A Fully-Autonomous Aerial Robotic Solution for the 2016 International Micro Air Vehicle Competition

Carlos Sampedro[1], Hriday Bavle[1], Alejandro Rodríguez-Ramos[1],
Adrian Carrio[1], Ramon A. Suárez Fernández[1], Jose Luis Sanchez-Lopez[1], and Pascual Campoy[1]

***Abstract*— In this paper, a fully-autonomous quadrotor aerial robot for solving the different missions proposed in the 2016 International Micro Air Vehicle (IMAV) Indoor Competition is presented. The missions proposed in the IMAV 2016 competition involve the execution of high-level missions such as entering and exiting a building, exploring an unknown indoor environment, recognizing and interacting with objects, landing autonomously on a moving platform, etc. For solving the aforementioned missions, a fully-autonomous quadrotor aerial robot has been designed, based on a complete hardware configuration and a versatile software architecture, which allows the aerial robot to complete all the missions in a fully autonomous and consecutive manner. A thorough evaluation of the proposed system has been carried out in both simulated flights, using the Gazebo simulator in combination with PX4 Software-In-The-Loop, and real flights, demonstrating the appropriate capabilities of the proposed system for performing high-level missions and its flexibility for being adapted to a wide variety of applications.**

I. INTRODUCTION

In recent years, Unmanned Aerial Vehicles (UAVs) have received an increasing attention from the research community, especially in multi-rotor aerial robots due to their high versatility and maneuverability in cluttered indoor environments. UAV navigation in indoor unstructured environments is a very challenging task due to the lack of Global Navigation Satellite System (GNSS) information, which impose an important limitation for localization purposes. Another important limitation is the absence of any knowledge about the area to be explored. This implies not only that the layout is unknown, but also that the location of the obstacles in the map is undetermined. For solving the mentioned limitations, research efforts should aim towards the development of algorithms for providing an appropriate understanding of the explored environment.

Most of the systems that have been developed for autonomous navigation in indoor environments can be classified into two categories: systems that make use of image sensors versus systems whose navigation capabilities are based on active sensors such as lasers, sonars, etc.

The main motivation of using image sensors is their light weight and low power consumption. For this purpose, several approaches have utilized visual markers for simplifying the localization problem [1], [2], [3], [4]. E.g. Pestana et al. [3] make use of AruCo visual markers [5], the approach proposed by Jayatilleke and Zhang [1] requires a prior knowledge of all the landmark poses, whereas in Faig et al. [2] the UAV is required to keep external markers always visible. In these approaches, the use of visual markers provides an accurate localization, with the drawback of altering the area to be explored.

Other approaches based on image sensors, perform the navigation and collision avoidance using the information computed by optical flow algorithms in order to obtain a depth map [6], [7], where Pyramidal Lukas-Kanade is used for the estimation of the optical flow.

Other interesting and more novel approaches use Deep Learning for indoor scene classification and navigation [8], by utilizing a Convolutional Neural Network (CNN) to learn a controller strategy based on data captured from flights performed by a human expert pilot.

Other approaches based on active sensors have obtained very good results for indoor navigation in cluttered environments. Several of these approaches rely on the pose estimation computed using laser measurements [9], [10], [11], [12]. In other approaches, depth information is used for obstacle identification and avoidance and is usually obtained using RGB-D cameras [13], [14], [15]. E.g. In Lange et al. [13] a Microsoft Kinect is used to identify obstacles and localize the UAV for navigating inside corridors. In Lipello et al. [14], 3D features extracted from a monocular odometry algorithm are coupled with depth data provided by the IR sensor of an Asus Xtion RGB-D camera.

This paper presents a complete system for solving the missions designed for the IMAV 2016 competition in a completely autonomous manner, but always with the purpose of developing generic and versatile approaches that can be utilized in many other applications. With this objective in mind, the proposed aerial robot has been designed and equipped with several active sensors (Hokuyo Laser scanner, laser altimeter, RGB-D front camera) and passive sensors (fisheye lens bottom camera), which together with the developed software architecture provide the UAV with the ability of performing high-level missions in unstructured indoor scenarios. For obtaining this high level of autonomy, in this paper we extend the functionalities of our previous works [3], [4], [16], [17], designing new algorithms and software components that improve the Mission Planning, Navigation and Mapping, Pose Estimation and Object Recognition capabilities.

The rest of the paper is structured as follows: Section II presents the problem statement proposed by the IMAV 2016 competition. In Section III the proposed hardware configuration for the designed UAV is described. In Section IV all the software components and algorithms developed for

[1]All authors are with the Computer Vision and Aerial Robotics (CVAR) Group, Centre for Automation and Robotics (UPM-CSIC), Calle José Gutiérrez Abascal 2, Madrid (Spain). `carlos.sampedro@upm.es`

solving the different missions of IMAV 2016 are explained in detail. Section V presents the experiments performed in simulated and real scenarios, with their respective results, and finally, section VI concludes the paper, as well as points towards future research directions.

