Cooperative Autonomous Search, Grasping and Delivering in a Treasure Hunt Scenario by a Team of UAVs

Cooperative Autonomous Search, Grasping and Delivering in a Treasure Hunt Scenario by a Team of UAVs

Vojtěch Spurný[*]
Department of Cybernetics
Faculty of Electrical Engineering
Czech Technical University in Prague
vojtech.spurny@fel.cvut.cz

Tomáš Báča[*]
Department of Cybernetics
Faculty of Electrical Engineering
Czech Technical University in Prague
tomas.baca@fel.cvut.cz

Martin Saska[*]
Department of Cybernetics
Faculty of Electrical Engineering
Czech Technical University in Prague
martin.saska@fel.cvut.cz

Robert Pěnička[*]
Department of Cybernetics
Faculty of Electrical Engineering
Czech Technical University in Prague
robert.penicka@fel.cvut.cz

Tomáš Krajník[†]
Department of Computer Science
Faculty of Electrical Engineering
Czech Technical University in Prague
tomas.krajnik@fel.cvut.cz

Justin Thomas[‡]
GRASP Laboratory
University of Pennsylvania
Philadelphia, US
jut@seas.upenn.edu

Dinesh Thakur[‡]
GRASP Laboratory
University of Pennsylvania
Philadelphia, US
tdinesh@seas.upenn.edu

Giuseppe Loianno[§]
Tandon School of Engineering
New York University
New York, US
loiannog@nyu.edu

Vijay Kumar[‡]
GRASP Laboratory
University of Pennsylvania
Philadelphia, US
kumar@seas.upenn.edu

Abstract

This paper addresses the problem of autonomous cooperative localization, grasping and delivering of colored ferrous objects by a team of UAVs. In the proposed scenario, a team of UAVs is required to maximize the reward by collecting colored objects and delivering them to a predefined location. This task consists of several sub-tasks such as cooperative coverage path planning, object detection and state estimation, UAV self-localization, precise motion control, trajectory tracking, aerial grasping and dropping, and decentralized team coordination. The failure recovery and synchronization jobs manager is used to integrate all the presented sub-tasks together and also to decrease the vulnerability to individual sub-task failures in real-world conditions. The whole system was developed for the Mohamed Bin Zayed International Robotics Challenge (MBZIRC) 2017, where it achieved the highest score and won Challenge No.3 - Treasure Hunt. This paper does not only contain results from the

[*]http://mrs.felk.cvut.cz
[†]http://cs.felk.cvut.cz
[‡]http://www.grasp.upenn.edu
[§]http://www.nyu.edu

MBZIRC 2017 competition but it also evaluates the system performance in simulations and field tests, that were conducted throughout the year-long development and preparations for the competition.

1 Introduction

Small autonomous Unmanned Aerial Vehicles (UAVs) are widely used in numerous applications of data collection due to their potential for rapid deployment and their ability to reach locations inaccessible by ground robots. While fixed wing UAVs have the advantage of stable flight at high speeds, long range, and long flight time, rotary wing UAVs (such as the popular multi-rotor helicopters) benefit from their capacity for high manoeuvrability, vertical take off and landing, flight in cluttered environments in close proximity to obstacles, and hovering in a desired position in a 3D environment. The ability to precisely reach a desired 3D position and hover in place is crucial for long-term information gathering, and especially for physical interaction with objects in the workspace. Delivery applications composed of acquisition, transport and drop-off provide an example requiring interaction with the environment during autonomous flight. This is the topic discussed in our paper.

A multiple cooperative delivery mission (called Treasure Hunt) was the most complex task in the 2017 Mohamed Bin Zayed International Robotics Challenge (MBZIRC[1]). In the competition, the delivery task was solved in its full complexity, including searching for objects with unknown positions, grasping moving objects, and cooperation among multiple UAVs working in concert. The deployment of a team of UAVs was motivated by the limited total mission time, and by including large objects with weights exceeding the maximum payload of the individual robots. In the mission, 23 objects (10 static, 10 dynamic and 3 large) had to be localized in an outdoor arena and collected by three UAVs of limited size. While the small objects (static and dynamic) could be lifted by a single UAV, the large objects required two UAVs to transport them.

The system that exhibited the best performance among all participants in the MBZIRC competition in the Treasure Hunt challenge is presented in this paper[2]. The system design is driven by the specific task proposed and precisely specified by the organizers. The approach is tailored to provide high robustness and performance to solve the challenging task by modification of available robotic methods and designing new algorithms where necessary. Nevertheless, the proposed system is easily re-usable in a large set of multi-UAV scenarios as shown in section 1.2. The core of the system is the Failure recovery and Synchronization jobs Manager (FSM), which is crucial for managing all subsystems and for coordinating all UAVs sharing the same workspace. The FSM is also needed in order to achieve the reliability required for the deployment of UAVs in real-world conditions, which requires the ability to recover from UAV failures and also from a malfunction of the localization and communication infrastructure. For example, the robots can easily collide with the objects being grasped due to a wind gust which, in combination with the ground effect, can create a hardly predictable external force on the UAV in the final phase of the approach to an object. Such a collision could result in a UAV crash, deadlock, or an overturned object. Moreover, malfunctions of UAV subsystems such as camera drop outs, incorrect rangefinder measurements, gripper failure or gripper feedback failure, and imprecise object gripping, can be expected in demanding outdoor conditions. All these eventualities need to be considered by the system to enable undisturbed operation of the remaining robots in the event of a UAV failure, limited operation of a UAV with a faulty subsystem, or an unsuccessful or interrupted grasping task. From this point of view, the proposed FSM concept can be considered as a hierarchical state machine with included synchronization and failure recovery abilities, which may be effectively re-used in any complex multi-UAV task involving environment interaction.

Although the rules of the MBZIRC competition allowed the use of GNSS (Global Navigation Satellite System) and the even more precise DGPS (Differential Global Positioning System) for UAV localization,

[1] `http://www.mbzirc.com/` - Accessed: 2018-07-17

[2] `http://mrs.felk.cvut.cz/projects/mbzirc` - Accessed: 2018-07-17

the availability of these systems was not guaranteed. For example, GNSS information was available only intermittently, due to interference with other transmitters located at the competition site and occlusion of satellites by the surrounding buildings and infrastructure. The provided Wi-Fi infrastructure was even less reliable and therefore the proposed FSM approach leverages the combination of different modes of the system based on the availability of Wi-Fi, GNSS, and DGPS. In addition to the FSM, a sensor fusion mechanism is presented for combining information from various onboard sensors (onboard IMU, GPS, DGPS, rangefinder, camera) which must be considered as potentially unreliable at any time. It is vital that the UAV may continue with the task despite lacking some sensor data (e.g. precise measured altitude above ground), because the competition rules did not allow any human intervention or debugging during the trials, and which is also the case in most of the real-world UAV applications.

Another important subsystem, which is crucial in tasks requiring interaction with the environment, is relative detection and estimation of the state of the objects requiring interaction. In the presented system, the relative localization technique relies on onboard vision, since the objects in the competition were designed to support such an approach. The shape and color of the objects were specified prior to the mission and a color-based key was used to identify the score for collecting the particular object and to distinguish the object type. Static, dynamic and long objects were labeled by different colors, all easily distinguishable from the background. Therefore, the vision approach is the simplest way to acquire all data required for the high-level planning (the score, type and position estimate), and also for the visual servoing in the grasping task (precise relative positions of objects). However, any alternative relative localization system can be easily integrated based on the application. State estimation of the object is necessary mainly for dynamic objects, where a velocity estimate of the object needs to be taken into account by the UAV control modules.

Two flight behaviours are required in the Treasure Hunt task: trajectory following and precise visual servoing. The trajectory tracking mode is used to search for the object in the environment, to approach the vicinity of the object, and to transport the object to the required location. The most important property of this controller is rapid and smooth movement along the trajectory provided by the high-level planning. The visual servoing applied in the final phase of grasping can be realized more slowly, but the requirements on precision are much higher. In the paper, we will present a novel Model Predictive Control (MPC)-based approach that allows integration of the UAV state estimation (including external forces produced by the wind and ground effect) and target state estimation (a position and velocity estimate of the currently observed object), enabling our robots to reach the target with a maximum position error of 8 cm, which is determined by the diameter of the object and the size of the gripper.

1.1 State-of-the-art

Rotorcraft or rotor-wing UAVs are suitable for tasks with object manipulation, due to their ability to hover on the spot. Their usage in this field has already been investigated in several publications, mainly for a single UAV, in particular sub-parts such as gripper or manipulator design, control techniques, and object detection.

The design of a manipulator for use in industrial applications, for aerial inspection by contact and also for aerial manipulation is described in (Fumagalli et al., 2016). The design of a multi-degree arm manipulator placed on UAVs is presented in (Morton and Toro, 2016) and in (Korpela et al., 2012). The idea of using a suction-based gripper for versatile aerial grasping is presented and experimentally verified in (Kessens et al., 2016). Other gripper designs are presented in (Pounds et al., 2011b; Mellinger et al., 2011).

A study about determining stability bounds, in which the changing mass-inertia parameters of the system due to the grasped object will not destabilize a PID flight controller for helicopters, is presented in (Pounds et al., 2011a). The authors of (Thomas et al., 2014) introduce a controller and a planner for high-speed aerial grasping, using a quadrotor UAV with a claw-like gripper. Their approach is used for grasping a cylindrical object relying on feedback from a monocular camera and an inertial measurement unit onboard the aerial robot. Images from the camera are used for computing the desired pitch angle, and the remaining

axes (roll and yaw) are controlled using feedback from the vision motion capture system. In (Ghadiok et al., 2012), a system for autonomous grasping of objects using a monocular IR camera is introduced. Detection of the objects is based on finding an IR beacon, which has to be placed on the objects. The authors also rely only on onboard sensors, but the position and yaw estimation is computed offboard on the ground station. A methodology for controlling a multi-arm manipulating aerial vehicle is presented in (Orsag et al., 2013). The control of a system where the control input is generated for the UAV and the manipulator joints simultaneously is described in (Kamel et al., 2016; Heredia et al., 2014; Kannan et al., 2016). The papers (Santamaria-Navarro et al., 2017; Kim et al., 2016; Lippiello et al., 2016) present a vision guidance approach using an image-based visual servo for an aerial manipulator. A method for planning a time-optimal trajectory for a quadrotor with the goal of grasping a moving target is introduced in (Spica et al., 2012). However the solution is presented only by simulations.

Detecting and estimating the object is a challenging task that needs to be investigated for autonomous grasping. Online detection of the known object and estimation of its position using features from images are described in (Ramon Soria et al., 2017). Another method for onboard object extraction based on stereo vision for autonomous grasping of objects is presented in (Ramon Soria et al., 2016). However, the aforementioned methods rely on stereo or depth sensors, which are not used on our UAVs. To detect the colored objects, we modified a computationally efficient method (Krajník et al., 2014), which already proved its reliability and accuracy in real-world conditions.

Ways of transporting large objects by multiple UAVs have already been investigated in (Parra-Vega et al., 2013; Gioioso et al., 2014; Mellinger et al., 2013). A control scheme for cooperative simultaneous manipulation of an object by a team of UAVs is described in (Parra-Vega et al., 2013). The idea of grasping and manipulating objects by a swarm of UAVs has been also studied in (Gioioso et al., 2014), where the swarm is teleoperated using the free motion of a human hand. Both these works lack experimental verification, because the systems were tested only in simulations. Transport of large objects by multiple UAVs had been achieved in (Mellinger et al., 2013). However the experiments were done in an indoor environment under the Vicon[3] motion capture system.

Solutions for the Treasure Hunt scenario have already been presented by two teams participating in the MBZIRC competition which had worked on this scenario autonomously. The approach used by the team from ETH Zurich is described in (Bähnemann et al., 2017), and the approach used by the team from the University of Bonn is presented in (Nieuwenhuisen et al., 2017). Both teams relied on an electro-permanent magnetic gripper for grasping ferrous objects, which are recognized using a color blob detection algorithm. They also used a similar approach for locating the objects. Firstly, the arena is cooperatively searched by UAVs to create a map of the objects, and then an attempt is made to grasp and deliver each object in the map. However, the solution in (Bähnemann et al., 2017) relies on a Wi-fi communication infrastructure, and the authors do not propose any alternative in the event of communication black-out. They also do not explain how they solve the problem of multiple UAVs coordination over the drop-off zone. In (Nieuwenhuisen et al., 2017), the authors mention a conservative solution for a disturbed communication network. However, this solution is not explained in detail, and therefore their approach cannot be directly replicated and evaluated. Furthermore, their controller does not compensate for external factors such as wind, which is a common disturbance in an outdoor environment.

1.2 Contribution

The contribution of this paper correlates directly with the expected contribution of the MBZIRC challenge. A board of respected scientists[4] from leading robotic groups worldwide selected the Treasure Hunt scenario as the most challenging task in the MBZIRC event for numerous reasons. This scenario extends state-of-the-art systems in various ways: deployment of multiple UAVs in the same outdoor workspace, multi-robot scanning of the environment with no prior information on the position of objects, online distribution of tasks

[3] `http://www.vicon.com/` - Accessed: 2018-07-17
[4] `http://www.mbzirc.com/committee` - Accessed: 2018-07-17

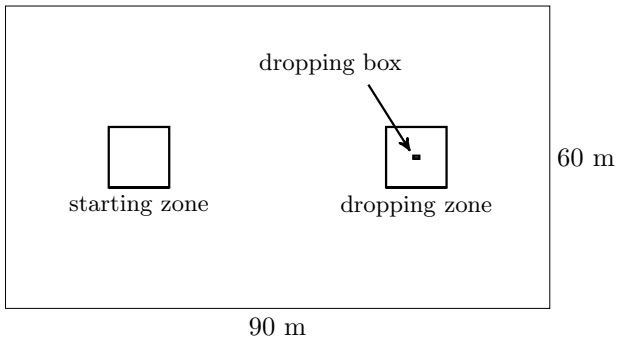
to UAVs based on the obtained information, and physical interaction with the environment. Indeed, physical interaction of UAVs with objects in an unknown outdoor environment, especially cooperatively (some objects require the cooperation of multiple UAVs), is a challenging and innovative task, mainly if it must be solved in demanding windy environments, such as the MBZIRC 2017 venue in Abu Dhabi. The strong wind gusts present in the location between the coast and the desert significantly influence the precision and the stability of the UAV controllers, particularly in the final phase of object grasping, where they are combined with the ground effect. Further, the light conditions (*e.g.* the strong and variable sunshine) make the vision task more complicated than in a laboratory environment. The multi-robot aspect requires rapid communication and coordination of UAVs, which seemed to be a bottleneck for the approaches presented by most of the other teams. Our solution to the challenges caused by unreliable communication is also a contribution to robotic research.

The overall contribution of our paper goes beyond the MBZIRC challenge, as it contains a comprehensive description of all components of the system that can be employed in various collaborative multi-UAV missions, including physical interaction of robots and the environment. Although the system is primarily designed for outdoor deployment with a GNSS signal available, it can be employed in GNSS-denied conditions with only a slight modification, since object grasping is realised by visual servoing, which relies on relative localisation only. Besides object grasping and delivery tasks, the system has been successfully deployed in numerous multi-UAV applications, including detection of sources of radiation and EMF fields (Saska, 2017), inspection and documentation of historical sites (Saska et al., 2017), reconnaissance and surveillance missions (Pěnička et al., 2017a; Pěnička et al., 2017b), etc.

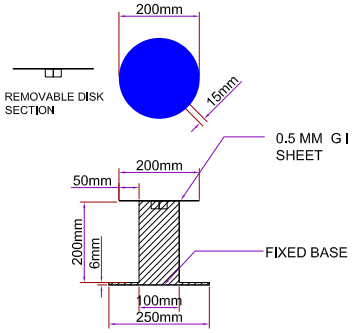
Another contribution of this paper for the robotic community is based on the fact that the next MBZIRC event intends to build on the achievements of MBZIRC 2017, and to propose even more challenging tasks that are beyond the current state-of-the-art in robotics. Although 143 teams applied to participate in the 2017 contest, including the best robotic labs worldwide, only 4 groups were able to grasp at least one object autonomously during the competition. To maximize the impact of future MBZIRC events and to encourage more competition, which will again push the limits of robotic systems, it is necessary for more teams to succeed in solving the challenging scenarios. A logical starting point is to use, or at least be inspired by, the approach that demonstrated the best performance in the 2017 MBZIRC, which is presented in this paper. Moreover, we would like to share and highlight the parts of the system and the phases in its development that brought added value in comparison with the systems of our competitors. Our experience and our solutions to the proposed challenges should be beneficial in further MBZIRC contests, in other robotic competitions, and also for the design of autonomous UAV systems for deployment in emergency applications. The rules of the competition forced teams to design a system for immediate deployment (the preparation time was only 20 minutes for the multi-UAV challenge) and for operation within a given time, without the option of postponing the start of the mission. This contrasts with most robotic experiments presented in the literature, where only successful trials and demos are presented. Short preparation time and a successful start on demand, without the possibility of repeated trials, are required by industry and in emergency applications, and the MBZIRC competition was designed to force teams to achieve these requirements.

