the proposed system.

As a UAV in the Gazebo simulator, we use a realistic multicopter model, whose capabilities are enhanced with a dual robotic arm mounted on-board. The model is equipped with a standard sensory set - inertial measurement unit (accelerometer and gyroscope) and a generic pose sensor (e.g. VI-sensor), which gives the copter's position and orientation. The estimated data is used as feedback in the low-level control algorithms, comprised of a classical PID cascade control structure. For high level controller, we use a mission planner comprised of several parts - a navigation algorithm (path planning and trajectory generation), a coordination algorithm and an action server. We utilize ROS *actionlib* stack to establish a client-server communication between the coordination algorithm and the navigation algorithm. The action server acts as a bridge - it receives actions from the coordination algorithm and triggers their execution by calling appropriate services of the navigation algorithm. The action server also publishes the estimated percentage of the completion of an ongoing action and notifies when an action is fully completed.

For the path planning task within the copter's navigation algorithm, we utilize The Open Motion Planning Library (OMPL) (Şucan et al., 2012). From a wide pallet of algorithms that this library offers, we have chosen Optimal Rapidly-Expanding Random Trees (RRT*). The output of the algorithm is a set of 3D points that, if followed, lead from the start point to the goal. Once the set is received, we generate trajectories from path segments given by RRT*. To that end, we employ a simple linear interpolation between two consecutive points that define a path segment, taking into account the copter speed limits.

To simulate a UGV in the Gazebo simulator, we use the model of a holonomic vehicle, equipped with 4 omnidirectional wheels. The vehicle's control architecture is similar to the copter's structure. On the low level, we use 3 PID algorithms to control vehicle's lateral position, longitude position and heading. The output of a generic pose sensor is used as feedback. The high level algorithms are basically identical to the copter's algorithms - the same coordination algorithm with its own TÆMS structure is employed to generate actions, the OMPL library and linear interpolation are used to generate trajectories in 2D and, finally, an action server is used to connect the algorithms.

The simulation of the use case scenario has been executed on a single computer. However, these algorithms and the software architecture allow for a completely decentralized implementation on several dedicated computers, which will be employed in the experimental verification of the system in a realistic laboratory testbed.

5.2 Simulation Results

We have validated the complete system by running extensive tests in a simulation testbed, shown in Fig. 1. The testbed offers several different possibilities of how to reach the valve by the cooperative robotic system from its initial point. To explore these possibilities, we executed simulations with several different sets of mission planning parameters. In particular, the cost of copter's and omnibot's travelled unit distance has been altered, which eventually resulted in different mission schedules generated by the coordination algorithm.

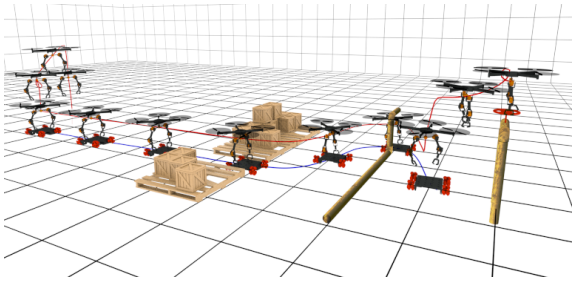

Figure 4: Results for mission planning parameters: $E_{UAV} = 15$, $E_{UGV} = 1$, $E'_{UAV} = 150$.

The most obvious solution is the one in which the copter flies over the obstacles all the way from the start point to the goal. Due to the simplicity of this outcome, we omit the simulation results. The second outcome, shown in Fig. 4, includes the copter landing on the omnibot, followed by the omnibot driving and carrying the copter all the way from the start to the goal. This outcome is achieved for costs of regular driving per unit distance of $E_{UAV} = 15$, and $E_{UGV} = 1$,

