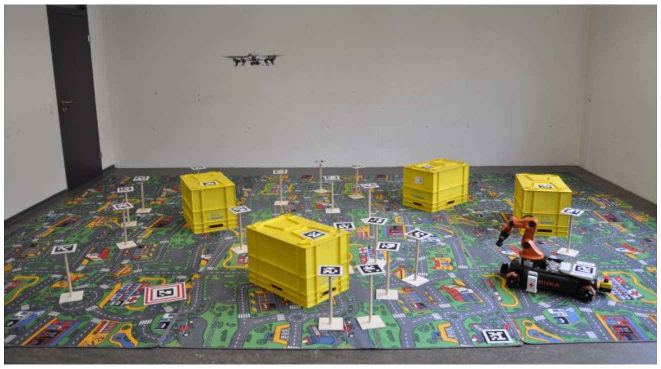
Aerial-guided Navigation of a Ground Robot among Movable Obstacles

Elias Mueggler, Matthias Faessler, Flavio Fontana and Davide Scaramuzza

***Abstract*— We demonstrate the fully autonomous collaboration of an aerial and a ground robot in a mock-up disaster scenario. Within this collaboration, we make use of the individual capabilities and strengths of both robots. The aerial robot first maps an area of interest, then it computes the fastest mission for the ground robot to reach a spotted victim and deliver a first-aid kit. Such a mission includes driving and removing obstacles in the way while being constantly monitored and commanded by the aerial robot. Our mission-planning algorithm distinguishes between movable and fixed obstacles and considers both the time for driving and removing obstacles. The entire mission is executed without any human interaction once the aerial robot is launched and requires a minimal amount of communication between the robots. We describe both the hardware and software of our system and detail our mission-planning algorithm. We present exhaustive results of both simulation and real experiments. Our system was successfully demonstrated more than 20 times at a trade fair.**

Multimedia Material

Demonstration of our system:
`http://www.youtube.com/watch?v=C5I190lzDdQ`
Presentation at the AUTOMATICA'14 trade fair in Munich:
`http://www.youtube.com/watch?v=OFPv3BegbFg`


(a) Our robots operating in a mock-up disaster site

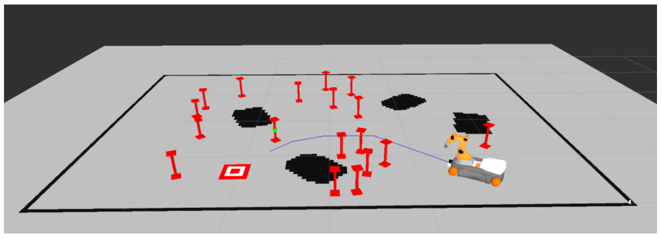

(b) Corresponding mission plan

Fig. 1: After mapping the area, the aerial robot is guiding the ground robot to the goal location. All paths are blocked by obstacles, some of which can be removed by the ground robot.

I. Introduction

Since 2001, rescue robots have been deployed for disaster response and have been used at least 29 times to date [1]. For example, after the earthquake and tsunami in Fukushima, Japan in 2011, ground robots were utilized to explore the situation in the contaminated reactor building [2].

In all previous disaster response missions, the robots were remote-controlled by trained professionals. Three operators per robot were required on average and the executed missions took very long (cf. [1]). Since time is the most critical factor in rescue missions, we propose to deploy teams of *heterogeneous* robots, namely ground and aerial robots, to speed up disaster response. Their sense-act capabilities are complementary: ground robots can carry high payloads and manipulators. However, their field of view is limited and they can be blocked by obstacles on the ground. Aerial robots, in contrast, can overcome obstacles with ease and can provide a bird's-eye view, which is ideal for mapping and monitoring tasks.

To reduce the number of required operators and speed up their mission, the robots must expose a good level of autonomy. Instead of sending low-level commands, they must be able to autonomously execute high-level tasks such as "grasp that object" or "fly to location X". This allows the operator to focus on the mission instead of low-level robot details. For reliable local navigation, robots must rely only on their onboard sensors, since the communication infrastructure is likely to be affected during disaster situations.

In this work, we demonstrate the benefits of a team of heterogeneous robots in a mock-up search-and-rescue scenario (see Figure 1a): a victim in an unknown environment must be found and provided with a first-aid kit. We first deploy an aerial robot that maps the environment and searches for the victim. Then, a mission is planned for a ground robot to reach the victim and provide it with a first-aid kit as fast as possible. Both the time required for driving and rubble removal are considered to plan the mission. Finally, the aerial robot guides the ground robot along the computed path to the victim.

The remainder of this paper is organized as follows. In Section II, we review related work both on collaboration of aerial and ground robots and on Navigation Among Movable Obstacles (NAMO). Our robots and the mock-up disaster scenario are introduced in Section III. In Sections IV, V, and VI we detail the mapping, planning, and execution of the

The authors are with the Robotics and Perception Group, University of Zurich, Switzerland—`http://rpg.ifi.uzh.ch`.

missions, respectively. Finally, we evaluate the performance of our mission planning algorithm and the entire system in Section VII.

II. Related Work

We first review related work on collaboration of ground and aerial robots in the search-and-rescue context. Then we provide a literature overview on Navigation Among Movable Obstacles (NAMO).

A. *Collaboration of Aerial and Ground Robots*

After the 2012 Mirandola earthquake in Italy, aerial and ground robots were deployed to build 3D maps of the interior of damaged buildings [3]. The robots were remote controlled and high stress and cognitive overload of both the aerial and ground robot operators were reported.