1.3 Problem statement

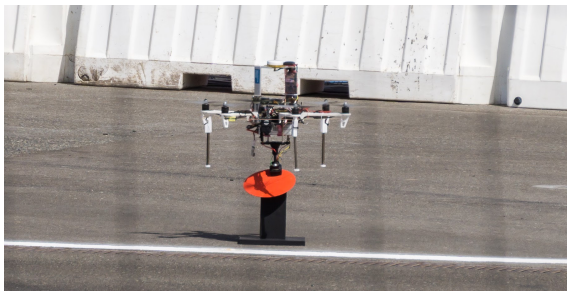
In the MBZIRC 2017 Treasure Hunt challenge, three UAVs (with a maximum size of 120 cm $\times$ 120 cm $\times$ 50 cm) must locate, grasp, deliver, and drop a set of objects into a given box within 20 minutes. The set should contain 10 moving and 10 stationary small objects, as well as 3 stationary large objects, all of which are randomly placed inside the arena. The small objects were $\sim$ 0.370 kg ferrous disks on a stationary stand or moving TurtleBot2 robot, as shown in Fig. 1(b)-1(d). Different colors of the objects – green, blue and red for the static objects, and yellow for the dynamic objects – were associated with different scores, one, two, three, and five points, respectively. The non-stationary objects were moving at random velocities not exceeding 5 km/h. Three large orange objects not exceeding 200 cm in length, and not exceeding 2 kg in weight, were valued at ten points each on successful transport and delivery by at least two cooperating UAVs into the dropping zone depicted in Fig. 1(a). If a large object was moved into the dropping zone by a single



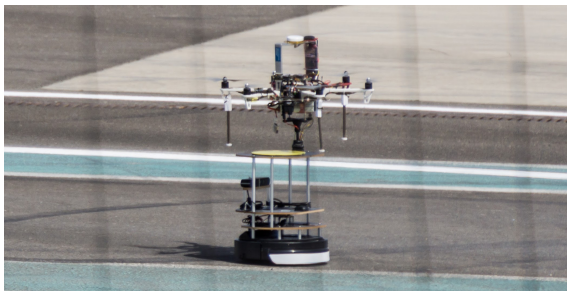
(a) Description of the MBZIRC arena.



(b) Description of the object used during the competition.



(c) Photo of a UAV grasping a static object from the stand.



(d) Photo of a UAV grasping a dynamic object from a TurtleBot2 mobile robot.

Figure 1: Description of the MBZIRC 2017 competition. For more information, visit `http://www.mbzirc.com`.

robot, the team obtained five points. The small objects could be grasped by a single UAV and dropped into a box placed inside the dropping zone. The objects could be picked up by a magnetic gripper, a suction gripper, or another device carried onboard the UAVs. Before the start of each trial, the three UAVs had to be in the start location.

2 Hardware

The specifications of the MBZIRC challenge described above influence the decision on which UAV platform to use. Our intention was to re-use the platforms and the entire system in our follow-up research, and to achieve simple replicability of the system in the future. Therefore, we tried to maximize the use of Commercially Available Off-the-Shelf (COTS) UAV components, and only a few 3D printed specialized tools (sensor holders and the gripper). This approach reduced development time, increased reliability, and now enables our system to be used by other universities with a minimum overhead for technology transfer. It also increases the impact of this paper, which can be considered as a comprehensive manual for building a robust multi-UAV system, even for research groups without any experience with UAVs.

The proposed UAV platform is a complex system composed of integrated active members, computational resources, and sensor modules, shown in a schematic view of the system in Fig. 2. The main structure of each UAV consists of a DJI hexacopter F550 frame and E310 DJI motors. This choice satisfies the size limitations of the MBZIRC event, the flight time, and the payload capability that is necessary for additional sensors, and also for carrying the objects. The system is controlled at the lowest level by a PixHawk flight controller (Meier et al., 2012) that contains a set of sensors, such as accelerometers, gyroscopes and magnetometers, which are necessary for stable UAV flight. The open-hardware and open-software architecture is advantageous for the

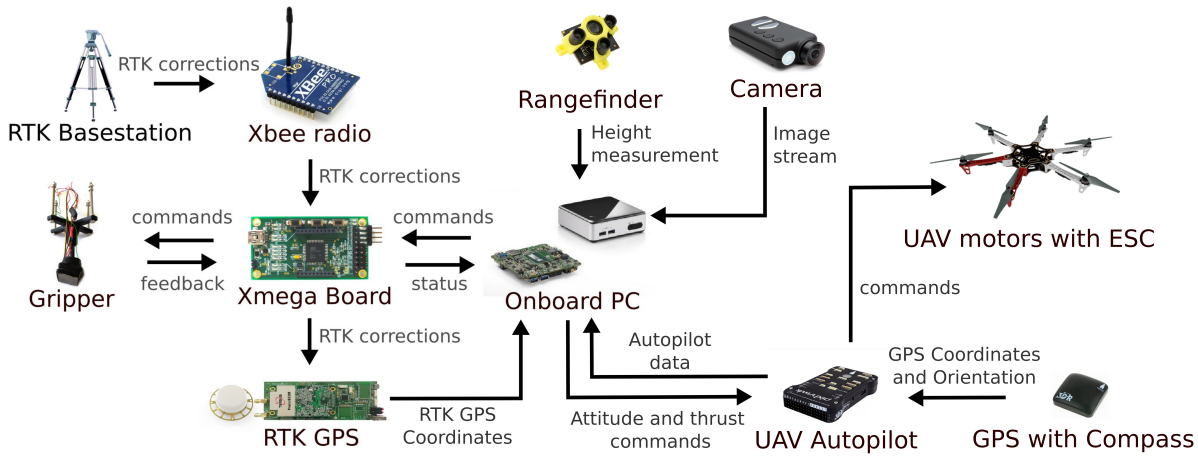


Figure 2: Description of components in our UAV platform.

MBZIRC competition, and also for research on multi-robot systems. An Intel NUC-i7 PC provides sufficient computation power to solve all the required onboard image processing tasks, and also UAV coordination, state estimation, and motion planning in the complex Treasure Hunt challenge. Transport of messages between the onboard PC and PixHawk autopilot is performed over a serial line using MAVlink protocol. Communication between the UAVs, which is important for their coordination, is provided by the WiFi module embedded in the PC. A high-resolution Mobius ActionCam camera is used for object detection, and for relative visual localization.

The rules of the competition allowed the use of GNSS (Global Navigation Satellite System) and even more precise navigation systems for localization. To maximize the accuracy and to increase reliability, our system uses a combination of the Real Time Kinematic (RTK) satellite, which enhances the precision of position data derived from satellite-based positioning systems (*e.g.* GPS, GLONASS, Galileo, and BeiDou), and a classical GNSS module attached to the PixHawk controller. Information on the position is provided in the RTK system by a PRECIS-BX305 GNSS RTK BOARD (GPS L1L2/GLONASS G1/BEIDOU B1B3) (Tersus-GNSS, 2017), with accuracy of 10 millimeters (mm) ± 1 parts-per-million (ppm) horizontally and 15 mm ± 1 ppm vertically when the RTK device is in the most accurate state, RTK FIX. This RTK system requires a stationary GNSS receiver, called RTK basestation, which is placed on a known location. The RTK basestation then broadcasts its position and measurements from all visible satellites (RTK corrections) to the UAVs using XBee Pro radio modules (Digi International, 2017). A custom board was designed to provide communication of the XBee module with the RTK device.

In principle, the vertical position (altitude) provided by the RTK GPS is measured above the mean sea level. However, the UAV does not have any information about the ground level profile or the distance to the objects that are to be grasped, based on the GPS. This information is obtained using the onboard *TeraRanger One* laser rangefinder, which is mounted face-down and is connected directly to the onboard PC, where its data are filtered and used for precise height control. Finally, the objects are grasped using an OpenGrab EPM v3 electro-permanent magnet, which combines the advantages of electro and permanent magnets and creates a very strong magnetic contact with ferrous objects (NicaDrone, 2017). Our custom board (previously mentioned for managing communication from the XBee module into the RTK device) also provides a low-level interface between the main computer and the gripper.

3 Software system structure

The proposed solution relies on the Robot Operating System (ROS), which is an open-source set of software libraries and tools commonly used in the robotic community. Using ROS, the complex MBZIRC tasks can easily be divided into smaller sub-tasks (nodes). This also improves and clarifies the structure of the proposed solution. Furthermore, the Gazebo robotic simulator can be used for Simulation In The Loop (SITL), together with firmware from PixHawk, which provides a very realistic testbed and significantly simplifies testing of the whole system. Using this realistic simulator, hardware experiments could be carried out in a shorter time and in a safer way than if direct HW is used. Because changes were double checked in the simulator, we did not experience any serious crash during more than one year of intensive preparation for the MBZIRC event.

In this section, the sub-components of the proposed system are described. The first two parts explain object detection, object estimation, and motion prediction for dynamic objects. In the next subsection, the estimation of the UAV state from all available sensors is introduced, followed by details on communication in the multi-robot network. Further, the nonlinear controller employed for UAV control is explained, together with the novel MPC-based approach used for online design of a feasible and smooth reference for the nonlinear controller. This is followed by details of high-level planning built upon MPC-based trajectory tracking, which is used for UAV coordination and collision avoidance when the same workspace is shared. Lastly, the FSM, which is crucial for managing all subsystems and for coordinating all UAVs sharing the same workspace, is described. All these sub-components are executed on the onboard PC Intel NUC-i7.

3.1 Object detection

Since the camera that is used to detect the colored objects has a rolling shutter, vibrations induced by the drone motors cause the acquired images to be subject to a specific 'jelly' or 'wobble' effect, which makes the use of geometry-based methods for object detection (e.g. the Hough transform) problematic (Afolabi et al., 2015), see Fig. 3. We therefore designed a computationally efficient ellipse detection algorithm, which relies on the use of statistics that are robust to this type of noise (Krajník et al., 2014). However, the original method described in (Krajník et al., 2014), which used adaptive thresholding to detect black-and-white patterns, had to be extended to process the color information. Since the perceived colors are influenced by

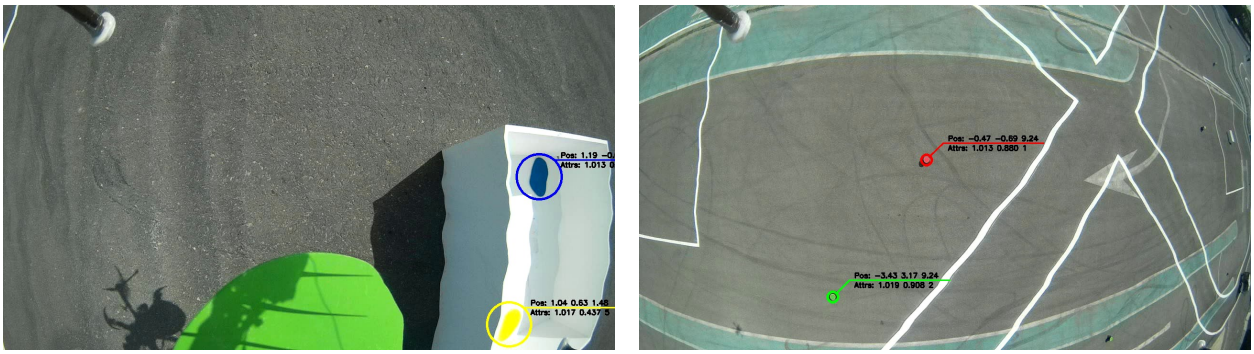


Figure 3: Object detection in on-board camera images affected by the 'jelly' or 'wobble' effect, which deforms lines (left image), as well as circular and square objects (right image). The detection results indicate the 3D relative position (top line) and attributes like roundness, eccentricity and type (1,2,3 for red, green and blue static objects and 5 for the yellow moving object.)

the light conditions, and the exact colors of the objects were not known until the actual contest, we created a semi-automatic autocalibration method that can learn a Gaussian-mixture-based model (GMM) of each color during a short hover over the objects. Once the GMMs are learned, they are used to create an RGB color map, which allows the image pixels to be classified rapidly into object candidates and the background.

The color map is then used in the method (Krajník et al., 2014), which searches for continuous segments of object-colored pixels, establishes their bounding box, the number of pixels, the centroid, convexity and compactness and uses these statistics to reject segments that cannot correspond to circular objects. Then, using the known object size and camera parameters, the method calculates the relative 3D position of the object. This position is then transformed to a global 3D coordinate frame, and objects that do not appear to be close to the ground plane are rejected as false positives. Finally, global 3D positions of the detected objects are forwarded to a mapping module, which integrates multiple detections of the objects into a single 3D representation, which is then used by the planning system.

The performance of the method during tests and in the contest itself indicated computational efficiency and robustness to changing illumination, which was one the key factors in the robustness of the entire system employed in the MBZIRC competition.

3.2 Object estimation and motion prediction

Localization of targets with onboard cameras tends to provide data that are inherently embedded with flaws. The data may be skewed by phenomena such as *signal noise*, *false positive detections*, *irregular detection rate*, *data blackouts*, etc. These issues can hardly be mitigated during the detection, and some of them (e.g. data blackouts) also depend on the external environment. Moreover, several moving targets appear in the MBZIRC challenge and so estimates of unobserved states such as velocities and heading may help to follow their position precisely. This leads to a need to filter the detected position of the targets. We also required the filtration system to be capable of sorting out measurements belonging to targets that have been marked as *unreliable*, for example due to data blackout being too frequent. Another requirement comes from the multi-robot nature of the task. A UAV should share information about parts of the map that are currently occupied. Targets in those areas should then be filtered out in other UAVs to prevent unrequired grasping of the same target by multiple UAVs.

In the event that there is a single target in the field of view of the UAV, an unscented Kalman Filter is used as a filter and as a predictor in conjunction with the car-like motion model

$$\begin{aligned}
\mathbf{x}^o_{[n+1]} &= \mathbf{x}^o_{[n]} + \dot{\mathbf{x}}^o_{[n]}\Delta t, \\
\dot{\mathbf{x}}^o_{[n+1]} &= \begin{pmatrix}\cos\phi_{[n+1]} \\ \sin\phi_{[n+1]}\end{pmatrix} v_{[n+1]}, \\
\phi_{[n+1]} &= \phi_{[n]} + \dot{\phi}_{[n]}\Delta t, \\
\dot{\phi}_{[n+1]} &= K_{[n]} v_{[n]}, \\
v_{[n+1]} &= v_{[n]} + a_{[n]}\Delta t, \\
K_{[n+1]} &= K_{[n]} + \dot{K}_{[n]}\Delta t,
\end{aligned} \tag{1}$$

where $\mathbf{x}^o_{[n]} = (x, y)^T_{[n]}$ is the position of the object in the global coordinate system, $\phi_{[n]}$ is its heading, $K_{[n]}$ is the curvature of its turn, $v_{[n]}$ is its scalar velocity, $a_{[n]}$ is its scalar acceleration, and Δt is the time difference. An estimate of the target heading allows its motion to be tracked, while the onboard camera is oriented with its wider field of view in favor of detecting sudden changes of the object's heading.

However, real-world scenarios might contain several objects in the field of view, while some of them are moving. In that case, the UAV needs to track a particular object independently of the movement of all the objects in the scene. This requires a local map of the objects to be actively maintained. Our map model was based on eq. (1) for an arbitrary number of independent objects. Another state has been included to cover the type of the object (its color and whether it is moving) as well as the time of its last update and whether it is currently active. Manipulation of the objects in the map obeys the following principles:

- An object that has not been seen for more than 5 seconds is *deactivated*. Deactivated objects stay in the map, but their movement is no longer predicted by the UKF.

- Objects that are deactivated for more than 3 seconds are deleted from the map.
- Measurements from the *object detector* (section 3.1) are paired with objects in the map using min-distance bipartite graph matching, constrained by the color of the objects.
- Objects located outside of the competition arena or in any of the locally banned areas (near the dropping zone or around other UAVs) are deleted from the map, and new measurements in these areas are discarded.