II. PROBLEM STATEMENT

The IMAV 2016 Indoor Competition proposed a complicated challenge, encouraging aerial robots to perform high-level missions (e.g. Explore, Find a Target, etc) in a fully autonomous way. For solving the proposed missions, UAVs were required to explore an unknown square area of 13 m $\times$ 13 m (named Mission Zone) with unknown layout and unknown location of obstacles, taking-off from another area of 13 m $\times$ 5 m (named Operating Zone) located 10 m away from the Mission Zone.

A detailed description of the different indoor missions to be performed by the UAV can be found in the official IMAV 2016 web page[1]. For a better understanding, a summary of the different missions is provided in the following points.

1. Taking-off, either from the floor or from a moving platform located in the Operating Zone.
2. Entering the Mission Zone via three possible entries: doorway (1.2 m $\times$ 2.0 m), window (1.0 m $\times$ 1.0 m) or chimney (1.0 m $\times$ 1.0 m).
3. Picking and releasing cylindrical objects, which are classified into two categories: Items ($\varnothing$0.1 m and 0.1 m height) and Buckets ($\varnothing$0.25 m and 0.3 m height), which can either be painted in red (BucketA, ItemA) or blue (BucketB, ItemB) color.
4. Exiting the Mission Zone via the door or window.
5. Landing inside the Operating Zone, either on the ground or the moving platform, which follows a periodic trajectory.
6. Generating an indoor map of the Mission Zone.

Considering all the aforementioned challenges, this paper focuses on the development of a fully autonomous aerial robotic platform that is capable of performing high-level missions in a consecutive way and is able to dynamically adapt the predefined exploration mission when an object recognition event is detected.

III. HARDWARE CONFIGURATION

The IMAV 2016 missions require strong miniaturization and long endurance capabilities. Furthermore, several sensors and high-end computing capabilities should be integrated onboard the UAV in order to perform the missions autonomously. Additionally, actuators should be integrated for interaction with objects. All these factors made the platform design a very complex challenge.

To overcome the aforementioned challenge, a custom UAV was built with a total takeoff weight of 3.2 kg, a maximum payload capacity of 1 kg and a maximum flight time of 12 min. An Intel NUC 6i5SYK was the onboard

[1]IMAV 2016 competition rules: http://www.imavs.org/2016/

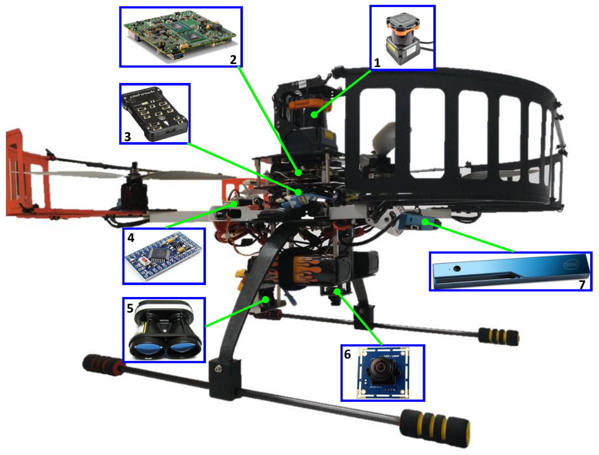


Fig. 1: Quadrotor aerial robot designed for IMAV 2016, showing 1) Hokuyo laser range finder UTM-30 LX. 2) Intel-NUC 6i5SYK computer. 3) Pixhawk autopilot. 4) Arduino pro-mini. 5) Lightware SF10/A altimeter. 6) Fisheye bottom camera. 7) Intel realsense RGB-D camera.

computer utilized. The autopilot used onboard was Pixhawk [18] with its low-level controllers. Proprioceptive sensors used included an Inertial Measurement Unit (IMU) and a magnetometer, both integrated in the Pixhawk autopilot. A Hokuyo Laser range finder UTM-30 LX, an Intel Realsense RGB-D camera, an standard RGB 180 degree fisheye lens camera (bottom camera) and a Lightware altimeter SF10/A were the exteroceptive sensors modularly integrated on the UAV framework (see Fig. 1). The communication between the autopilot, proprioceptive, exteroceptive sensors and the onboard computer was performed over a USB connection.

For performing the release tasks of IMAV 2016, two analog servos were included on the UAV framework and were controlled using PWM signals from an onboard Arduino pro-mini board. The communication between the Arduino and the onboard computer was performed using a USB connection.