In [4], collaborative mapping of a damaged building after the 2011 Tokuhu earthquake with a ground and an aerial robot was presented. First, the ground robot was manually controlled through the building creating a 3D voxel grid. Second, locations inaccessible for the ground robot were mapped by the aerial robot.

Teams of aerial and ground robots for monitoring and tracking tasks were presented in [5]. Aerial robots search for targets which are then verified and tracked by ground robots. Only limited, high-level user interaction was required. However, the main focus was on an ad-hoc wireless network that is maintained during the mission.

In [6], an aerial robot is described to support ground personnel in searching for victims. Several search strategies are discussed. The system was tested outdoors with rescue professionals.

In [7], an aerial robot is described to assist a ground robot to reach a goal location. However, only local planning was performed and the aerial robot was piloted manually.

Since virtually all robots in today's missions are remote controlled, human-robot interaction for these scenarios was investigated by several authors, e.g. [8], [9]. Also, the recently started European project SHERPA [10] focuses on the interaction and collaboration of humans with both aerial and ground robots in alpine rescue missions.

Our work differs from those mentioned above in that we do not only map the environment, but also use these maps for mission planning and environment interaction. In particular, we make use of the ground robot's capability to *interact* with the environment using its robotic manipulator. In addition, our system is fully autonomous: no user interaction is required at any point after launching the aerial robot.

B. *Navigation Among Movable Obstacles (NAMO)*

Planning in modifiable environments was introduced in [11]. It was shown that, even for the simplest cases, the problem is NP-hard. A heuristic search algorithm was presented in [12]. More recently, Navigation Among Movable Obstacles (NAMO) [13], [14] became an active research topic in the field of humanoid robots. While very similar to our case, there is a difference in how we can manipulate obstacles. In all above-mentioned papers, the obstacles could be *pushed* by the robot. Here, contrarily, we can *lift* the obstacles and place them again at an *arbitrary* location.

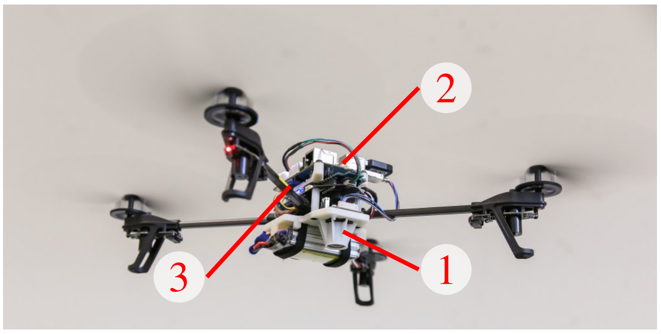


Fig. 2: A closeup of our quadrotor: down-looking camera (1), Odroid U3 quad-core computer (2), and PIXHAWK autopilot (3).

III. System Overview

We propose a system consisting of a quadrotor equipped with a monocular camera (see Figure 2) and a ground robot consisting of an omni-directional base and a 5-DOF manipulator (see Figure 3). When operating together, the two robots have all the capabilities we require for the considered search-and-rescue missions. Combining all the required capabilities in one single robot would render the system impractical and less flexible.

We use a laptop as ground station to visualize the progress of the mission. On the ground station, we also run our mission-planning algorithm and send high-level commands to both robots.[1] As soon as the robots have received their high-level commands, they both navigate fully autonomously while running all the required algorithms onboard. In the following, we describe the quadrotor platform, the ground robot, and the mock-up disaster scenario in more detail.

A. *Aerial Robot*

We built our quadrotor from selected off-the-shelf components and custom 3D-printed parts (see Figure 2). The components were chosen according to their performance and their ability to be easily customized.

Our quadrotor relies on the frame of the Parrot AR.Drone 2.0[2] including their motors, motor controllers, gears, and propellers. The platform is powered by one 1350 mA h LiPo battery, which allows a flight time of 10 min.

We completely replaced the electronic parts of the AR.Drone by a PX4FMU autopilot and a PX4IOAR adapter board developed in the PIXHAWK Project [15]. The PX4FMU consists, among other parts, of an IMU and a micro controller to read the sensors, run some low-level control to track desired body rates, and command the motors. Additionally to the PX4 autopilot, our quadrotors are

[1]Note that this could also run onboard one of the robots, rendering the laptop unnecessary.

[2]http://ardrone2.parrot.com/

equipped with an Odroid-U3 single-board computer.[3] It contains a 1.7 GHz quad-core processor running XUbuntu 13.10[4] and ROS.[5] The PX4 micro controller communicates with the Odroid board over UART, whereas the Odroid board communicates with the ground station over 5 GHz WiFi.

Our platform is easily reparable due to off-the-shelf components, inexpensive (1000 USD), lightweight (below 450 g), and therefore safe to use.

To stabilize the quadrotor, we make use of the gyros and accelerometers of the IMU on the PX4FMU as well as a downward-looking MatrixVision mvBlueFOX-MLC200w 752×480-pixel monochrome camera.[6]

The images from the downward-looking camera are processed on the Odroid by means of our Semi-Direct Visual Odometry (SVO[7]) pipeline [16]. The visual-odometry pipeline outputs an unscaled pose which is then fused with the IMU readings in an Extended Kalman Filter framework (Multi Sensor Fusion (MSF) [17]) to compute a metric state estimate. From this state estimate and a desired trajectory, we compute the desired body rates and collective thrust, which are then sent to the low-level controller on the PX4FMU.