Additionally, it can be anticipated that grasping attempts may not be successful at all times. The filter allows a temporarily ban on an area around a particular object, to avoid deadlock in the grasping state machine. Such a ban is valid for 30 s in a radius of 4 m around the object.

3.3 UAV position estimation

Automatic control of UAVs relies on estimates of the states of the UAV dynamical system. Namely, knowledge of position and velocity (both vertical and horizontal) is required to coordinate the movement for precise picking up and delivery of the object. Our platform is equipped with several independent sources of information, which are fused to obtain a single, reliable and smooth estimate of the UAV pose. An important requirement is to ensure smoothness of the resulting signal, since the $SO(3)$ state feedback is sensitive to noise.

The main source of data for both the vertical and the horizontal axes is the PixHawk flight controller. Its Extended Kalman Filter fuses present-day inertial sensors – a three-axis accelerometer and a gyroscope with an altitude pressure sensor and a GPS receiver. Although the aircraft is already capable of autonomous flight with this off-the-shelf setup, we make use of other sensors to provide more precise localization and thus better precision of object manipulation.

3.3.1 Horizontal position estimation

The position estimates in the lateral axes are based on the estimate provided by PixHawk, namely positions $\mathbf{x}^p$, and velocities $\dot{\mathbf{x}}^p$. Although the precision of the estimates may be satisfactory locally for short time intervals, they are prone to significant drift in time spans of minutes. To correct this drift, and thus to ensure repeatability of the experiments and e.g. locating the dropping zone, the horizontal position from PixHawk is corrected by differential RTK GPS. Position measurements from the RTK GPS receiver are fused using the Linear Kalman Filter with the model

$$\mathbf{A} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \mathbf{B} = \begin{pmatrix} \Delta t \\ \Delta t \end{pmatrix}, \tag{2}$$

where $\mathbf{x}^e_{[n+1]} = \mathbf{A}\mathbf{x}^e_{[n]} + \mathbf{B}\mathbf{u}_{[n]}$ is the linear system equation, $\mathbf{x}^e_{[n]} = (x, y)^T_{[n]}$ is the state vector finally used for control, and $\mathbf{u}_{[n]}$ is the system input. According to our experience,

$$\mathbf{x}^p_{[0]} + \sum_{n=0}^{k} \dot{\mathbf{x}}^p_{[n]} \Delta t_{[n]} = \mathbf{x}^p_{[k]}, \forall k \in \mathbb{N} \tag{3}$$

does not hold for the position and velocity estimate provided by PixHawk. This is a very useful observation for somebody building a fully autonomous UAV system using an off-the-shelf controller. The input vector $\mathbf{u}$ consists of velocities obtained as differentiated positions $\mathbf{x}^p$ (later integrated by the filter), which ensures that the proposed filter does not introduce any drift into the resulting estimate when no RTK GPS corrections are received. In situations when the position is not being corrected, the resulting estimate follows the same relative state trajectory as $\mathbf{x}^p$, just shifted according to the latest correction. The final tuning of the filter resulted in process covariance $\mathbf{Q} = \mathrm{diag}\,(1, 1)$ and measurement covariance $\mathbf{R} = \mathrm{diag}\,(10e3, 10e3)$. Moreover,

the RTK GPS corrections were saturated to ever impose maximally 0.25 m difference from the internal state of the filter. Such technique limits sudden changes of the estimated position, which was necessary for safety of the flight.

The multi-robotic scenario requires a coordinate space to be shared among all three UAVs. The base of our Cartesian system is set to predefined GPS coordinates and its orientation is according to the *ENU* convention. Therefore, the 1st, 2nd and 3rd axis point to the East, North and Upwards, respectively. A point of origin is measured using the RTK GPS, to which all independent coordinate systems of all UAVs are then shifted after each of them is powered up. The common base station of the differential RTK GPS then ensures that all UAV estimates are corrected to lie within the same global coordinates.

3.3.2 Vertical position estimation

In contrast with the horizontal position, estimates of the height rely much less on PixHawk. The linear Kalman Filter for the vertical axis also uses the differentiated PixHawk height in the same manner as the horizontal axis. However, height corrections come not only from the differential RTK GPS, but also from the down-facing TeraRanger rangefinder and the object detector, which is able to provide an estimate of the relative distance, when flying above an object. The estimator provides an option to switch between these sources of data, depending on the current task and the circumstances.

It is feasible to correct the height using the TeraRanger rangefinder, when flying above uneven ground, but it cannot be used reliably when the down-facing sensor is obstructed e.g., when carrying an object, or when there might be a foreign object on the ground, namely the dropping box. RTK GPS can provide precise relative height measurements, but only when RTK FIX has been established. This depends on the strength of the GNSS signal and on the quality of the communication link between the base station and the UAV. Finally, correcting the altitude using data from the *object detector* may bring in unexpected steps in the signal due to false positive detections. Since none of the additional sources is completely reliable, we implemented a safety mechanism for detecting anomalies, which can toggle off any of the above-mentioned sensors from being fused.

3.4 Communication between UAVs

In multi-robot systems, reliable communication is required mainly if there is a need for direct cooperation between multiple autonomous vehicles, as in the case when large objects are to be carried cooperatively. However, a reliable communication channel is a crucial tool even for coordinating the UAV team sharing the same workspace for grasping small objects individually, as was demonstrated in the MBZIRC competition. The rules of the MBZIRC event specified that all teams are obliged to share the same 5GHz Wi-Fi network, the reliability of which was influenced by interference occurring during transmission. This may easily lead to packet loss, which can interrupt the connection. Decreased reliability of the communication link during the entire mission is not limited to the MBZIRC case. It is a typical feature of most UAV applications in demanding outdoor conditions. The MBZIRC contest therefore provided an interesting and realistic evaluation scenario for multi-UAV systems, in which it cannot be assumed that a complete communication network is available at all times. In our opinion, our system achieved significantly better performance in the multi-UAV scenario than the other teams, due to the following strategy. We attempted to maximize utilization of the communication channel, if it was available, in order to achieve optimal behavior of the system. However, it was important to be able to degrade into a system not relying on the communication infrastructure at all. This was done at the cost of decreased performance, but our system still provided safe flight operation of multiple UAVs solving the given task. A smooth and possibly repeated transition between the optimal behavior relying on communication and the non-optimal but safe and still working system without communication, and back, is provided by the FSM approach described in section 3.8.

The software part responsible for managing communication between UAVs is based on the ROS master

within the ROS network. To increase the robustness of the communication net in the event of a failure of the robot that is the leader in the ROS master scheme, the proposed method relies on multiple independent ROS masters assigned to each of the UAVs. The ROS package multimaster_fkie (Tiderko, 2017) is used to maintain communication between these ROS masters. This package offers a set of nodes to establish and manage a multimaster network, which is necessary for such tasks with the team of UAVs in the event of an unreliable communication infrastructure.

To reduce the load of the communication channels managed by the ROS master network, only selected information (topics) are exchanged between the team members:

- the actual position of the UAV in the global coordination system,
- the actual state of the high-level state machine being part of the FSM,
- the estimated position of the object during grasping,
- the planned trajectory.

These topics are used in nodes for proactive collision-free planning, fail-safe reactive collision avoidance, and object estimation. The bandwidth of the Wifi network necessary for transmission of all mentioned information for a single UAV is ~ 10 kB/s.

3.5 Low-level UAV control

The position controller uses the estimated state as feedback to follow the trajectories given as an output of the high-level trajectory planner. In many previous works, a backstepping approach is used for UAV control, because the attitude dynamics can be assumed to be faster than the dynamics governing the position, so linearized controllers are used for both loops (Mellinger et al., 2013; Weiss et al., 2011; Herissé et al., 2012). However, we need the system to be capable of large deviations from the hover configuration during operations like fast mapping of objects, or for strong wind compensation. We therefore use a nonlinear controller. Let us consider an inertial reference frame denoted by $\begin{bmatrix}\mathbf{e}_1 \ , \ \mathbf{e}_2 \ , \ \mathbf{e}_3\end{bmatrix}$ and a body reference frame centered in the center of mass of the vehicle with an orientation denoted by $\mathbf{R} = \begin{bmatrix}\mathbf{b}_1 \ , \ \mathbf{b}_2 \ , \ \mathbf{b}_3\end{bmatrix}$, where $\mathbf{R} \in SO(3)$. The dynamic model of the vehicle can be expressed as

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{v}, \\ m\dot{\mathbf{v}} &= f\mathbf{R}\mathbf{e}_3 + mg\mathbf{e}_3, \\ \dot{\mathbf{R}} &= \mathbf{R}\hat{\mathbf{\Omega}}, \\ \mathbf{J}\dot{\mathbf{\Omega}} + \mathbf{\Omega} \times \mathbf{J}\mathbf{\Omega} &= \mathbf{M}, \end{aligned} \tag{4}$$

where $\mathbf{x} \in \mathbb{R}^3$ is the Cartesian position of the vehicle expressed in the inertial frame, $\mathbf{v} \in \mathbb{R}^3$ is the velocity of the vehicle in the inertial frame, $m \in \mathbb{R}$ is the mass, $f \in \mathbb{R}$ is the net thrust, $\mathbf{\Omega} \in \mathbb{R}^3$ is the angular velocity in the body-fixed frame, and $\mathbf{J} \in \mathbf{R}^{3\times3}$ is the inertia matrix with respect to the body frame. The hat symbol $\hat{}$ denotes the skew-symmetry operator according to $\hat{\mathbf{x}}\mathbf{y} = \mathbf{x}\times\mathbf{y}$ for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^3$, g is the standard gravitational acceleration, and $\mathbf{e}_3 = \begin{bmatrix}0 & 0 & 1\end{bmatrix}^\top$. The total moment $\mathbf{M} \in \mathbb{R}^3$, with $\mathbf{M} = \begin{bmatrix}M_1 & M_2 & M_3\end{bmatrix}^\top$, along all axes of the body-fixed frame and the thrust $\tau \in \mathbb{R}$ are control inputs of the plant. The dynamics of the rotors and propellers are neglected, and it is assumed that the force of each propeller is directly controlled. The total thrust, $f = \sum_{j=1}^{6} f_j$, acts in the direction of the z-axis of the body-fixed frame, which is orthogonal to the plane defined by the centers of the six propellers. The relationship between a single motor thrust f_j,

the net thrust f, and the moments $\mathbf{M}$ can be written as

$$\begin{bmatrix} f \\ M_1 \\ M_2 \\ M_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ sd & 1 & sd & -sd & -1 & -sd \\ -cd & 0 & cd & cd & 0 & -cd \\ -1 & 1 & -1 & 1 & -1 & 1 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \end{bmatrix} \tag{5}$$

where $c = \cos(30°)$, $s = \sin(30°)$ and d is the distance from the center of mass to the center of each rotor in the $\mathbf{b}_1$, $\mathbf{b}_2$ plane. For non-zero values of d, eq. (5) can be inverted using the right pseudo-inverse.

For control, we build on the work in (Lee et al., 2013) and in (Mellinger and Kumar, 2011) with control inputs $f \in \mathbb{R}$ and $\mathbf{M} \in \mathbb{R}^3$ chosen as

$$\mathbf{M} = -k_R \mathbf{e}_R - k_\Omega \mathbf{e}_\Omega + \boldsymbol{\Omega} \times J\boldsymbol{\Omega} - J\left(\hat{\boldsymbol{\Omega}}\mathbf{R}^T\mathbf{R}_c\boldsymbol{\Omega}_c - \mathbf{R}^T\mathbf{R}_c\dot{\boldsymbol{\Omega}}_c\right), \tag{6}$$

$$f = -\left(-k_x\mathbf{e}_x - k_{ib}\mathbf{R}\int_0^t \mathbf{R}(\tau)^T\mathbf{e}_x d\tau - k_{iw}\int_0^t \mathbf{e}_x d\tau - k_v\mathbf{e}_v - mg\mathbf{e}_3 + m\ddot{\mathbf{x}}_d\right)\cdot\mathbf{R}\mathbf{e}_3 = \mathbf{f}\cdot\mathbf{R}\mathbf{e}_3, \tag{7}$$

with $\ddot{\mathbf{x}}_d$ the desired acceleration, and k_{iw}, k_{ib}, k_x, k_v, k_R, k_Ω positive definite terms. We extend the referenced controllers by including two integral terms which accumulate the error in the body frame and in the world frame, respectively. We include both terms to provide the opportunity to capture external disturbances (*e.g.,* wind) separately from internal disturbances (*e.g.,* an inefficient prop or a payload imbalance), particularly when the vehicle is permitted to yaw or rotate about the vertical axis. The thrust and the moments are then converted to motor rates according to the characteristic of the proposed vehicle. Subscript C denotes a commanded value, and $R_C = \begin{bmatrix}\mathbf{b}_{1,C}, & \mathbf{b}_{2,C}, & \mathbf{b}_{3,C}\end{bmatrix}$ is calculated as

$$\mathbf{b}_{2,des} = \begin{bmatrix}-\sin\psi_{des}, & \cos\psi_{des}, & 0\end{bmatrix}^\top,\ \mathbf{b}_{3,C} = \frac{\mathbf{f}}{||\mathbf{f}||}, \quad \mathbf{b}_{1,C} = \frac{\mathbf{b}_{2,des}\times\mathbf{b}_3}{||\mathbf{b}_{2,des}\times\mathbf{b}_3||}, \quad \mathbf{b}_{2,C} = \mathbf{b}_3\times\mathbf{b}_1,$$

Note that here we have to define $\mathbf{b}_{2,des}$ based on the yaw, instead of defining $\mathbf{b}_{1,des}$ as it was defined in (Mellinger and Kumar, 2011), due to a different Euler angle convention (we use the ZYX convention instead of ZXY). The quantities

$$\begin{gathered} \mathbf{e}_R = \frac{1}{2}\left(\mathbf{R}_C{}^\top\mathbf{R} - \mathbf{R}^\top\mathbf{R}_C\right)^\vee,\ \ \mathbf{e}_\Omega = \boldsymbol{\Omega} - \mathbf{R}^\top\mathbf{R}_C\boldsymbol{\Omega}_C, \\ \mathbf{e}_x = \mathbf{x} - \mathbf{x}_d,\ \ \mathbf{e}_v = \dot{\mathbf{x}} - \dot{\mathbf{x}}_d, \end{gathered} \tag{8}$$

represent the orientation, the angular rate errors, and the translation errors, respectively. The symbol $.^\vee$ represents the *vee* map $\mathfrak{so}(3) \to \mathbb{R}^3$. If the initial attitude error is less than 90°, the zero equilibrium of the tracking error is exponentially stable, *i.e.*, $\begin{bmatrix}\mathbf{e}_x^\top & \mathbf{e}_v^\top & \mathbf{e}_\Omega^\top & \mathbf{e}_R^\top\end{bmatrix}^\top \equiv \begin{bmatrix}\mathbf{0}^\top & \mathbf{0}^\top & \mathbf{0}^\top & \mathbf{0}^\top\end{bmatrix}^\top$.

3.6 Trajectory tracking

The state feedback, described in section 3.5, which provides precise position and velocity control, requires a smooth and feasible reference. The reference consists of all states of the translational dynamics – position, velocity and acceleration – and is provided at 100 Hz, the same rate as the resulting control signal. There are various ways to create the reference. Typically, thanks to the differential flatness of the UAV dynamical system, a QP optimization can be solved to find a polynomial given the initial and final state conditions (Mellinger and Kumar, 2011), which can then be sampled to create the reference. In our case, we chose to generate the reference using a Model Predictive Control approach. MPC ensures that the resulting trajectory satisfies a given model as well as the dynamical constraints, which are imposed on the model. As it optimizes control actions over a prediction horizon, it can react adequately to unfeasible changes in the

reference trajectory, and can also create proper feed-forward pro-actions to minimize the control error in the future.