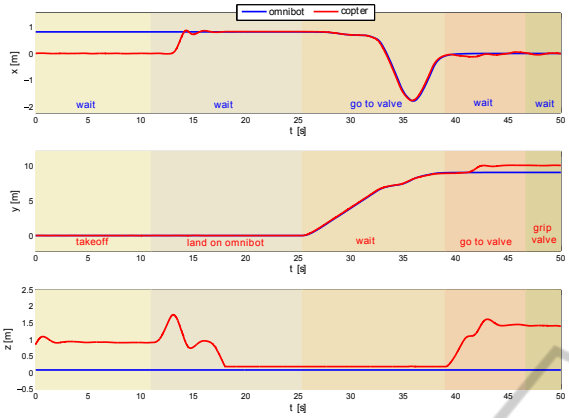

Figure 5: Position of the copter and omnibot during the mission execution, in which the UGV carries the UAV all the way to the valve.

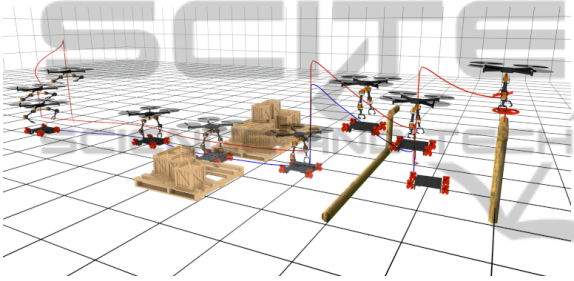

Figure 6: Results for mission planning parameters: $E_{UAV} = 15$, $E_{UGV} = 1$, $E'_{UAV} = 20$.

and cost of flying with UGV of $E'_{UAV} = 150$. The third outcome, shown in Fig. 6, is comprised of the sequence in which the copter lands on the omnibot, the omnibot drives and carries the copter to the front of the nearest obstacle, and the copter lifts and carries the omnibot across the obstacle. Eventually, the copter completes the mission by approaching the valve and turning it. This outcome is achieved with a decreased cost of carying UGV, $E'_{UAV} = 20$. Although this outcome was achieved through parameter tuning, a similar effect could be noted, if for instance the distance between the pipeline and the valve would increase.

To extend our simulation analysis, in Fig. 5 we show the trajectories of both robotic units for the second mission outcome. This is repeated in Fig. 7 for the third outcome. Each figure is plotted with respect to time, clearly marking specific time frames of the mission (i.e. takeoff, grab valve, land on omnibot, etc.).

Even though simulations show stable and robust coordination based on the specified criteria, results can be further improved if we allow mission rescheduling during execution, that would account for

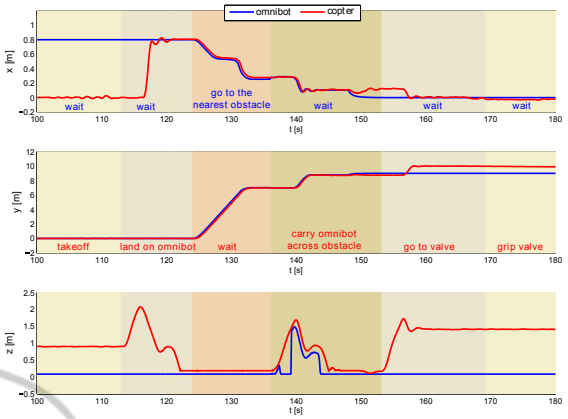

Figure 7: Position of the copter and omnibot during the mission execution, in which the UGV carries the UAV to the front of the nearest obstacle on its way to the valve, the UAV then lifts the UGV across the obstacle and finishes the mission by turning the valve.

changes in the environment and more complex sequences of behavior primitives.