B. Ground Robot

We use a KUKA youBot [18] as ground robot (see Figure 3). It consist of a mobile base and a 5-DOF manipulator with a two-finger gripper. The mobile base includes four omni-directional wheels, which allow the robot to also move sideways and rotate on the spot. This is a great advantage over standard wheels when navigating in confined spaces. Furthermore, the base is designed robust enough to carry a payload of 20 kg. The manipulator is able to lift up to 0.5 kg with its gripper. For controlling the arm and the base, the youBot comprises a mini ITX PC board with embedded Intel®Atom Dual-Core CPU running an Ubuntu operating system and ROS. To grasp objects fast and reliably, we developed a torque controller for the youBot arm [19], which allows to precisely track trajectories with its gripper.

To measure the relative position of obstacles in front of the youBot, we mounted a Hokuyo URG-04LX-UG01[8] laser scanner in front of its base. Finally, for this rescue mission, the youBot carries a first-aid kit on the side of its base as shown in Figure 3.

C. Mock-up Disaster Site

In this work, we consider a search-and-rescue mission after a disaster where robots are sent into areas that are too dangerous to be entered by human rescuers.

As a mock-up disaster site for our experiments (see Figure 4), we consider an area of known size (in our case

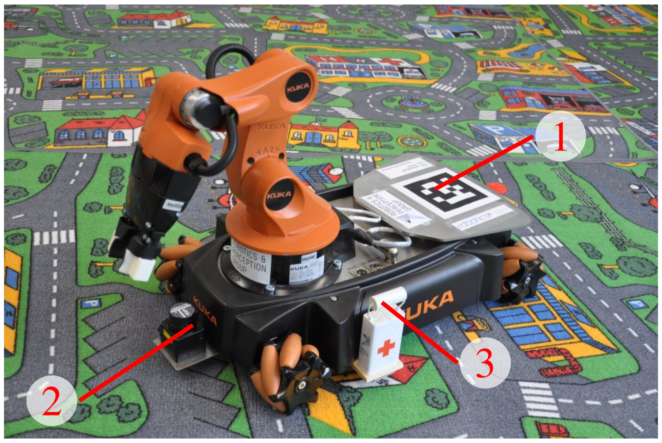


Fig. 3: KUKA youBot equipped with an AprilTag (1), a laser scanner (2), and a first-aid kit (3).

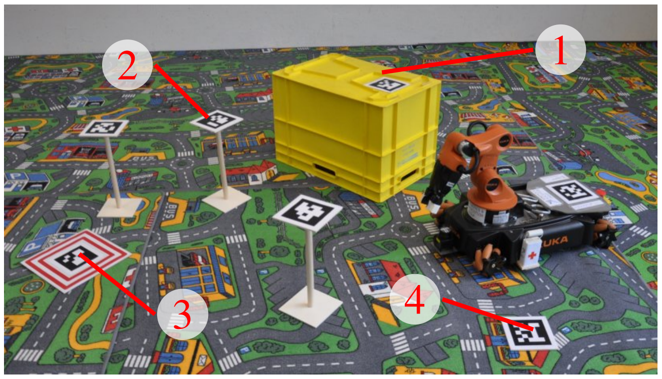


Fig. 4: Mock-up disaster site elements: fixed obstacle (1), movable obstacle (2), goal (3), and origin tag (4).

$4\,\text{m} \times 6\,\text{m}$) with different types of obstacles: some of them can be removed by the ground robot ("movable obstacles") and some can only be avoided ("fixed obstacles"). Furthermore, there is a goal location that represents a victim, which has to be provided with a first-aid kit by the ground robot as fast as possible. The locations of the obstacles and the goal are not known a priori.

All obstacles, the goal, and the ground robot are marked with an AprilTag [20] such that they can easily be detected in the camera image of the flying robot (see Figure 2). We make use of an additional AprilTag as origin tag, which provides a common reference frame for both robots. The origin tag is also used as reference point to define the search area that the aerial robot has to cover.

IV. Mapping

To map the defined area, we compute a *lawn-mower pattern* covering the entire area for the flying robot to take images. Images are only taken at specified locations on the lawn-mower pattern instead of being streamed continuously in order to minimize the required communication band with. These images, together with the onboard pose estimate of the quadrotor, are sent to the ground station while the flying robot is mapping. The obstacles are then detected in the

[3]http://www.hardkernel.com/main/products/prdt_info.php?g_code=G138745696275
[4]http://www.xubuntu.org/
[5]http://www.ros.org/
[6]http://www.matrix-vision.com/USB2.0-single-board-camera-mvbluefox-mlc.html
[7]http://github.com/uzh-rpg/rpg_svo
[8]http://www.hokuyo-aut.jp/02sensor/07scanner/download/products/urg-04lx-ug01/

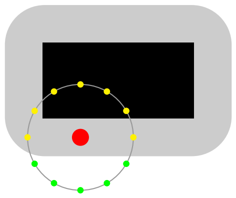


Fig. 5: The movable obstacle (red) can be grasped by the ground robot from positions on a circle around it. To avoid a collision with the fixed obstacle (black), the robot center cannot move inside the inflated area (gray). Thus, only the grasp locations marked in green are feasible.

image by their attached tag. To build a metrically consistent map, we run a pose-graph optimization over all detected tags of all images. We initialize the tag poses with the estimate of the quadrotor pose and run the optimization using iSAM [21].