The Model Predictive Control Tracker (MPC Tracker) uses a QP formulation of a minimal sum-of-squares problem, where the optimal control action $\mathbf{u}$ is found for a future prediction horizon of states $\mathbf{x}_{[n]} = (x, \dot{x}, \ddot{x}, y, \dot{y}, \ddot{y}, z, \dot{z}, \ddot{z})^T_{[n]}$ by minimizing the function

$$\begin{aligned} \mathrm{V}\left(\mathbf{x}_{[0,\dots,m-1]}, \mathbf{u}_{[0,\dots,m-1]}\right) = \quad & \frac{1}{2}\sum_{i=1}^{m-1}\left(\mathbf{e}^T_{[i]}\mathbf{Q}\mathbf{e}_{[i]} + \mathbf{u}^T_{[i]}\mathbf{P}\mathbf{u}_{[i]}\right), \\ \text{s.t.} \quad & \mathbf{x}_{[0,\dots,m-1]} \geq \mathbf{x}_L, \\ & \mathbf{x}_{[0,\dots,m-1]} \leq \mathbf{x}_U, \end{aligned} \tag{9}$$

where $\mathbf{e}_{[n]} = \mathbf{x}_{[n]} - \tilde{\mathbf{x}}_{[n]}$ is the control error, $\tilde{\mathbf{x}}_{[n]}$ is the setpoint for the MPC, m is the length of the prediction horizon, and $\mathbf{x}_L$ and $\mathbf{x}_K$ represent box constraints on states. The control error $\mathbf{e}_{[n]}$ requires the formation of a general prediction of $\mathbf{x}_{[n]}$, which was described previously in (Baca et al., 2016). In our case, the optimized control action is not directly used to control the real UAV. Instead, it controls a model of the UAV translational dynamics in real-time simulation. States of the simulated model are then sampled at 100 Hz to create the reference for the state feedback. This is a novel approach in UAV control, where benefits of both nonlinear control and linear MPC are used together.

An important notion is the difference between the trajectory setpoint $\tilde{\mathbf{x}}$ and the reference, which is generated by the MPC tracker. The trajectory setpoint $\tilde{\mathbf{x}}$ is provided by an operator or a program. No requirements are imposed on $\tilde{\mathbf{x}}$. In contrast, the reference produced by the MPC Tracker is feasible, satisfies the UAV dynamics and state constraints, and serves as a control reference for the $SO(3)$ state feedback.

The simulated model is an LTI system covering the 3rd order translational dynamics of the UAV with sampling of $\Delta t = 0.01$ s. In our MPC formulation, Δt is different for the first iteration ($\Delta t = 0.01$ s) and for all the other iterations ($\Delta t = 0.2$ s). This allows smooth control of the simulation, if the MPC is executed at 100 Hz, while there is a relatively sparse distribution of further states, which allows us to have a much longer prediction horizon than there would normally be with Δt being constant. As in traditional MPC, only the control action in the step is used to control the model in the simulation. In the meantime, a new instance of the optimization task is formulated, starting from new initial conditions, which results in a fresh control action for the next step. This method is valid only if the MPC can be solved repeatedly within 0.01 s.

The penalization parameters $\mathbf{Q}$ and $\mathbf{P}$ in eq. (9) were found empirically. As in our previous work (Baca et al., 2016), we used the *move blocking* technique to effectively prolong the prediction horizon while maintaining the computational complexity. The particular control action distribution for the MBZIRC competition was as

$$\mathbf{U} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 5 & 5 & 5 & 5 & 5 & 10 \end{pmatrix}, \tag{10}$$

which results in an 8 s prediction horizon with only 33 variables in the optimization task.

Creating the control reference for the state feedback with MPC has several advantages over conventional solutions. It produces a reference that is feasible according to the specified model, which makes it safe to execute. If the setpoint for MPC is not feasible, the resulting reference is feasible with respect to eq. (9). The inherent predictive nature of MPC provides trajectory tracking optimizing actions over the future, which makes it ideal for tracking moving targets, such as the moving objects in the competition.

As defined in eq. (9), MPC handles state constraints as linear constraints. We impose maximum acceleration and velocity box constraints on the UAV to ensure safe and feasible resulting trajectories. The optimization being solved lies in the family of Linearly Constrained Quadratic Programming, which acquires a global optimum in a convex polytope. A custom solver, based on a sequential closed-form solution, has been implemented to ensure guaranteed real-time performance.

MPC-based trajectory tracking operates in two modes, as follows. The first simple positioning mode, used mainly for short distance position changes, applies either relative or absolute position commands, and tries to reach a given position in the fastest way with respect to the MPC scheme. The second trajectory-following mode used by high-level trajectory planning (section 3.7) uses a pre-computed path plan, and tries to precisely track the trajectory while respecting the plan waypoints schedule, which is crucial for multi-robot collision-free operation.

Having the predictions of the future movement for all UAVs allows us to extend the capability of the MPC Tracker to avoid future collisions. When communication between the aircraft is established, they exchange their future trajectory predictions and act according to a decentralized mechanics, which will alter their courses to avoid the collision, based on sorting the UAVs by priorities. If there is a potential collision between two UAVs, the UAV with lower priority will avoid the other UAV by changing to a higher flight level. The system also allows priorities to be reassigned dynamically in the following cases:

- UAV should be avoided at all times (its priority is higher by definition). This may occur when it is currently grasping an object, or when its avoidance mechanism is accidentally turned off.
- UAV should avoid the other aircraft even if it has higher priority. Such a situation occurs when the other machine does not comply with the mechanics for any reason.

3.7 High-level trajectory planning

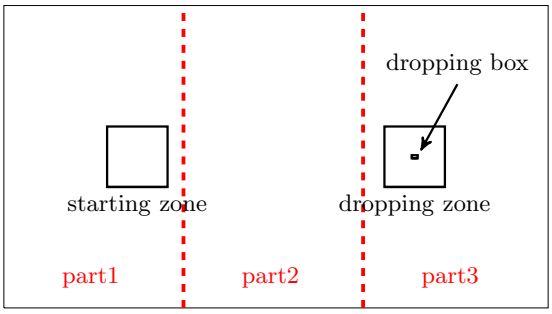


Figure 4: Decomposition of the MBZIRC arena into three equally large zones.

High-level trajectory planning is built on top of MPC-based *Trajectory tracking*, which is employed for precise tracking of the planned trajectories. The onboard online trajectory planning mechanism is used in two main parts of the Treasure Hunt scenario. The first task is **Sweeping** of the arena, where the team of UAVs is required to localize the objects within the arena, and either save their locations to the global map (at the beginning of the mission) or immediately try to grasp the first detected object (later in the mission, once all objects detected in the initial map have been processed - the grasping was successful, or failed repeatedly). The second online trajectory planning is utilized in **Proactive collision-free planning**, which is involved in cases where one UAV has to fly into another position. For example, when it holds the grasped object and wants to drop it into the dropping box.

3.7.1 Sweeping

Sweeping the arena designed for the MBZIRC Treasure Hunt challenge involves localizing both dynamic and static objects. The trajectory planning for so-called sweeping can be described as Coverage Path Planning (CPP) (Galceran and Carreras, 2013), where for a given area the CPP should find a path from which the entire workplace can be scanned with an onboard sensor, in our case an onboard camera.

The proposed multi-robot coverage path planning algorithm is based on simple area decomposition into three equally large zones that split the area along the larger side (see Fig. 4). Each arena zone has one UAV

assigned to localize and pick up the objects from. All UAVs then plan the coverage path using Boustrophedon coverage (Choset and Pignon, 1998) in each part of the area separately. Using Boustrophedon coverage we create zigzag paths, as shown in Fig. 5, such that the reduced field of view (FOV) entirely covers the particular arena zone. The reduced field of view is set based on the required overlap in the coverage (set to 20% during the competition) and on the real FOV camera projection to the ground plane with respect to the sweeping altitude that is used.

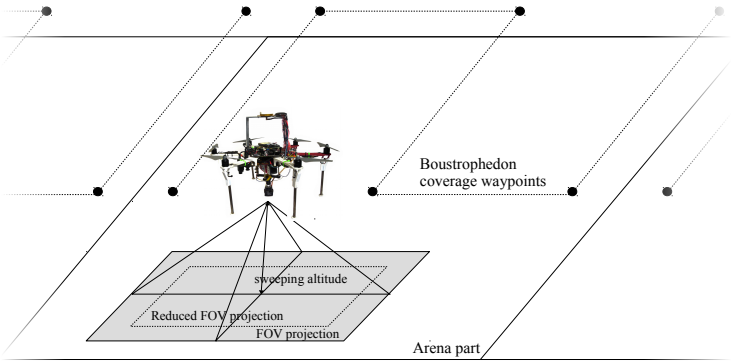


Figure 5: Boustrophedon coverage of the decomposed competition arena.

In order to produce smooth trajectories for constant speed object detection, the Dubins vehicle model (Dubins, 1957) is employed to create the final path between the waypoints. The minimal turning radius $\rho = v_c^2/a_{max}$ of the Dubins vehicle was selected based on the desired constant velocity v_c ($\sim 3\ \mathrm{m\,s^{-1}}$) and the maximal acceleration of the UAV a_{max} ($\sim 2\ \mathrm{m\,s^{-2}}$), using an equation of circular motion with constant speed. The sweeping high-level trajectory planning is summarized in Fig. 6, where the shown trajectories for all three UAVs were further used in the two following approaches in different stages of the Treasure Hunt scenario.

In the first approach, called static sweeping, the UAVs follow the created trajectories at a height (~ 7 m) and simultaneously detect the colored objects while the global map of the static objects is being created. After this initial coverage, the approximate positions of the detected static objects are estimated based on multiple detections of the same object. The second approach, called dynamic sweeping, is applied later in the schedule of the task, and the UAVs use similar paths as in the static sweeping. However, the sampled trajectories are used repeatedly (not just once, as in the static sweeping) and the UAVs do not create a global map. Instead, each UAV tries to find and estimate the position of any object while following the sweeping trajectory. When any object is located, the trajectory following is stopped and the UAV tries to grasp the object immediately. Either after successful grasping and dropping of the object, or after a number of unsuccessful grasps, the UAV continues with dynamic sweeping from the last trajectory sample.

3.7.2 Proactive collision-free planning

Our strategy for covering the Treasure Hunt competition arena is based on decomposition into three equally large zones for each of the UAVs (see Fig. 4). Unfortunately, the dropping zone is located in one third of the competition arena. After successful grasping, the UAV in part 1 therefore has to fly through the remaining zones to drop the object. Because there is a possibility of colliding with another UAV during this flight through the remaining zones, proactive collision-free planning has to be used. The actual positions of the UAVs are known due to information sharing, as was explained in section 3.4. However, the Wi-Fi communication infrastructure is not reliable and, as mentioned, a multi-robot system deployed in real world conditions should be robust to losing Wi-Fi communication. Therefore, we decided to use different flying heights for each of the UAVs, which minimizes the possibility of a collision, without any additional planning. Unfortunately, while completing this task the UAVs cannot maintain only these heights during the mission, as they have to descend for events such as grasping the objects and then dropping them. These events take

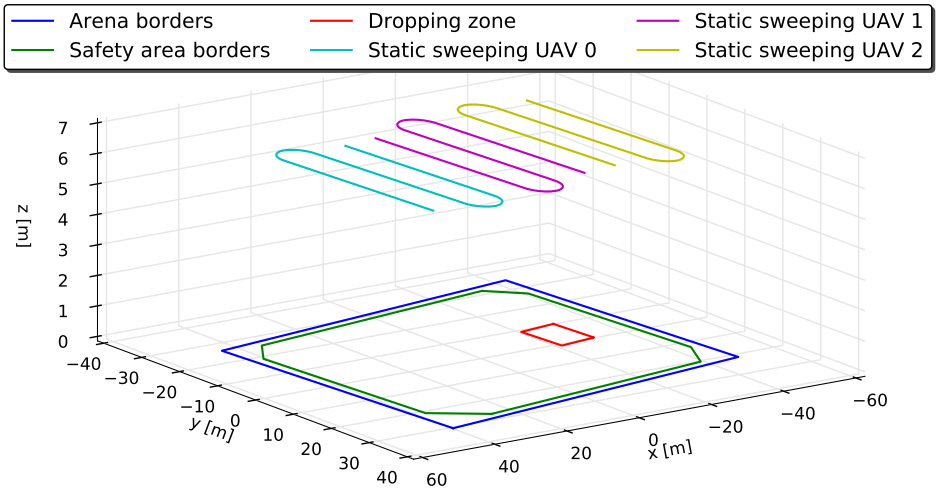


Figure 6: Sweeping trajectories based on Boustrophedon coverage using the Dubins vehicle and decomposition of the arena into three distinct parts, one for each UAV.

(a) A path without obstacles. (b) A path with one obstacle. (c) A path with two obstacles. (d) A path with two obstacles.

Figure 7: Examples of trajectories generated by fast proactive collision-free planning. The black circles denote obstacles. The red path shows the shortest collision-free trajectory, and the gray paths denote other collision-free paths.

most of the overall flight time, because they require a complicated grasping manoeuvre and hovering in front of the dropping zone, if it is sharing with other UAVs. Moreover, the grasping manoeuvre can be repeated several times before the object is gripped.

The proposed solution for finding a collision-free trajectory uses four assumptions derived from the MBZIRC rules, which are, however, valid for most cooperative transport applications:

- A Collision can occur only if a UAV leaves its dedicated height.
- The position of the UAV in the x-axis and in the y-axis does not alter rapidly in the event that it flies out of its safe altitude (the grasping and dropping manoeuvres are carried out following strictly vertical trajectories that accept grasping of dynamic objects, but where the lateral movement is also minor).
- The shape of the competition workspace is convex.
- At most three UAVs are employed in the environment (this assumption is valid only for the MBZIRC Treasure Hunt task, but an extension of the approach is straightforward for different numbers of robots).

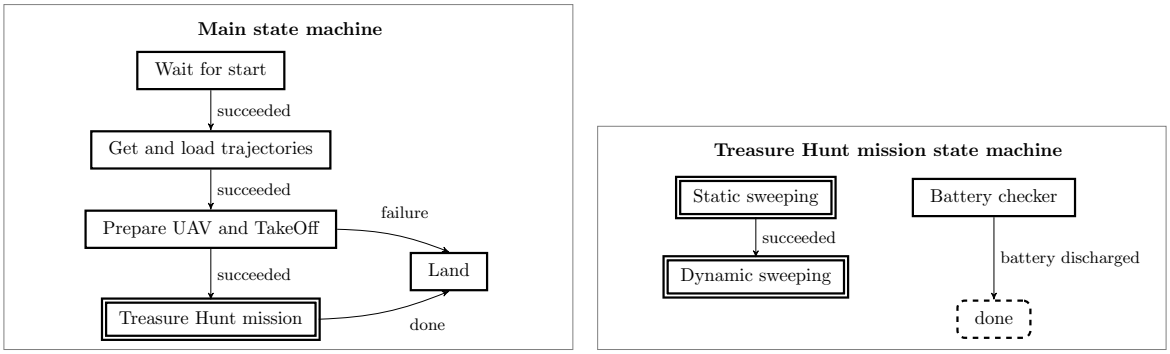
Thanks to these assumptions, the method for very rapidly computing a collision-free trajectory can be simplified to finding a collision-free path in 2D (at the dedicated height) between two points, where only two obstacles can occur. These obstacles are circles centered on the x and y coordinates of neighboring

UAVs with safety radius r_a. It is prohibited to encroach on these circles. The safety radius of the circles depends on the speed of the UAVs which, for security reasons in the MBZIRC competition, was restricted to a maximum of 30 km/h. We used a detection radius (the relative distance between UAVs in which the avoidance maneuver is initiated) of 5 m radius during the competition, while the critical radius in which the UAVs are considered to be in a collision is 0.8 m.

Based on the previously realized experimental comparison of available path planning approaches (Saska et al., 2006), a visibility graph method (Lozano-Pérez and Wesley, 1979) was applied to solve the collision free planning problem. The method provides the shortest path and it is sufficiently fast in simple situations including limited number of obstacles. Only four possible paths in the graph consisting of tangent lines to circles, which represent the obstacle, and the circle segments can be considered as a candidate solution in our case of two obstacles. The solution can, therefore, be found analytically in a very short time (possibly in each control step) with negligible burden on the processor. See examples of trajectories generated by proactive collision-free planning in Fig. 7(a) to Fig. 7(d).

A collision-free trajectory exists only for described planning when the start points or the end points are not inside the safety radius r_a of another UAV. In situations when a UAV is already inside the safety radius r_a of another UAV, the UAV finds a plan into the nearest position that is not in conflict with a UAV, and the collision-free planning procedure is initiated. If the high-level planning system requires to fly into a position, which is occupied by another UAV, then a temporary goal position is set instead. This position is the closest feasible position to the original goal such that it lies on the original trajectory. The UAV then waits for up to 1.5 min until the goal position is available again. If the goal position is not freed within this time, it is assumed that the information about the occupation of the goal position is incorrect. During the MBZIRC competition, the planning was repeated five times per second, and in the event of a communication interruption, the last received states of other UAVs were considered as correct for 5 s.

3.8 Failure recovery and synchronization jobs manager



(a) Diagram of the main state machine.