6 CONCLUSION

In this paper we present the concept of a heterogeneous cooperative robotic system comprised of a MMUAV and a holonomic UGV. The driving function for such a system design is the idea to utilize various capabilities of the MMUAV and UGV to complete the mission of turning the valve with minimal energy consumption in an imagined disaster stricken environment. The model of the system and the simulation testbed are implemented in the Gazebo simulator using its ROS interface. The MMUAV and UGV are controlled by a low-level cascade of PID controllers. Furthermore, a mid-level navigation algorithm generates obstacle-free trajectories in 3D space, that are fed to the low level controllers. Finally, for high level mission control, we utilize TÆMS framework to specify a mission and its tasks in a hierarchical manner. We developed a decentralized coordination algorithm, based on the GPGP framework, whose two instances are executed - the first instance gives commands to the MMUAV and the second instance generates commands for the UGV. A negotiation between these two instances is introduced which ultimately gives a sequence of interconnected tasks assigned to the MMUAV and UGV in a decentralized fashion. Through extensive simulation tests we have shown how different cost values, used in mission planning phase, result in different mission schedules, which all eventually lead to a successful execution of the mission.

In the future work, we plan to experimentally validate the system on a mockup laboratory testbed. Furthermore, we will expand the robotic system with several different UAVs and UGVs and deploy it to an unknown environment to perform a set of complex tasks, such as finding a gauge, simultaneously reading the gauge and turning a valve to control the pressure of a pipeline, finding and connecting an electric plug, etc.

ACKNOWLEDGEMENTS

This work was supported in part by the Air Force Research Laboratory, under agreement number FA8655-13-1-3055 and the European Community Seventh Framework Program under grant No. 285939 (ACROSS). The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes not withstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory or the U.S. Government.

REFERENCES

Chen, W. and Decker, K. (2004). Managing multi-agent coordination, planning, and scheduling. In *Autonomous Agents and Multiagent Systems, 2004. AAMAS 2004. Proceedings of the Third International Joint Conference on*, pages 1360–1361.

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

Fumagalli, M., Naldi, R., Macchelli, A., Carloni, R., Stramigioli, S., and Marconi, L. (2012). Modeling and control of a flying robot for contact inspection. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 3532–3537.

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

Jimenez-Cano, A., Martin, J., Heredia, G., Ollero, A., and Cano, R. (2013). Control of an aerial robot with multi-link arm for assembly tasks. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 4916–4921.

Kim, S., Choi, S., and 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*, pages 4990 – 4995, Tokyo, Japan.

Korchenko, A. and Illyash, O. (2013). The generalized classification of unmanned air vehicles. In *Actual Problems of Unmanned Air Vehicles Developments Proceedings (APUAVD)*, pages 28–34. IEEE.

Korpela, C., Orsag, M., and Oh, P. (2014). Towards valve turning using a dual-arm aerial manipulator. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 3411–3416. IEEE.

Korpela, C., Orsag, M., Pekala, M., and Oh, P. (2013). Dynamic stability of a mobile manipulating unmanned aerial vehicle. In *IEEE International Conference on Robotics and Automation*, pages 4922–4927.

Lesser, V., Decker, K., Wagner, T., Carver, N., Garvey, A., Horling, B., Neiman, D., Podorozhny, R., Prasad, M., Raja, A., Vincent, R., Xuan, P., and Zhang, X. (2004). Evolution of the gpgp/tms domain-independent coordination framework. *Autonomous Agents and Multi-Agent Systems*, 9(1-2):87–143.

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

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

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

Raman, V. (2014). Reactive switching protocols for multi-robot high-level tasks. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 336–341.

Saha, I., Ramaithitima, R., Kumar, V., Pappas, G. J., and Seshia, S. A. (2014). Automated composition of motion primitives for multi-robot systems from safe ltl specifications. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Scholten, J., Fumagalli, M., Stramigioli, S., and Carloni, R. (2013). Interaction control of an uav endowed with a manipulator. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 4910–4915.

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

Şucan, I. A., Moll, M., and Kavraki, L. E. (2012). The Open Motion Planning Library. *IEEE Robotics & Automation Magazine*, 19(4):72–82.

Thomas, J., Loianno, G., Sreenath, K., and Kumar, V. (2014). Toward image based visual servoing for aerial grasping and perching. In *IEEE International Conference on Robotics and Automation*, pages 2113–2118.

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