The detected fixed obstacles are then inserted into a grid map together with the boundaries of the defined area as illustrated in Figure 8 in black. In addition to this grid map, we also provide the planner with the position of the movable obstacles, the pose of the ground robot, and the goal location.

V. Mission Planning

When planning a mission for the ground robot, we consider its position to be the center point of its base and we represent the map as a grid where we inflate all fixed obstacles by the radius of the robot base. To grasp an obstacle, the robot's position must be on a circle around the obstacle with a radius corresponding to the distance from the gripper to the base center (see Figure 5). We denote the points on this circle as possible grasp locations of that obstacle.

Our planning is based on the A* algorithm with the mission-execution time as cost. First, for each movable obstacle, we calculate the minimum cost to the goal location when no other movable obstacle would be present. This gives us a lower bound on the cost to go from each movable obstacle to the target location and is therefore an admissible heuristic. Then, we search for every feasible shortest path from the start location to any of the grasp locations of every movable obstacle or the goal location directly. For every path to a movable obstacle that we find, we add the cost to remove it and compute all feasible paths to all other removable obstacles or the goal location. All the possible missions that are created this way are stored in a priority queue according to their estimated cost. This process is continued until we find a feasible mission to the goal location.

Until this point, we do not consider the cost of driving the obstacle to a feasible place location, which can be necessary as described further in Section V-A. Therefore, all the computed costs are lower bounds and not necessarily the actual costs of the corresponding mission. When a feasible mission to the goal is found, we compute the precise cost including possible driving backwards to place obstacles. Note that this can only increase the cost of the mission. If the actual cost of the considered mission is now higher than the estimated cost of other missions in the priority queue, we have to continue the planning steps for these missions as described above. The time-optimal mission is found if its actual cost is smaller than the estimated cost of every other mission in the priority queue. This procedure is summarized in Algorithm 1.

Algorithm 1 Mission Planning

```
calculate lower bound on cost to go from each obstacle
$min\_cost \leftarrow \infty$
add empty mission with start pose to Priority Queue (PQ)
while $cost(PQ.top()) < min\_cost$ do
    $current\_mission \leftarrow PQ.top()$
    add remaining obstacles to map
    for all paths to these obstacles do
        if lower bound of new mission $< min\_cost$ then
            $new\_mission \leftarrow current\_mission + path$
            add $new\_mission$ to PQ
        end if
    end for
    compute path to goal
    if path is feasible then
        $current\_mission \leftarrow current\_mission + path$
        refine mission
        if $cost(current\_mission) < min\_cost$ then
            $min\_cost \leftarrow mission\_cost$
            $best\_mission \leftarrow current\_mission$
        end if
    end if
end while
```

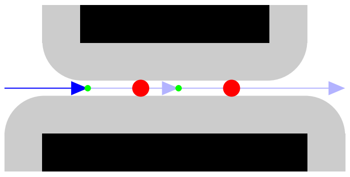
A. *Place Positions of Removed Obstacles*

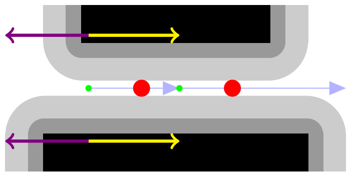
When removing an obstacle, we ideally try to place it on the side of the ground robot while the robot stands still in order to save time. Nonetheless, in narrow passages (e.g., in a passage as in Figure 6), the robot cannot just place the obstacle to the left or right, but needs to carry it to a feasible place location. To do so, we search for unoccupied place locations along two lines parallel to the driving path. First, we search in forward direction up to the next obstacle grasp location. If we cannot find a feasible place position in that direction, we then also search in backward direction. This happens for example if two obstacles are in a narrow passage (cf. Figure 6): the first obstacle must be carried out of the passage by driving backwards. The second obstacle can be placed when exiting the passage.

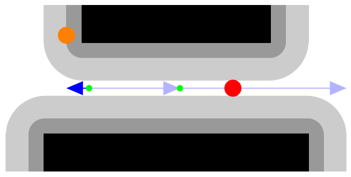
B. *Path Refinement*

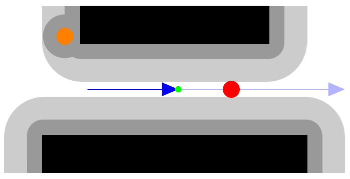
We search paths on the grid map using single-source Dijkstra's algorithm. However, these paths should be smoothed for a ground robot to be executed (see Figure 7). We therefore refine these paths using a simple algorithm (see Algorithm 2).