(b) Diagram of the Treasure Hunt mission state machine.

Figure 8: The structure of the FSM tool.

The main core of the system is the FSM concept, which is used for managing all subsystems. It increases the robustness of the entire code structure resolving the remaining few subsystem failure cases due to wrong sequential and concurrent operations. In the proposed system, the FSM is designed using SMACH, a ROS-independent Python library (Bohren, 2017), and it is fully integrated into the designed ROS framework.

As was mentioned in the introduction, the entire FSM structure may be considered as a hierarchical state machine with synchronization and failure recovery abilities. For simplicity, we will refer to the components of the FSM as state machines in this section. In Figs. 8-10 and Fig. 12, the internal states of the FSM levels (the so-called state machines) are visualized by rectangles, and the nested lower-level state machines are visualized by double line rectangles, such as the *Treasure Hunt mission* state machine introduced in Fig. 8(a) by the

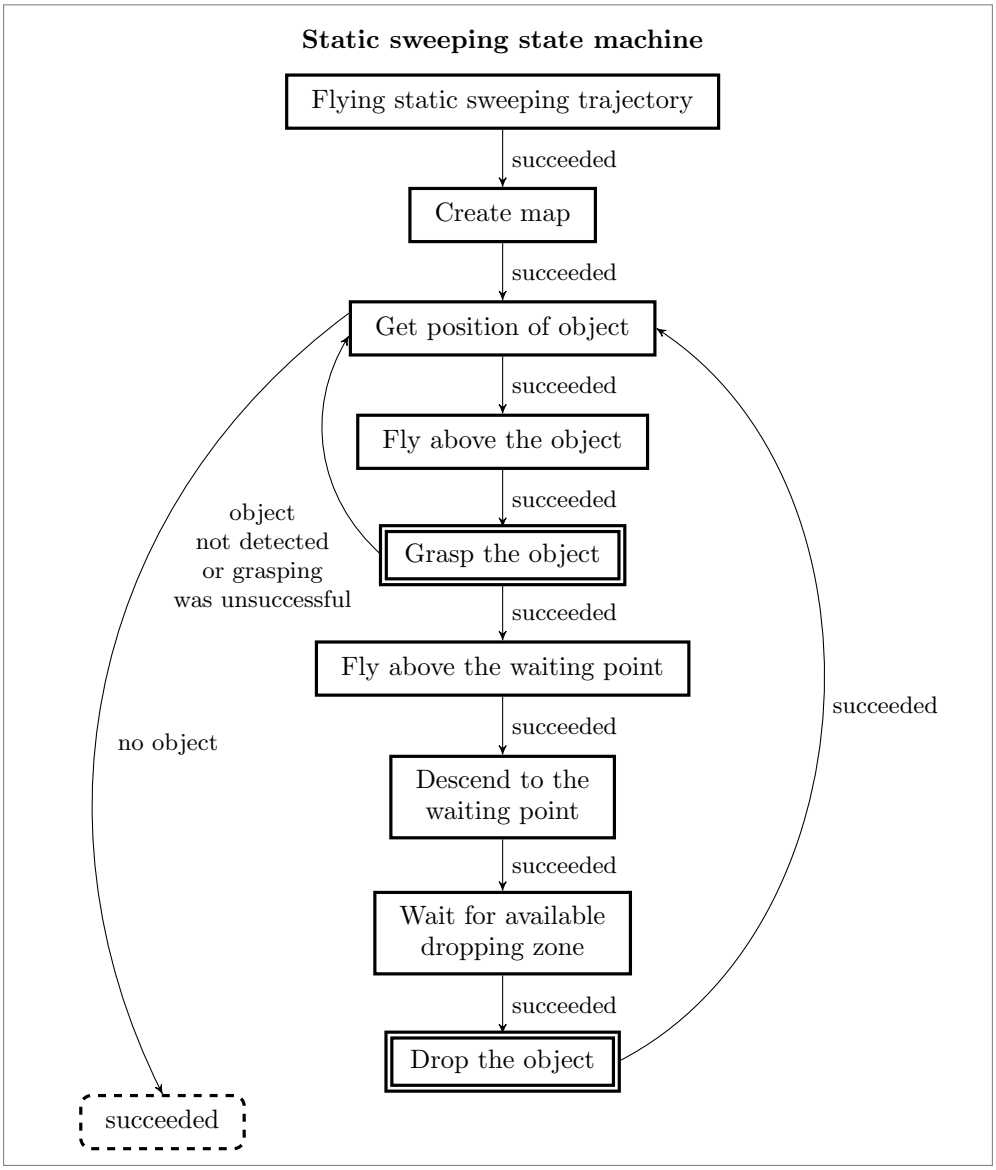


Figure 9: Diagram of the static sweeping state machine.

Treasure hunt mission rectangle, and described in detail in Fig. 8(b). Transitions between two states and from one state to a lower-level state machine are marked by the arrow with a label of an outcome describing the transition. Dotted terminal states represent the transition that is called after returning to a higher level state machine. The land event is called whenever any state produces an outcome that means that the UAV cannot continue in its mission.

The diagram of the main state machine is visualized in Fig. 8(a). In the first step, the trajectories for static sweeping and also for dynamic sweeping in the predefined part of the competition arena (see section. 3.7.1) are loaded, and an automatic take off is called. Once the UAV is in the air, the mission state machine is activated (see Fig. 8(b)). The mission is a concurrent state machine that sequentially runs the static sweeping procedure and the dynamic sweeping procedure, while simultaneously controlling the voltage of the battery. If the battery is discharged, the state machine terminates all currently executed tasks of the UAV, and a land event is called. The level of voltage for battery discharge was set experimentally for each

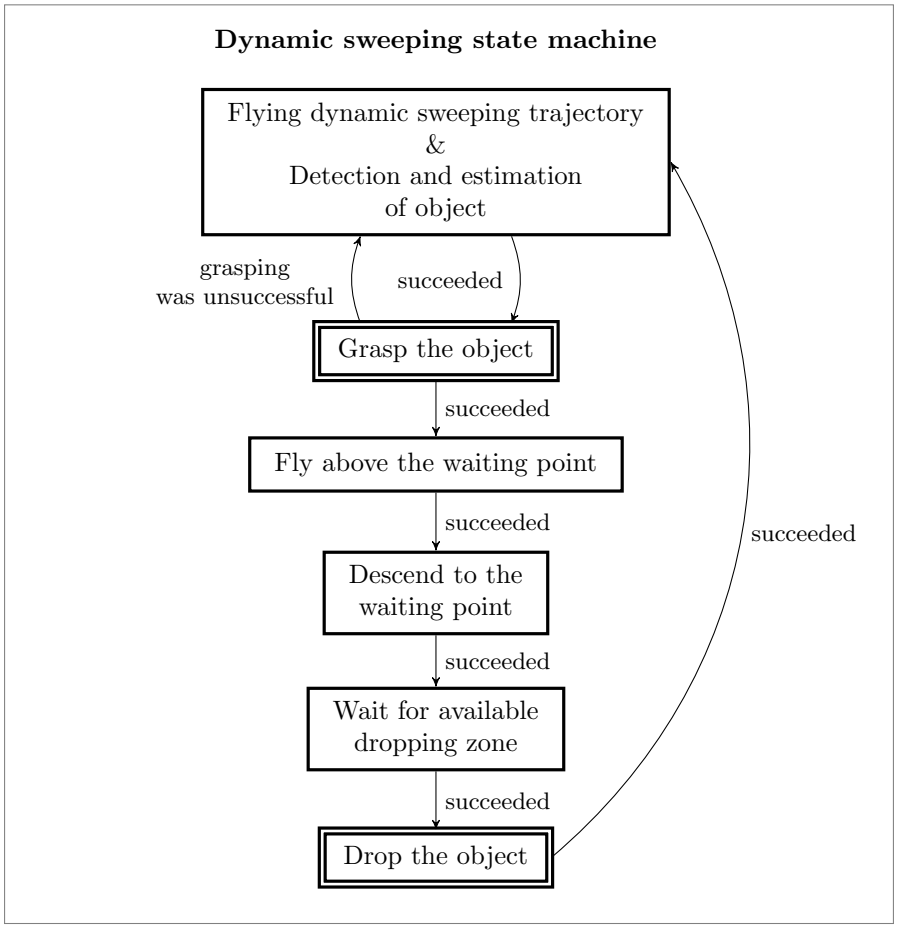
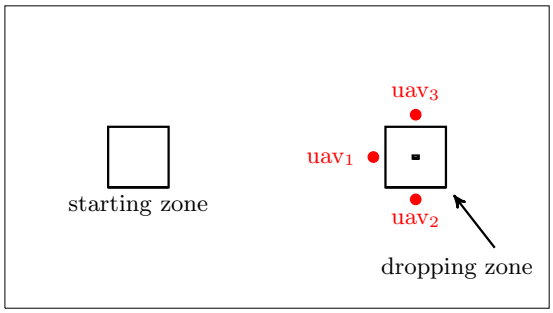
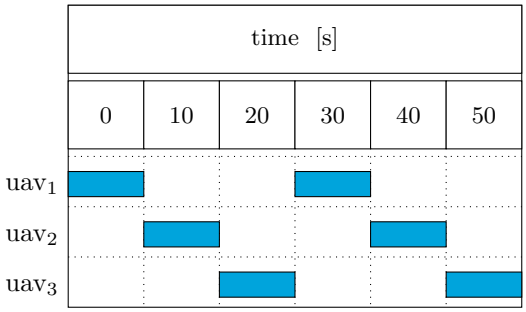


Figure 10: Diagram of the dynamic sweeping state machine.



(a) Position of the waiting point in the MBZIRC arena.



(b) Gantt diagram of the time window for particular UAVs.

Figure 11: Waiting position around the dropping zone, and a Gantt diagram of the proposed time window strategy. The index of the UAV indicates to which part of the arena the UAV belongs.

battery type.

The static sweeping state machine (Fig. 9) starts by following the sweeping trajectory and creating a map with approximate positions of the static objects. After this initial coverage of the competition arena, an attempt is made to grasp the nearest estimated static object in the map. The grasping procedure is shown in Fig. 12. Initially, the state machine starts with the object detection mechanism. Whenever an object is

located, the UAV tries to align itself horizontally above the estimated position of the object and then to descend to the grasping height of 1.5 m above the ground. Once the UAV has reached the desired height and it is aligned above the object, it tries to grasp the object. Whenever the object is lost in the steps after descending to the grasping height, the UAV ascends and repeats these steps again. The steps are also repeated if the grasping fails. Only two attempts are made to grasp the estimated object. If the UAV was not successful in these attempts, the state machine returns the UAV to the safe flying height and it is terminated with the outcome that the grasping was unsuccessful. After a successful grasp, the UAV also ascends to the safe flying height, but the grasping state machine outputs that the grasp was successful. The decision as to whether the UAV is carrying an object is made via a feedback from the Hall effect sensors that are placed on the gripper. To avoid deadlock, the state machine is terminated in the first node if the object is not found within a certain time.

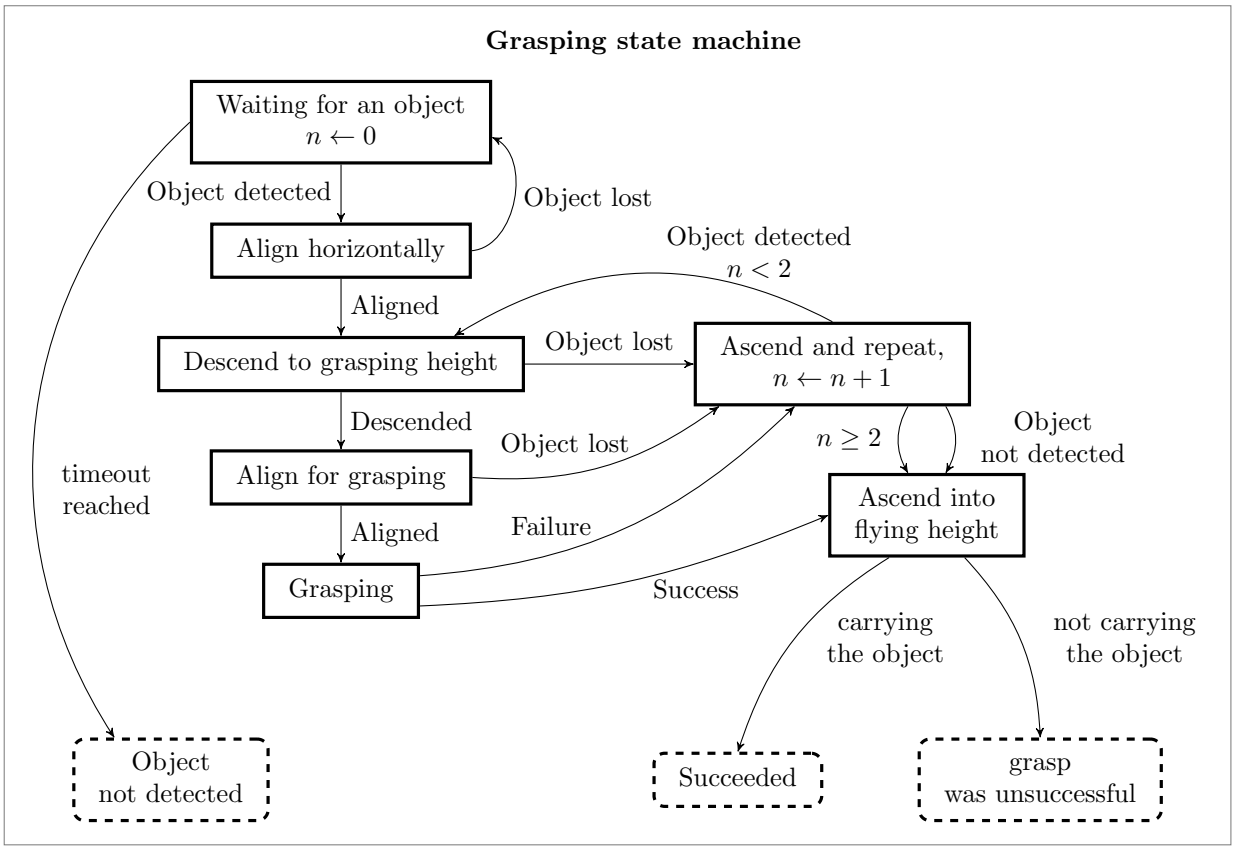


Figure 12: Diagram of the grasping state machine.

The static sweeping state machine reacts to unsuccessful outcomes from the grasping procedure by selecting a new object for grasping from the map. When the grasping attempt was successful and the UAV is carrying the object, the UAV flies at its safe height to a position above the waiting point. The waiting point is selected based on the part of the competition arena assigned to the UAV. During the MBZIRC competition, the safe flying heights for part 1, part 2, and part 3 were 3 m, 4 m, and 5 m, respectively. The waiting points were located 7 m (measured in x, y plain) from the center of the dropping box (see Fig. 11(a)). Once the UAV reaches the position above the waiting point at its safe height, it descends to the dropping height of 1.5 m above the ground. At this waiting point, the UAV hovers until the moment when the dropping zone is not occupied by any other UAV, if the communication infrastructure is available, or until the UAV has access to the dropping zone based on the time windows, if the communication channel cannot be used for negotiation and for sharing the status of the dropping zone.

The negotiation about access to the dropping zone is based on queries of the current UAV on its waiting position addressed to neighboring UAVs. The neighboring UAV responds with confirmation that allows the current UAV to access the zone, but only when the neighboring UAV is not inside this zone, or if the neighboring UAV has not been waiting for access for longer than the current UAV. The current UAV starts with the dropping maneuver only when it receives confirmations from all neighboring UAVs. The negotiation about access to the dropping zone is carried out repeatedly until the UAV receives confirmations.

If communication has been lost for more than a predefined time during the mission, all UAVs will switch to a strategy with time windows for accessing the zone in order to avoid collisions in the dropping zone. Time windows 10 s in length are used for each UAV. This range of time windows provides two time intervals for dropping for each UAV per minute. The UAV in part 1 can be in the restricted area around the dropping zone in the $0-9$ s time interval, the UAV in part 2 can be there in the $10-19$ s time interval, and the UAV in the part 3 can be there in the $20-29$ s time interval. This strategy is the same for accessing the dropping area in the second half of the minute, so the intervals are offset by 30 s (see Fig. 11(b)). The UAV can call the dropping procedure only when it is in the waiting position at the dropping height, and its time window starts. This strategy is not as effective as negotiation and sharing of the status of the dropping zone, but it is safer in the case of a problematic communication network. This strategy requires the clocks on the UAVs to be initially synchronized within a few milliseconds using chrony - an implementation of the Network Time Protocol (NTP).