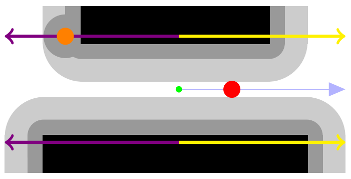
Although there exist more sophisticated planning algorithms on grid maps (such as Theta* [22]), this is of minor


(a) Drive to first obstacle and grasp it


(b) Search for possible place location


(c) Drive backwards to place first obstacle


(d) Drive to second obstacle and grasp it


(e) Search for possible place location

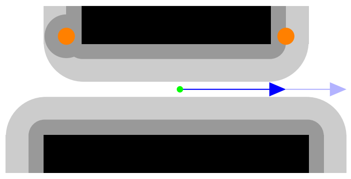

(f) Drive and place second obstacle

Fig. 6: To illustrate our obstacle-placing algorithm, we consider a narrow corridor with two movable obstacles inside. Fixed obstacles (black) are inflated to mark the area where the robot cannot drive (light gray) or cannot place obstacles (dark gray). Green dots indicate the grasp locations of movable obstacles (red). Since we place obstacles to the side of the robot, we search along parallel lines of the path (blue) for place locations. First, we search in forward direction (yellow) up to the next grasp location. If we cannot find a place location there, we start searching in backward direction (violet). The chosen obstacle place position is indicated in orange. The time for driving backwards is taken into account by our mission-planning algorithm which possibly affects the optimal mission (cf. Figures 9a and 9b). An example of a corridor with more movable obstacles is shown in Figure 10.

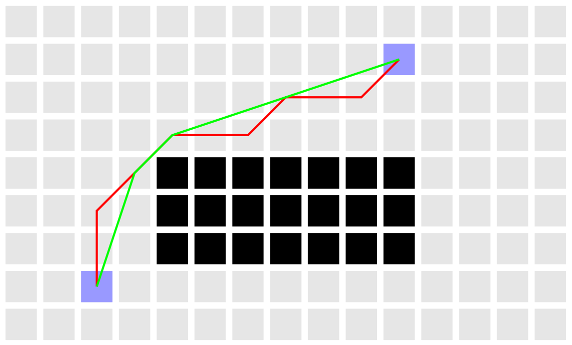


Fig. 7: The output of Dijkstra's algorithm (red) is refined (green) to find a smoother trajectory for the ground robot.

Algorithm 2 Optimizing Grid Map Paths

```
add first cell of old path to new path
for all cells in old path do
    if no line of sight to last cell in new path then
        add previous cell of old path to new path
    end if
end for
add last cell of old path to new path
for all cells in new path do
    update orientation such that it points to next cell
end for
```

concern in our situation, where the robot is much larger than the grid resolution.

VI. Mission Execution

A feasible mission plan for the ground robot consists of a series of actions such as driving straight line path segments, removing obstacles, and delivering the first-aid kit. Each of these actions for the ground robot are commanded by the aerial robot in an iterative fashion. Once an action is commanded, the ground robot executes it without any external feedback. The aerial robot is then following the ground robot to command the next action. Commanding the ground robot in this iterative fashion to perform open-loop maneuvers eliminates problems with communication delays. Furthermore, it keeps the required communication at a minimum.

When the ground robot has to drive a straight line path segment, it is first localized in the map by the aerial robot. It is then commanded to execute a motion relative to the current location using its wheel odometry without feedback from the aerial robot. When the ground robot has executed the open-loop motion, it is again localized in the map by the aerial robot to compensate for accumulated drift of the wheel odometry. To remove an obstacle, the ground robot is commanded to drive to the chosen grasp location such that it can reach it with its gripper. When in front of the obstacle, the ground robot makes use of its laser scanner to measure the relative position of the obstacle precisely. It then grasps the obstacle and places it to a location that is again commanded by the aerial robot. After removing all the obstacles in the way, the ground robot can approach the goal location. Since we want to deliver the first-aid kit directly onto the goal location, the ground robot stops in front of the goal location such that it is within the gripper's reach (cf. Figure 8). Once it is there, the ground robot is commanded to place the first-aid kit.

VII. Results

In this section, we evaluate both our mission planner and the overall system performance. We analyze the mission plans for special, engineered cases. Further, we evaluate the computation time depending on the number of movable obstacles in the scene. Finally, we present the overall system performance during many demonstrations at a trade fair.

As parameters for the mission planning, we used 0.2 m/s driving speed, 30 °/s rotational speed, and 15 s for grasping and placing an obstacle. We set the driving speeds of the youBot accordingly and measured the required grasp-and-place. For creating the grid map, we used a cell size of 5 cm.

A. Mission Planning

We evaluate our mission-planning algorithm using special and random cases. The visualization of the output is explained in Figure 8.

We demonstrate the influence on the optimal mission when driving backwards to place an obstacle in Figure 9. Even if the south path in Figure 9a is shorter, the mission time would be longer due to the time required for additional driving backwards to place the first obstacle. However, if the north path is blocked, the mission planner will choose the other path (see Figure 9b). If many obstacles are in a small corridor as in Figure 10, the ground robot must carry all but one out of it backwards to be able to traverse the corridor.

In Figure 11, we demonstrate the influence of the time required to remove an obstacle on the optimal mission.

Randomly generated scenes and the respective missions are shown in Figure 12.

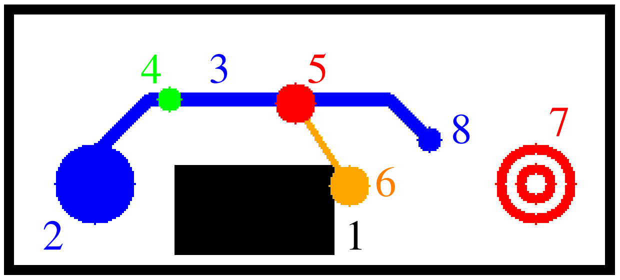
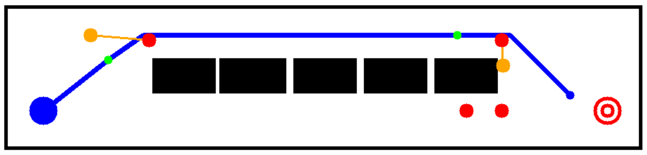
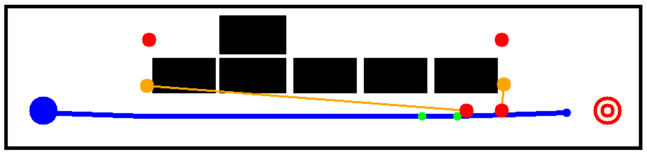


Fig. 8: Explanation of mission planner output: fixed obstacle (1), start location of the ground robot (2), path to drive (3), grasp location (4) of movable obstacle (5), place location (6), and goal (7). From the end of the path (8), the first-aid kit is delivered to the goal.

To evaluate the computational performance of our mission planner, we fixed a scene (cf. Figure 12a) and varied the number of movable obstacles. For every chosen number of movable obstacles we run 50 trials with randomly placed movable obstacles to measure the computation time. The results are shown in Figure 13. In two out of 250 cases, the planner could not find a feasible mission.



(a) Our mission-planning algorithm computes the fastest mission and not the shortest path. Therefore, it chooses the north path since it requires no driving backwards to place obstacles.



(b) If the path of Figure 9a is blocked by an additional fixed obstacle, the south path is chosen. Note that this mission takes longer than the one above, since the robot has to drive backwards to place the obstacle.

Fig. 9: Placing movable obstacle might require additional backwards driving and, thus, increase the mission time. Therefore, the output of our mission-planning algorithm can be different from the shortest path. Color-coding is explained in Figure 8.

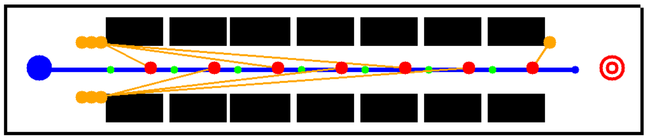


Fig. 10: All but one obstacle in a narrow corridor must be carried back to where the robot entered the corridor. Only the last obstacle is placed at the end of the corridor. Color-coding is explained in Figure 8.

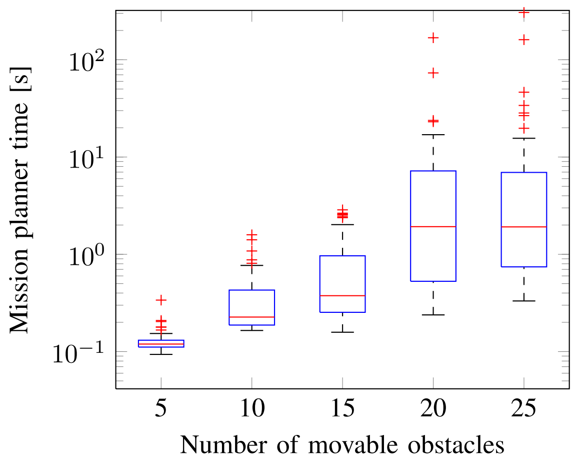
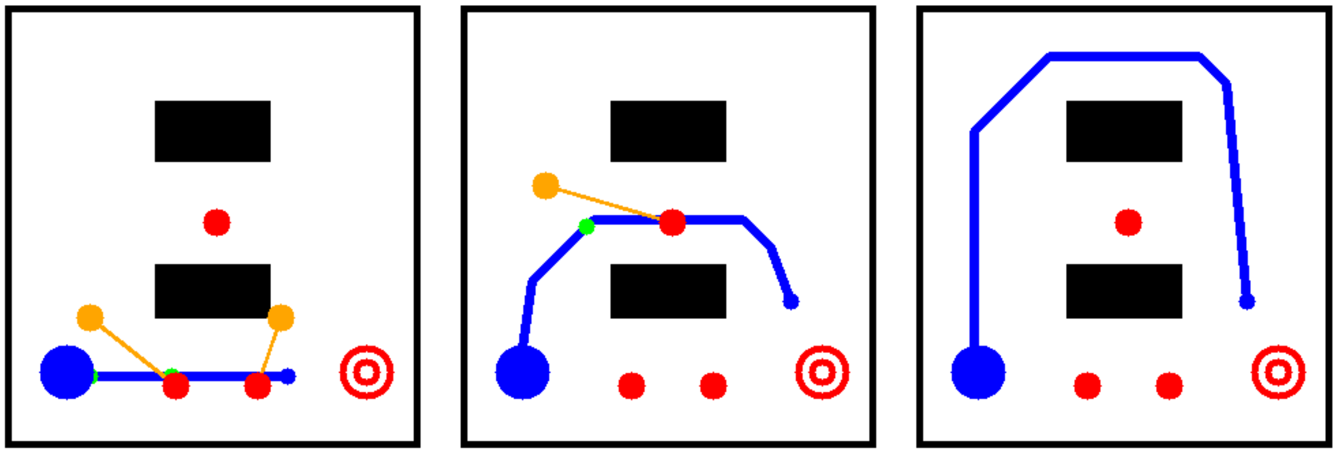
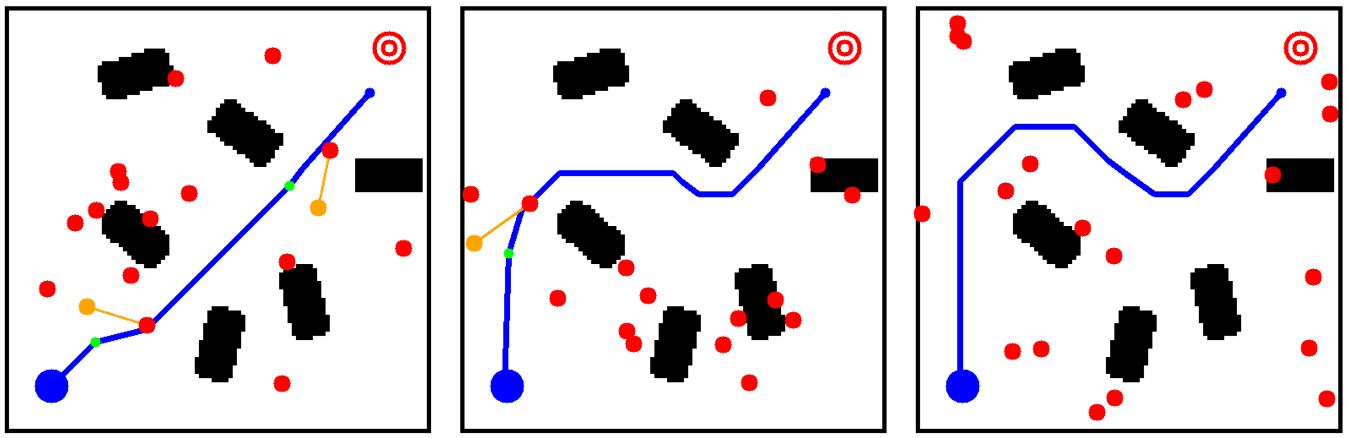