The dropping maneuver is done in sequence: flying above the dropping box at the dropping height, dropping the object, and returning to the UAV safe height above the waiting position. After dropping the object, the state machine initializes the grasping procedure with the next estimated object in the map. This is done until all detected objects have been grasped, or an attempt has been made to grasp them, in the case of a grasping failure.

In the dynamic sweeping state machine (Fig. 10), the UAV flies the dynamic sweeping trajectory, and when any object is detected and its position is estimated, the UAV immediately tries to grasp it. After successful grasping and dropping, the UAV flies back into the dynamic sweeping trajectory and continues with dynamic sweeping while simultaneously looking for the remaining objects. This approach is not as effective as the initial static sweeping procedure, where the UAVs could fly for another object in the map directly, and minimize the overall flight time, but it is more robust. In the ideal case of perfect mapping and grasping procedures, all static objects are grasped during the static sweeping part, and only the dynamic objects are hunted during the dynamic sweeping. In the demanding real world conditions of the MBZIRC arena, with changing light conditions and wind gusts, many objects were not grasped in the first phase of the mission. This was due to a safety procedure that allowed a limited number of grasping attempts per object to avoid a deadlock. These missed objects could be grasped later, in the dynamic sweeping part, as the local environment conditions changed.

Another interesting property of this approach is the possibility to exchange the sweeping trajectories, and therefore the operational zones and waiting positions between the UAVs after a given period of the mission. This increases the robustness and the performance of the overall system in the event of a failure or a malfunction of a UAV subsystem. Even if all components of all UAVs are fully functional as designed, each UAV in the team behaves differently in different tasks, and it often happened that a UAV could accomplish a task in which another team member failed, and vice versa. This is another useful lesson learned during the MBZIRC event that should be adapted for designing multi-robot systems, if possible. Finally, splitting the static object grasping in the initial sweeping part and the subsequent grasping of dynamic objects and the remaining static objects increases the overall system robustness. There is a much lower probability of a UAV crash during static object grasping. This has been shown in numerous realistic complex simulations, and also during system testing and its deployment in the competition.

4 Experimental results

In this section, we present both the experimental results achieved while preparing for the Treasure Hunt scenario, and also the performance of the system during the MBZIRC competition. The remainder of this section is divided into main parts, where we present the experimental results achieved in the simulator, during the preparations for the competition in South Bohemia, in the final tests in a challenging desert environment and in the course of the MBZIRC competition. A video attachment to this paper is available at website `http://mrs.felk.cvut.cz/jfr2018treasurehunt`.

4.1 Robotic simulator

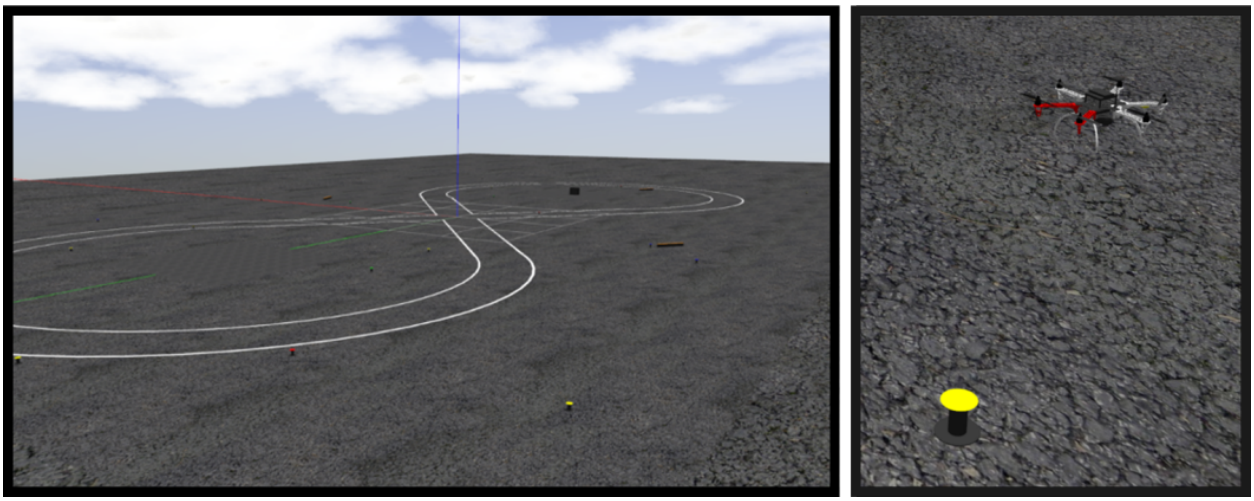


Figure 13: Snapshots from the simulation developed for the MBZIRC competition.

The system was initially developed using the Gazebo robotic simulator, which was employed as the simulation in the loop, together with the PixHawk firmware. Using the robotic simulator, the process of developing the sub-systems and integrating the entire system was carried out significantly faster and more safely than when using the real system directly. In addition, by modeling the whole scenario in the simulator and by testing the behavior of the complex FSM approach in it, the complete system achieved the necessary level of reliability for deployment in tasks such as the Treasure Hunt.

The underlying layers of the control pipeline, namely the UAV state estimation, control, tracking and predictive collision avoidance were extensively tested using the Gazebo simulator. To show the system robustness, we conducted 24-hour simulated flights of 5 UAVs in an area of 100×100 m. Each UAV followed an independent random walk reference in the same height. Without the collision avoidance technique, the median time of the first collision between any of the UAVs was 104 s, from total of 495 simulated scenarios (simulation was always restarted after the first collision). With the collision avoidance mechanism, there was not a single collision withing the 24 hours of the experiment, while the minimal registered distance between the UAVs was 1.21 m, which is still 50% more than the collision distance 0.8 m of the employed platforms. See Tab. 1 for the comparison of percentiles of duration of the experiment before the first collision occurred.

The results from 20 simulations of the complete MBZIRC 2017 Treasure Hunt scenario are shown in Tab. 2. Each of these simulations contained 10 static and 10 dynamic objects, which were randomly placed in a simulated MBZIRC arena. Snapshots from simulation are shown in Fig. 13. We expected that the dynamic objects will move according to some motion model that is predictable and smooth. Therefore, we modeled the movement of the dynamic objects in simulation using the car-like motion model, where the velocity of the object did not exceed 0.3 m/s. Due to the movement type of dynamic objects being uncertain, the mission was divided into two part. The first part is the safe part of the mission, where only the static objects

percentile	0.5	0.75	0.95	0.99
without the avoidance	104 s	152 s	264 s	431 s
with the avoidance	–	–	–	–

Table 1: Percentiles of duration of the experiment before the first collision occurred. The results were obtained in two 24 h simulated flights (one with and one without the collision avoidance mechanism employed) with 5 UAVs, conducting a 2D random walk on 100×100 m area. The total of 495 collisions were recorded if the collision avoidance mechanism was not used.

	Mission time [min]	Time needed for grasping of the static object [s]	Time needed for grasping of the dynamic object [s]	Smallest distance between UAVs [m]
min	12.1	23.7	35.0	1.9
max	17.4	36.4	51.2	3.3
mean	13.6	30.6	43.6	2.5

Table 2: Results from 20 simulations of Challenge 3, in which the objects (10 static and 10 moving) were randomly placed. UAVs in a distance closer than 0.8 m are colliding in the simulation as well as in the real system, which never happened in simulations and real flights if the collision avoidance approach was employed.

are attempted to be grasped and delivered. After this part is finished, the rest of objects will be targeted regardless of whether they are static or dynamic. Results from the simulations in Tab. 2 show, that the system is capable of collecting all targets to the dropping box in the competition time interval of 20 minutes. The best time of finishing the mission was 12.1 min and the worst was 17.4 min. The average time needed was 13.6 min. Results in Tab. 2 also show that all static objects were grasped faster than the fastest dynamic object. Furthermore, thanks to using collision avoidance methods, there was no collision between members of the team during the mission. The closest any UAVs got to each other was 1.9 m, which only happened in one of the simulations, and in general the mutual distances were higher than that.

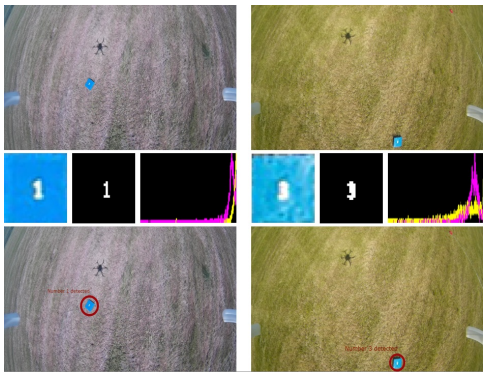
4.2 Experimental camps in the Czech Republic

Key parts of the proposed system were tested in the course of experimental camps held in the countryside of South Bohemia in the Czech Republic throughout the year before the competition (see Fig. 14). Repeated experimental verification of key parts of the proposed system was necessary in order to test phenomena that are difficult to simulate, and also to discover issues that were not present in our previous hardware experiments without physical interaction of the robot with the real-world environment. One issue that was discovered was the influence of the force produced by the propellers on the carried objects. This exposed the need for a stronger magnetic gripper, which we then designed. Another discovered issue was the ground effect caused by the objects. This manifested itself as turbulence in the last phase of the grasping maneuver.

The most crucial parts of the system were the low-level UAV control and the MPC-based trajectory tracking, used for precise positioning of the UAV. These were thoroughly tested to obtain the centimeter precision required for the grasping task. The MPC-based trajectory tracking used during the colored object mapping is shown in Fig. 14(a). In addition, initial testing of the object detection was carried out. However, in accordance with the initial specification of the shape of the object expected in the competition, we designed square shaped objects with numbers describing the rewards (see Fig. 14(b)). Videos showing initial attempts for grasping and dropping of the object, and trajectory following are available at `http://mrs.felk.cvut.cz/jfr2018treasurehunt-video1`.

One of the experimentally verified sub-systems was the MPC-based collision avoidance implemented for reactive avoidance of collisions between multiple UAVs in the MBZIRC competition. Using the MPC pre-

(a) MPC-based trajectory tracking with low-level UAV control during the mapping of the colored objects spread throughout the experimental field.



(b) Object detection and number (reward) recognition of square-shaped colored objects.

Figure 14: Experimental verification of the MPC-based trajectory tracking method and the object detection algorithm during the experimental camps in the countryside of South Bohemia in the Czech Republic. `http://mrs.felk.cvut.cz/jfr2018treasurehunt-video1`

dictions of the future parts of the trajectory of other UAVs (discussed in section 3.6), each UAV can avoid collisions with other UAVs by a simple change of flight height in potential collision parts of the future trajectory. It is necessary to employ this method in scenarios with a problematic communication network. This is because after reestablishing communication the proactive collision-free planning may not be able to deal with a suddenly discovered imminent collision, or may not even be active in the current phase. This safety mechanism is implemented on the lowest level of control in all phases of the mission. Fig. 15 shows the verification of MPC-based collision avoidance, with two UAVs exchanging their position and one hovering UAV between the two positions. A video showing this verification is available at `http://mrs.felk.cvut.cz/jfr2018treasurehunt-video2`. Such collision avoidance requires only a small number of messages to be shared between UAVs. These messages contain the MPC future trajectory predictions of each UAV, and are distributed with a very low frequency of 2 Hz. Although the proposed collision avoidance technique requires only a low communication bandwidth ($\sim$ 6 kB/s for three UAVs), the collision avoidance was not always used during the competition, due to drop outs of communication between UAVs, which was observed by all teams in the competition.

Another evaluated subsystem was the object detection and mapping. In particular, the datasets gathered were used to compare computational efficiency of our object detection method to the MSER (Matas et al., 2004) and 'SimpleBlobDetector' methods included in the OpenCV library (Bradski, 2000). The results indicated that the system presented achieved significantly higher framerates compared to the aforementioned two methods. This confirmed the experiments in (Krajník et al., 2014), which introduced an algorithm our detection was based on.

4.3 Desert testing in the United Arab Emirates

Finally, the complete system was thoroughly tested for a period of three weeks just before the competition, in the desert near Abu Dhabi in the United Arab Emirates. The desert environment was challenging, due to the uneven terrain and the rapidly changing wind conditions. By tuning the system for such weather and terrain conditions, our system was better prepared for the environment at the Yas Marina Circuit in Abu Dhabi, where the competition was held. The rapidly changing terrain profile in the dunes of the desert also had an influence on the quality of the communication network. The frequent interruptions of the connection inspired our solution, which does not rely on the communication network.

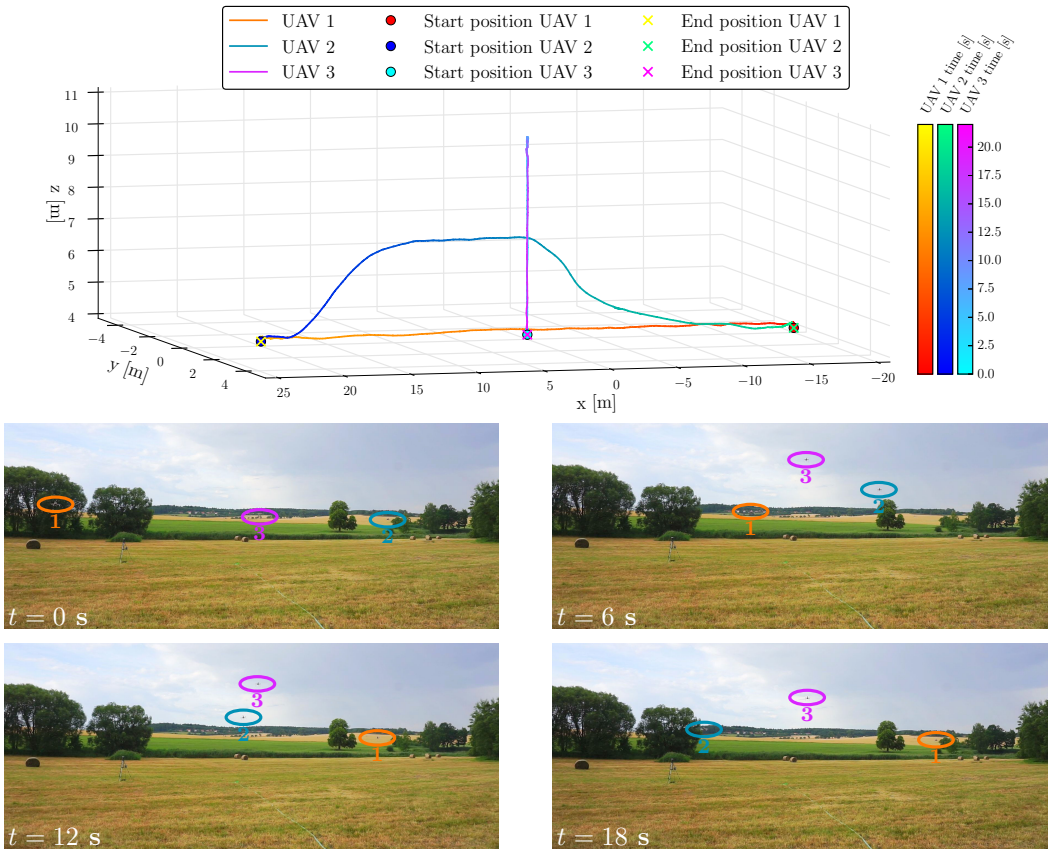


Figure 15: MPC-based collision avoidance between three drones. Two drones (UAV 1 and UAV 2) exchange their positions, while the third UAV 3 hovers in a position colliding with their trajectories. Using MPC future trajectory prediction, the UAVs avoid a collision by changing their trajectory height. `http://mrs.felk.cvut.cz/jfr2018treasurehunt-video2`

As we have mentioned, several important features of our system were, in our opinion, the dominant factors that led to our winning performance in all trials of the Treasure Hunt challenge in the MBZIRC competition. Most of the other teams did not take into consideration external disturbances such as wind in their controller. Surprisingly, the MBZIRC competition arena was not perfectly flat, and some teams had relied on its flatness. Finally, relying on a robust communication network was the main bottleneck of the competitive solutions.