Fig. 13: Computation time of our mission planning algorithm for different numbers of movable obstacles. For every chosen number of movable obstacles we run 50 trials while placing the obstacles randomly. Example scenes are shown in Figure 12.



(a) Removal time 5 s (b) Removal time 11 s (c) Removal time 15 s

Fig. 11: The time-optimal mission depends on the time required to remove an obstacle. Color-coding is explained in Figure 8.



(a) 15 movable obstacles (b) 15 movable obstacles (c) 20 movable obstacles

Fig. 12: Movable obstacles are placed randomly in a scene to evaluate the computation time of our mission-planning algorithm. Results are shown in Figure 13 and color-coding is explained in Figure 8.

B. Overall System Performance

Our system was demonstrated at the AUTOMATICA'14 trade fair in Munich on four subsequent days, once every hour (see Figure 14). The entire setup was dismounted after every demonstration. Both the movable and fixed obstacles were placed randomly before each run. The mission was accomplished successfully 23 out of 27 times (81%). Reasons for failures were mapping errors and wireless communication issues. In all these failure cases, our system reacted in a fail-safe way: the robots stopped to move and the error was reported to the operator.

VIII. Conclusion

In this paper, we demonstrated the autonomous collaboration of an aerial and a ground robot in a mock-up disaster scenario. We detailed the algorithm for mission planning for the ground robot and evaluated it for both special and random scenarios. The high success rate during a trade fair showed the robustness of our system.

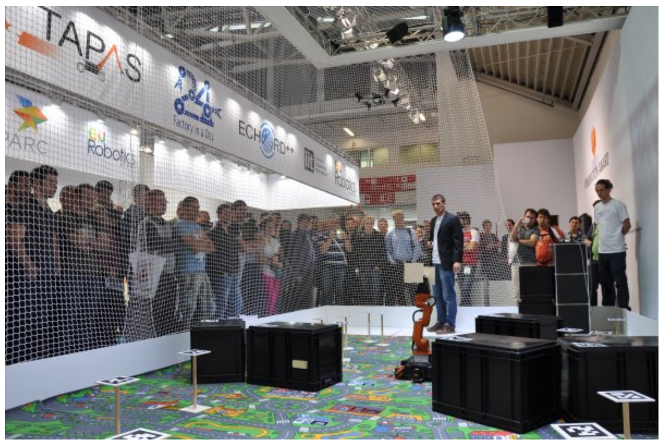


Fig. 14: Demonstration of our system at the AUTOMATICA trade fair in Munich, Germany in June 2014.

Wireless communication is a major concern when dealing with multiple robots. We tackled this by performing all crucial computations onboard the robots. Thus, we only need to communicate high-level commands and sparse information, which do not require low-latency or high-bandwidth communication links.

A. Future Work

In real-world scenarios, two main assumptions of this paper are not valid: first, in our case, obstacles are marked with tags. Second, we assume the world to be flat and, therefore, plan missions only in 2D. We plan to overcome these limitations by building a 3D map using the images from the aerial robot, e.g., using our real-time dense reconstruction algorithm (REMODE) [23]. The maps from the aerial robot could also be combined with the ones from the ground robot [24]. Since wheeled robots, such as the KUKA youBot used here, are limited to flat surfaces, we aim at deploying legged robots for such missions.

Other directions of research include covering larger areas using fixed-wing aerial robots, extending the setup to multiple aerial and ground robots, computing and executing mission plans already while the flying robot is mapping, and adapting the mission plan when the environment changes during execution.

References

[1] R. Murphy, *Disaster Robotics*. The MIT Press, Cambridge, MA, 2014.

[2] K. Nagatani, S. Kiribayashi, Y. Okada, K. Otake, K. Yoshida, S. Tadokoro, T. Nishimura, T. Yoshida, E. Koyanagi, M. Fukushima, and S. Kawatsuma, "Emergency response to the nuclear accident at the Fukushima Daiichi nuclear power plants using mobile rescue robots," *J. of Field Robotics*, vol. 30, no. 1, pp. 44–63, 2013.

[3] G.-J. Kruijff, V. Tretyakov, T. Linder, F. Pirri, M. Gianni, P. Papadakis, M. Pizzoli, A. Sinha, E. Pianese, S. Corrao, F. Priori, S. Febrini, and S. Angeletti, "Rescue robots at earthquake-hit Mirandola, Italy: A field report," in *IEEE Intl. Symp. on Safety, Security, and Rescue Robotics (SSRR)*, 2012.

[4] N. Michael, S. Shen, K. Mohta, Y. Mulgaonkar, V. Kumar, K. Nagatani, Y. Okada, S. Kiribayashi, K. Otake, K. Yoshida, K. Ohno, E. Takeuchi, and S. Tadokoro, "Collaborative mapping of an earthquake-damaged building via ground and aerial robots," *J. of Field Robotics*, vol. 29, no. 5, pp. 832–841, Sep. 2012.

[5] M. A. Hsieh, A. Cowley, J. F. Keller, L. Chaimowicz, B. Grocholsky, V. Kumar, C. J. Taylor, Y. Endo, R. C. Arkin, B. Jung, D. F. Wolf, G. S. Sukhatme, and D. C. MacKenzie, "Adaptive teams of autonomous aerial and ground robots for situational awareness," *J. of Field Robotics*, vol. 24, no. 11–12, pp. 991–1014, 2007.

[6] M. A. Goodrich, B. S. Morse, D. Gerhardt, J. L. Cooper, M. Quigley, J. A. Adams, and C. Humphrey, "Supporting wilderness search and rescue using a camera-equipped mini UAV," *J. of Field Robotics*, vol. 25, no. 1–2, pp. 89–110, 2008.

[7] M. Garzon, J. Valente, D. Zapata, and A. Barrientos, "An aerial-ground robotic system for navigation and obstacle mapping in large outdoor areas," *Sensors*, vol. 13, no. 1, pp. 1247–1267, 2013.

[8] T. Perkins and R. R. Murphy, "Active and mediated opportunistic cooperation between an unmanned aerial vehicle and an unmanned ground vehicle," in *IEEE Intl. Symp. on Safety, Security, and Rescue Robotics (SSRR)*, 2013.

[9] M. Langerwisch, T. Wittmann, S. Thamke, T. Remmersmann, A. Tiderko, and B. Wagner, "Heterogeneous teams of unmanned ground and aerial robots for reconnaissance and surveillance - a field experiment," in *IEEE Intl. Symp. on Safety, Security, and Rescue Robotics (SSRR)*, 2013.

[10] L. Marconi, C. Melchiorri, M. Beetz, D. Pangercic, R. Siegwart, S. Leutenegger, R. Carloni, S. Stramigioli, H. Bruyninckx, P. Doherty, A. Kleiner, V. Lippiello, A. Finzi, B. Siciliano, A. Sala, and N. Tomatis, "The SHERPA project: Smart collaboration between humans and ground-aerial robots for improving rescuing activities in alpine environments," in *IEEE Intl. Symp. on Safety, Security, and Rescue Robotics (SSRR)*, 2012.

[11] G. Wilfong, "Motion panning in the presence of movable obstacles," in *Proc. ACM Symp. Computat. Geometry*, 1988.

[12] P. Chen and Y. Hwang, "Practical path planning among movable obstacles," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 1991.

[13] M. Stilman and J. J. Kuffner, "Navigation among movable obstacles: Real-time reasoning in complex environments," *Intl. J. of Humanoid Robotics*, vol. 2, no. 4, pp. 479–504, 2005.

[14] M. Stilman, K. Nishiwaki, S. Kagami, and J. J. Kuffner, "Planning and executing navigation among movable obstacles," *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2006.

[15] L. Meier, P. Tanskanen, L. Heng, G. H. Lee, F. Fraundorfer, and M. Pollefeys, "PIXHAWK: A micro aerial vehicle design for autonomous flight using onboard computer vision," *Autonomous Robots*, vol. 33, no. 1–2, pp. 21–39, 2012.

[16] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2014.

[17] S. Lynen, M. Achtelik, S. Weiss, M. Chli, and R. Siegwart, "A robust and modular multi-sensor fusion approach applied to MAV navigation," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2013.

[18] R. Bischoff, U. Huggenberger, and E. Prassler, "KUKA youBot - a mobile manipulator for research and education," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2011.

[19] B. Keiser, "Torque control of a KUKA youBot arm," Master's thesis, University of Zurich, 2013.

[20] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2011.

[21] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental smoothing and mapping," *IEEE Trans. Robotics*, vol. 24, no. 6, pp. 1365–1378, Dec. 2008.

[22] A. Nash, K. Daniel, S. Koenig, and A. Felner, "Theta*: Any-angle path planning on grids," in *AAAI Conf. on Artificial Intelligence*, 2007.

[23] M. Pizzoli, C. Forster, and D. Scaramuzza, "REMODE: Probabilistic, monocular dense reconstruction in real time," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2014.

[24] C. Forster, M. Pizzoli, and D. Scaramuzza, "Air-ground localization and map augmentation using monocular dense reconstruction," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2013.