Photos from the tests of the system in the desert are shown in Fig. 16. The grasping procedure is captured in the image on the right, and the dropping maneuver is shown in the image on the left. A video showing the behavior of the complete system with three UAVs in this environment is available at `http://mrs.felk.cvut.cz/jfr2018treasurehunt-video3`. During this testing, the yellow objects were stationary as opposed to the competition, where they were dynamic. This means, that in this phase, the system was tested for the static objects only. In addition, the paths traveled by the UAVs during the experiment presented in the video are shown in Fig. 17. In this figure, the z-axis denotes the height above the level of the starting position as measured by the differential RTK GPS. The UAVs were kept at constant height above the ground and therefore the graph shows how uneven the terrain was. Furthermore, the Fig. 17 depicts the positions and colors of the objects that were collected.

Figure 16: Photos from the tests of the proposed system in the desert near Abu Dhabi, United Arab Emirates. The grasping procedure is captured in the image on the right, and the dropping maneuver is shown in the image on the left. `http://mrs.felk.cvut.cz/jfr2018treasurehunt-video3`

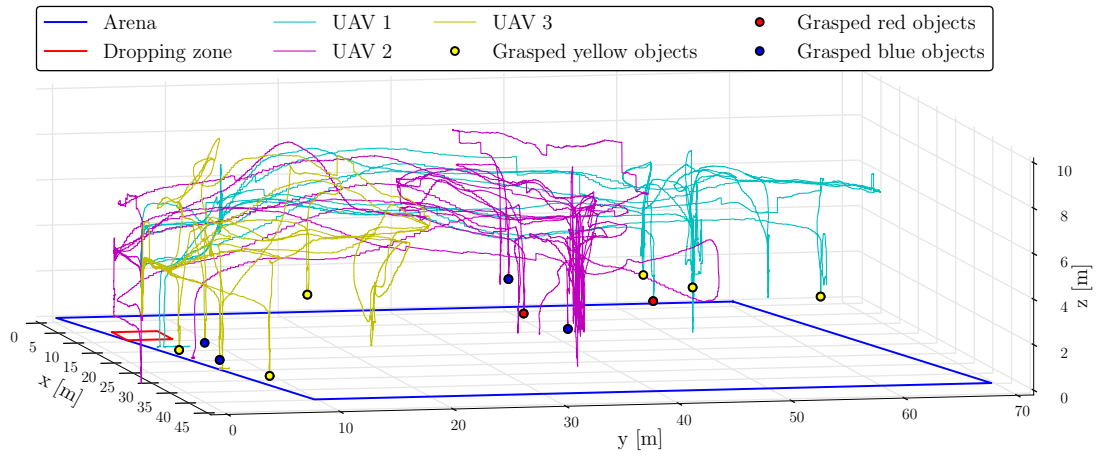


Figure 17: The paths traveled by individual UAVs during the desert experiment. The colored points denotes the positions of the objects that were collected.

4.4 Results from the MBZIRC competition

Our system was applied four times in the Treasure Hunt scenario during the final MBZIRC competition. During the competition, the number of dynamic (yellow) objects was decreased from announced 10 to 3 for this scenario for organizational reasons. The results, i.e. the number of colored objects that were collected, are shown in Tab. 3. The first two attempts, denoted as TRIAL 1 and TRIAL 2, are the results from Challenge 3, which contained only the Treasure Hunt scenario. The remaining two trials (GRAND 1 and GRAND 2) were a part of the Grand Challenge, where the Treasure Hunt scenario was undertaken simultaneously with the scenario of landing on a moving ground vehicle (Challenge 1), and the scenario where a ground robot had to locate and reach a panel, and further physically operate a valve located on the panel (Challenge 2). During these four trials within the competition, twenty-five objects overall were successfully placed into the dropping zone. The best performance according to the number of grasped and placed objects was achieved during the second trial of the Grand Challenge, when eight objects, including a non-stationary object were brought into the dropping zone. The system described in this paper won first

place in Challenge 3, and contributed to our third place in the Grand Challenge. A video showing results from the MBZIRC competition is available at `http://mrs.felk.cvut.cz/jfr2018treasurehunt-video4`.

	placed into the box	placed outside the box but inside the dropping area
TRIAL 1	2R, 2G	1G
TRIAL 2	2R, 3G	
GRAND 1	1R, 1G, 2B, 1Y	1G, 1B
GRAND 2	2R, 3G, 1B, 1Y	1G

Table 3: Numbers of the objects collected in the Treasure hunt scenario during the MBZIRC 2017 competition. TRIAL 1&2 - trials of MBZIRC Challenge 3, GRAND 1&2 - trials of the MBZIRC Grand Challenge, R - red static object, G - green static object, B - blue static object, Y - yellow non-stationary object.

One part of the system for the Treasure Hunt scenario involved localizing objects using sweeping trajectories (described in section 3.7.1). The static sweeping paths traveled by UAVs in the trials of Challenge 3 are shown in Fig. 18 and Fig. 19. The flight time of the described UAV platform with fully charged 4 cell batteries with 6750 mAh capacity is up to 15 minutes, which is less than allowed time per trial. The organizers allowed to change the batteries during the trial without any penalization. The trajectories before changing the batteries are labeled in the graphs as part 1, and after the batteries are changed, they are labeled as part 2. Furthermore, on these graphs, the colored points denote the detections of the objects that were observed, and the larger circles denote the estimated positions of these objects. After processing the data from the first trial, we decided to decrease the sweeping trajectory height from 7 m to 6.5 m. This modification made objects more visible in camera images, which improved object detection. A disadvantage of this change was that it prolonged the trajectories, because the condition of at least 20% of overlap in the coverage could not be satisfied by following the same trajectory (in the xy plane). For this reason, the sweeping trajectories differ between these two trials.

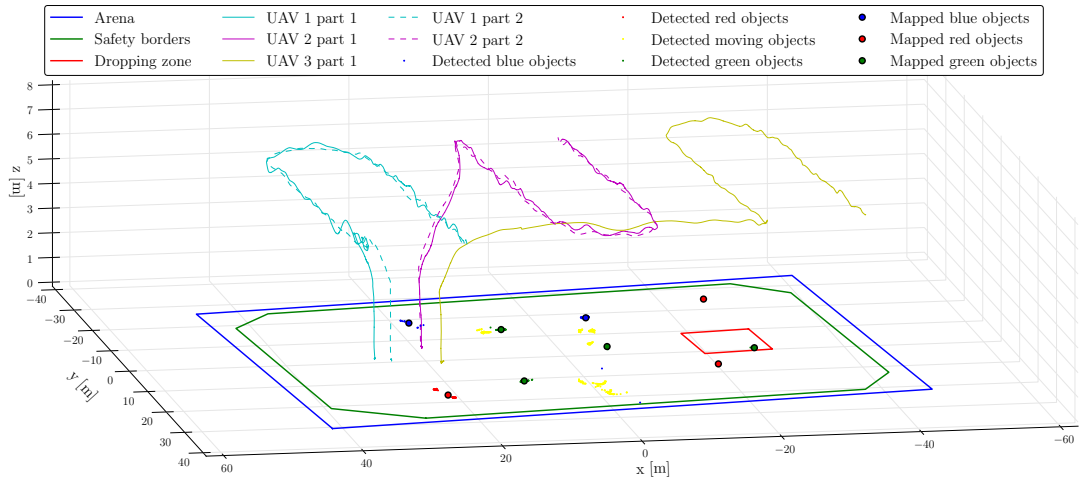


Figure 18: Mapping sweep during the first trial of Challenge 3. The colored points denote the detections of the objects that were observed, and the larger circles denote the estimated positions of these objects.

Another important part of described system is the grasping capability, where the UAV has to grasp a ferrous object. The overall grasping approach has been presented in section 3.8, where the grasping state machine is depicted in Fig. 12. Switching of the phases of the grasping state machine is shown in Fig. 20, where an attempt at grasping was repeated after being aborted once. For a visualization of the transition between these phases, the resolution of the graph in Fig. 20(a) is 0.05 m in the x-axis and in the y-axis. In addition, detections of the object in three parts, which are indicated by dotted arrows, are shown in Fig. 20(b)-(d).

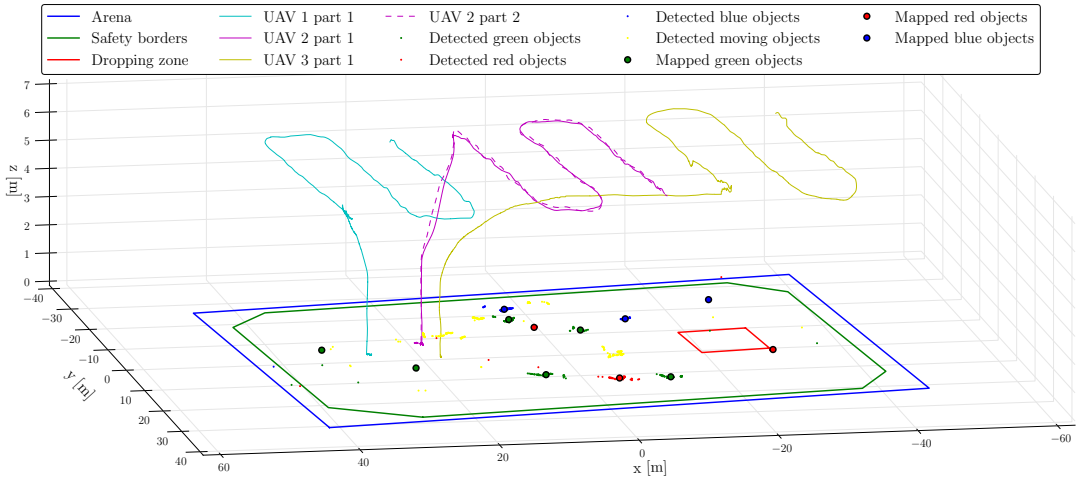
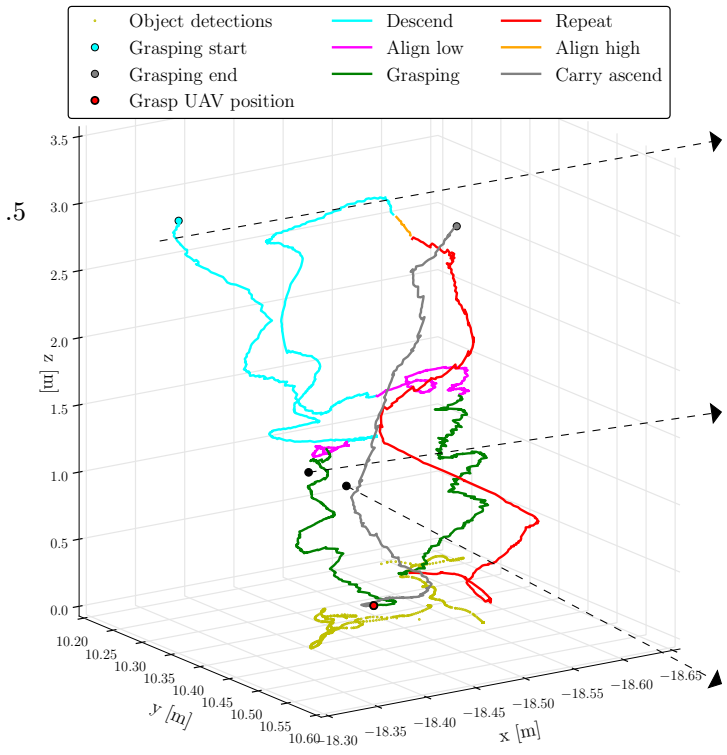
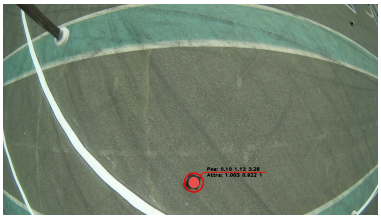


Figure 19: Mapping sweep during the second trial of Challenge 3. The colored points denote the detections of the objects that were observed, and the larger circles denote the estimated positions of these objects.



(a) UAV position with grasping states and object detections.



(b) Start of the grasping state machine.

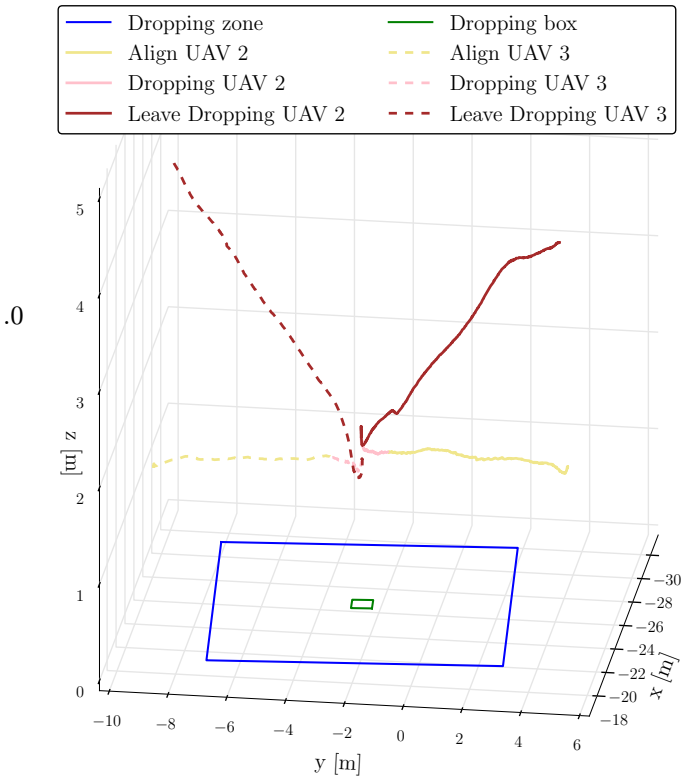
(c) Before the final grasping attempt.

(d) Carrying a grasped object.

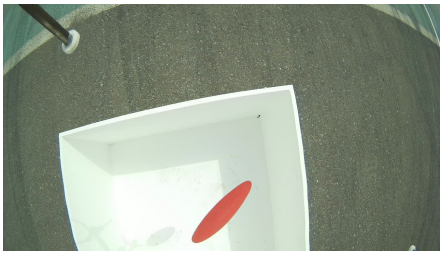
Figure 20: Phases of the grasping state machine, UAV position estimation and object detection during grasping of a red static object, with a successful second grasping attempt.

The dropping approach for delivering the grasped objects into the dropping box has been described in section 3.8. Switching the phases of the dropping state machine is shown in Fig. 21(a), where the dropping procedure was carried out by two UAVs. Objects were dropped by each UAV at a different time, and thus there was no collision between them. Fig. 21(c) and Fig. 21(b) show snapshots from the onboard cameras

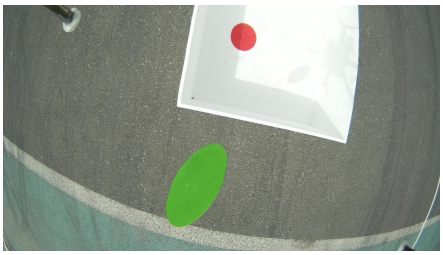
on the UAVs during the dropping maneuver.



(a) UAV positions with dropping states.



(b) UAV 3 onboard camera image after dropping red object



(c) UAV 2 onboard camera image after dropping green object

Figure 21: Phases of the dropping state machine of two UAVs.

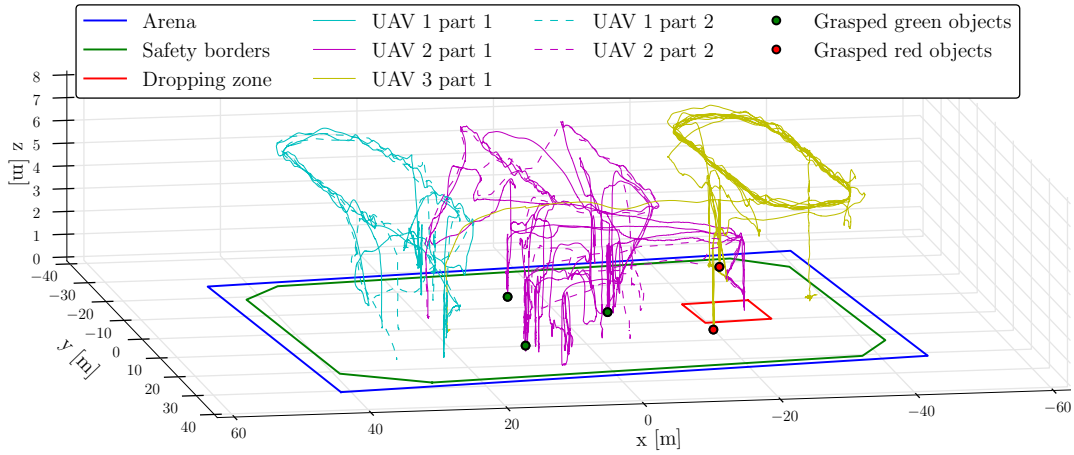


Figure 22: Paths traveled by individual UAVs during the first trial of Challenge 3. The colored points denotes the positions of the objects that were collected.

Photos from the competition are shown in Fig. 24. The upper image shows the UAV following the static sweeping trajectory. The images in the middle row and the image on the left in the lower part of the figure capture moments when the UAVs were grasping objects. The remaining image shows an object being dropped into the dropping box. In addition, the paths traveled by the UAVs during the first trial of Challenge 3 are shown in Fig. 22, and the paths traveled in the second trial of the same challenge are shown in Fig. 23.

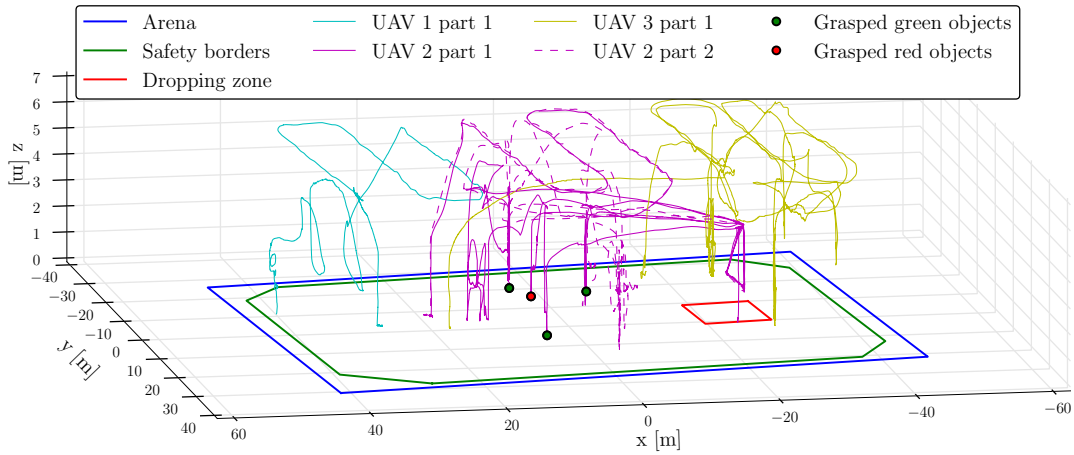


Figure 23: Paths traveled by individual UAVs during the second trial of Challenge 3. The colored points denotes the positions of the objects that were collected.

Furthermore, in these graphs, the colored points denotes the positions of the objects that were collected.

5 Lessons learned

Although the competition results can be considered a major success, it was not without hurdles, mainly during implementation, testing and tuning of the proposed system. From the implementation part, it was convenient to develop the system compatible with the ROS, which allows to divide the system into independent components that were implemented separately by different research groups. Furthermore, their testing were significantly simplified by employing the Gazebo robotic simulator together with the firmware from PixHawk, which speeds up the overall progress of the development.

The required usage of more vehicles simultaneously even increased the complexity of the task. Every UAV is equipped with several sensors that could be a source of unreliability. By testing the behavior of the proposed system in desert near Abu Dhabi in the United Arab Emirates, our system was well prepared for the environment at the Yas Marina Circuit in Abu Dhabi, where the competition was held. The system was tuned to properly react to strong wind, decreased visibility due to sand, and to problems occurring by intensive light from sun. Hence we stress the significance of the real-world outdoor experiments above simulation, to obtain real sensor data.

5.1 Toward a more general solution

Despite our best effort to develop a general solution capable of autonomous searching, picking, and placing objects, several sub-systems have been tailored specifically to the competition scenario. The vision system was designed to locate the objects with colors and shapes specified by the rules of the competition. In the case of an object of more difficult shape and color patterns, a different approach for its localization would be required, e.g., based on artificial neural networks. Further, estimation and prediction of the object movement using a car-like model provide a framework suitable for tracking the most common ground vehicles. A more precise model could be used to better estimate state of a specific vehicle (e.g. with differential drive model, or if capable of 3D motion). The presented proactive collision-free planning, using different flying heights and the visibility graph method, has been selected due to simple scenario with three UAVs. A requirement of a higher number of independent flying heights would occur with higher number of deployed UAVs. Then a different splitting of the arena would be required since it is not efficient to often ascend and descend for the

Figure 24: Photos from the MBZIRC competition. The upper image shows the UAV while following the static sweeping trajectory. The images in the middle row and the left on the lower part of the figure capture the moments when the UAVs were grasping objects. The remaining image shows an object being dropped into the dropping box. During four trials within the competition (two for Challenge 3 and two for the Grand Challenge), twenty-five objects overall were placed into the dropping zone (see Tab. 3). `http://mrs.felk.cvut.cz/jfr2018treasurehunt-video4`

UAV. In this case, each individual UAV will be re-solving a possible collision only with other UAVs, that are assigned to arena parts through which the UAV will need to fly. Taking these observations, the presented approach can be applied to various outdoor multi-robot scenarios, as shown in our consequent research after the competition listed in section 1.2.

6 Conclusions

A system designed for Challenge 3 of the MBZIRC competition has been described in this paper. The paper has focused on the properties of the design that in our opinion were the most important factors leading to the best performance of the system in all trials in the Treasure Hunt challenge. The system is able to solve object manipulation tasks in demanding outdoor environments, and to do so cooperatively in a team of three UAVs.

While many of the methods described here do not represent the bleeding edge of robotics research, they were designed to be versatile and substitutable. This allowed their easy integration into a complex modular system, which enabled efficient testing of the individual modules, making us aware of these modules deficiencies and possible faults during their deployment in real conditions. Our knowledge of the faults encountered during the field tests was reflected in the design of the core module of our system, the Failure Recovery and Synchronization jobs Manager (FSM). This module ensured that occasional faulty behaviour of the individual modules did not result in a critical failure or system deadlock. Still, the development of this complex system led to numerous significant contributions beyond the state-of-the-art in robotics, which could facilitate the deployment of multi-UAV platforms in challenging scenarios motivated by current needs of the industry. This was the main motivation for our paper and also for the MBZIRC competition itself.

The results shown in numerous realistic simulations in Gazebo and in experiments in a demanding desert environment have been presented in this paper following by analysis of necessary improvements of the system towards more general applications, which go beyond the MBZIRC 2017 competition. However, the most meaningful and credible verification of the performance and the reliability of the system was achieved in the MBZIRC competition, where our approach won the first place Challenge 3, on the basis of achieving the best score among 17 finalists selected from 142 registered teams.

Acknowledgments

The outstanding results of this project could not have been achieved without the full cooperation of each member of our team, comprising people from the Czech Technical University in Prague, the University of Pennsylvania and the University of Lincoln, UK (Fig. 25). The work has been supported by CTU grant no. SGS17/187/OHK3/3T/13, Research Center for Informatics project CZ.02.1.01/0.0/0.0/16_019/0000765, the Grant Agency of the Czech Republic under grant no. 17-16900Y, ARL grant W911NF-08-2-0004, ONR grants N00014-07-1-0829, N00014-14-1-0510, and by Khalifa University.

References

Afolabi, D., Man, K. L., Liang, H.-N., Guan, S.-U., and Krilavičius, T. (2015). 1543. monocular line tracking for the reduction of vibration induced during image acquisition. *Journal of Vibroengineering*, 17(2).

Baca, T., Loianno, G., and Saska, M. (2016). Embedded model predictive control of unmanned micro aerial vehicles. In *21st International Conference on Methods and Models in Automation and Robotics (MMAR)*, pages 992–997.

Bähnemann, R., Schindler, D., Kamel, M., Siegwart, R., and Nieto, J. (2017). A decentralized multi-agent unmanned aerial system to search, pick up, and relocate objects. In *Proceedings of 2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, page 8088150.

Figure 25: Team members that were involved in the MBZIRC competition in Abu Dhabi, United Arab Emirates.

Bohren, J. (2017). SMACH a task-level python execution framework for rapidly composing complex robot behaviors, Robot Operating System (ROS). Available: http://wiki.ros.org/smach (cited on 2018-07-17).

Bradski, G. (2000). The OpenCV library. *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, 25(11):120–123.

Choset, H. and Pignon, P. (1998). *Coverage Path Planning: The Boustrophedon Cellular Decomposition*, pages 203–209. Springer London, London.

Digi International, I. (2017). XBee-PRO RF Modules. Available: https://www.sparkfun.com/datasheets/Wireless/Zigbee/XBee-Datasheet.pdf (cited on 2018-07-17).

Dubins, L. E. (1957). On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of mathematics*, 79(3):497–516.

Fumagalli, M., Stramigioli, S., and Carloni, R. (2016). Mechatronic design of a robotic manipulator for unmanned aerial vehicles. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4843–4848.

Galceran, E. and Carreras, M. (2013). A survey on coverage path planning for robotics. *Robotics and Autonomous Systems*, 61(12):1258 – 1276.

Ghadiok, V., Goldin, J., and Ren, W. (2012). On the design and development of attitude stabilization, vision-based navigation, and aerial gripping for a low-cost quadrotor. *Autonomous Robots*, 33(1):41–68.

Gioioso, G., Franchi, A., Salvietti, G., Scheggi, S., and Prattichizzo, D. (2014). The flying hand: A formation of uavs for cooperative aerial tele-manipulation. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4335–4341.

Heredia, G., Jimenez-Cano, A. E., Sanchez, I., Llorente, D., Vega, V., Braga, J., Acosta, J. A., and Ollero, A. (2014). Control of a multirotor outdoor aerial manipulator. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3417–3422.

Herissé, B., Hamel, T., Mahony, R., and Russotto, F. X. (2012). Landing a vtol unmanned aerial vehicle on a moving platform using optical flow. *IEEE Transactions on Robotics*, 28(1):77–89.

Kamel, M., Comari, S., and Siegwart, R. (2016). Full-body multi-objective controller for aerial manipulation. In *2016 24th Mediterranean Conference on Control and Automation (MED)*, pages 659–664.

Kannan, S., Quintanar-Guzman, S., Dentler, J., Olivares-Mendez, M. A., and Voos, H. (2016). Control of aerial manipulation vehicle in operational space. In *2016 8th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, pages 1–4.

Kessens, C. C., Thomas, J., Desai, J. P., and Kumar, V. (2016). Versatile aerial grasping using self-sealing suction. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3249–3254.

Kim, S., Seo, H., Choi, S., and Kim, H. J. (2016). Vision-guided aerial manipulation using a multirotor with a robotic arm. *IEEE/ASME Transactions on Mechatronics*, 21(4):1912–1923.

Korpela, C. M., Danko, T. W., and Oh, P. Y. (2012). Mm-uav: Mobile manipulating unmanned aerial vehicle. *Journal of Intelligent & Robotic Systems*, 65(1):93–101.

Krajník, T., Nitsche, M., Faigl, J., Vaněk, P., Saska, M., Přeučil, L., Duckett, T., and Mejail, M. (2014). A practical multirobot localization system. *Journal of Intelligent & Robotic Systems*, 76(3-4):539–562.

Lee, T., Leok, M., and McClamroch, N. H. (2013). Nonlinear robust tracking control of a quadrotor uav on se(3). *Asian Journal of Control*, 15(2):391–408.

Lippiello, V., Cacace, J., Santamaria-Navarro, A., Andrade-Cetto, J., Trujillo, M. ., Esteves, Y. R., and Viguria, A. (2016). Hybrid visual servoing with hierarchical task composition for aerial manipulation. *IEEE Robotics and Automation Letters*, 1(1):259–266.

Lozano-Pérez, T. and Wesley, M. A. (1979). An algorithm for planning collision-free paths among polyhedral obstacles. *Commun. ACM*, 22(10):560–570.

Matas, J., Chum, O., Urban, M., and Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 22(10):761–767.

Meier, L., Tanskanen, P., Heng, L., Lee, G. H., Fraundorfer, F., and Pollefeys, M. (2012). Pixhawk: A micro aerial vehicle design for autonomous flight using onboard computer vision. *Autonomous Robots*, 33(1):21–39.

Mellinger, D. and Kumar, V. (2011). Minimum snap trajectory generation and control for quadrotors. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2520–2525.

Mellinger, D., Lindsey, Q., Shomin, M., and Kumar, V. (2011). Design, modeling, estimation and control for aerial grasping and manipulation. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2668–2673.

Mellinger, D., Shomin, M., Michael, N., and Kumar, V. (2013). *Cooperative Grasping and Transport Using Multiple Quadrotors*, pages 545–558. Springer Berlin Heidelberg, Berlin, Heidelberg.

Morton, K. and Toro, L. F. G. (2016). Development of a robust framework for an outdoor mobile manipulation uav. In *2016 IEEE Aerospace Conference*, pages 1–8.

NicaDrone (2017). OpenGrab EPM v3 electropermanent magnet. Available: http://nicadrone.com (cited on 2018-07-17).

Nieuwenhuisen, M., Beul, M., Rosu, R. A., Quenzel, J., Pavlichenko, D., Houben, S., and Behnke, S. (2017). Collaborative object picking and delivery with a team of micro aerial vehicles at mbzirc. In *2017 European Conference on Mobile Robotics (ECMR)*, pages 1–6.

Orsag, M., Korpela, C., Pekala, M., and Oh, P. (2013). Stability control in aerial manipulation. In *2013 American Control Conference*, pages 5581–5586.

Parra-Vega, V., Sanchez, A., Izaguirre, C., Garcia, O., and Ruiz-Sanchez, F. (2013). Toward aerial grasping and manipulation with multiple uavs. *Journal of Intelligent & Robotic Systems*, 70(1):575–593.

Pěnička, R., Faigl, J., Váňa, P., and Saska, M. (2017a). Dubins orienteering problem. *IEEE Robotics and Automation Letters*, 2(2):1210–1217.

Pěnička, R., Saska, M., Reymann, C., and Lacroix, S. (2017b). Reactive dubins traveling salesman problem for replanning of information gathering by uavs. In *European Conference of Mobile Robotics (ECMR)*, pages 259–264.

Pounds, P. E. I., Bersak, D. R., and Dollar, A. M. (2011a). Grasping from the air: Hovering capture and load stability. In *2011 IEEE International Conference on Robotics and Automation*, pages 2491–2498.

Pounds, P. E. I., Bersak, D. R., and Dollar, A. M. (2011b). Practical aerial grasping of unstructured objects. In *2011 IEEE Conference on Technologies for Practical Robot Applications*, pages 99–104.

Ramon Soria, P., Arrue, B. C., and Ollero, A. (2017). Detection, location and grasping objects using a stereo sensor on uav in outdoor environments. *Sensors*, 17(1):103.

Ramon Soria, P., Bevec, R., Arrue, B. C., Ude, A., and Ollero, A. (2016). Extracting objects for aerial manipulation on uavs using low cost stereo sensors. *Sensors*, 16(5):700.

Santamaria-Navarro, A., Grosch, P., Lippiello, V., Sola, J., and Andrade-Cetto, J. (2017). Uncalibrated visual servo for unmanned aerial manipulation. *IEEE/ASME Transactions on Mechatronics*, PP(99):1–1.

Saska, M. (2017). Large sensors with adaptive shape realised by selfstabilised compact groups of micro aerial vehicles. In *International Symposium on Robotic Research*.

Saska, M., Kratky, V., Spurny, V., and Baca, T. (2017). Documentation of dark areas of large historical buildings by a formation of unmanned aerial vehicles using model predictive control. In *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–8.

Saska, M., Kulich, M., and Preucil, L. (2006). Elliptic net-a path planning algorithm for dynamic environments. In *ICINCO-RA*, pages 372–377.

Spica, R., Franchi, A., Oriolo, G., Blthoff, H. H., and Giordano, P. R. (2012). Aerial grasping of a moving target with a quadrotor uav. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4985–4992.

Tersus-GNSS (2017). PRECIS-BX305 GNSS RTK Board. Available: https://www.tersus-gnss.com (cited on 2018-07-17).

Thomas, J., Loianno, G., Polin, J., Sreenath, K., and Kumar, V. (2014). Toward autonomous avian-inspired grasping for micro aerial vehicles. *Bioinspiration & Biomimetics*, 9(2):025010.

Tiderko, A. (2017). multimaster_fkie - Multi-master ROS Package, Robot Operating System (ROS). Available: http://wiki.ros.org/smach (cited on 2018-07-17).

Weiss, S., Scaramuzza, D., and Siegwart, R. (2011). Monocular-slam–based navigation for autonomous micro helicopters in gps-denied environments. *Journal of Field Robotics*, 28(6):854–874.