IV. SOFTWARE ARCHITECTURE

Having as reference the software architecture presented in our previous works [16], [17], in this paper we extend their functionalities by implementing new components for improving the Planning, Situation Awareness, Feature Extraction and Motor systems (see Figure 2), in order to solve the missions proposed in the IMAV 2016 Competition. In the next sections, a detailed explanation of the new functionalities implemented in each component is provided.

A. *PLANNING SYSTEM*

1) Global Mission Planner: The Global Mission Planner (GMP) centralizes the global intelligence of the system in terms of mission planning. This component has been designed for dynamic mission generation and management for a single UAV or for a swarm of UAVs, as presented in [17]. In the next paragraphs the new functionalities implemented in the GMP are described in detail.

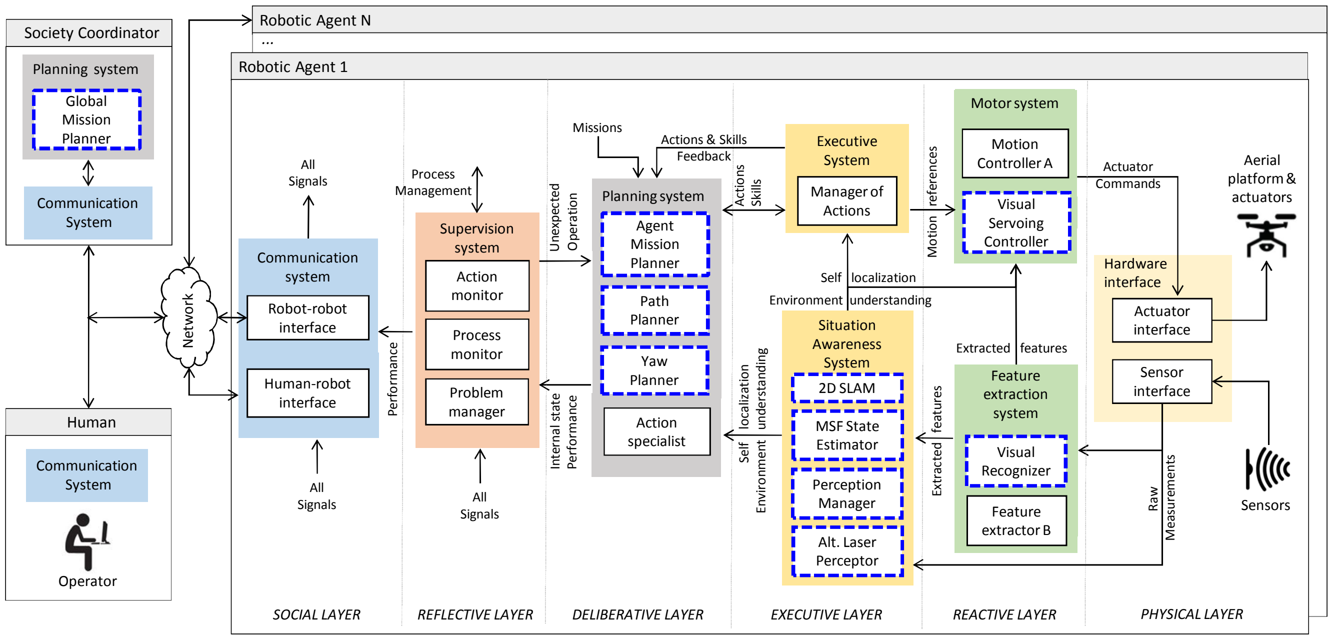


Fig. 2: Software architecture based on Aerostack [16]. Blue and dotted lines depict the components that have been implemented or whose functionalities have been extended for solving IMAV 2016 missions.

- Mission generation: The GMP allows the mission generation in two different operating modes:
  - Manual mission generation: In this new operating mode, the mission is provided by a human operator, which can define the default mission (task by task) to be performed by the UAV (or swarm of UAVs) in an XML-based language.
  - Automatic mission generation: In this operating mode, as explained in [17], the input to the GMP from the human operator, is a high-level mission command (e.g. Explore, Search & Rescue, Find target, etc) and the dimensions of the area in which the mission has to be performed.
- Concatenation of missions: A new functionality for concatenating missions has been implemented in the GMP. In this way, the GMP provides the ability of dynamically reacting to some events that can occur during the execution of the current mission (e.g. the target object has been found in the exploring area). Once an event has been detected, the GMP generates a new mission based on the detected event, thus, saving the state of the previous mission that was being performed and commanding the new generated mission to the Agent Mission Planner (AMP) (see Section IV-A.2).

  Once the new generated mission has been accomplished by the UAV, the GMP recovers the previous mission in its corresponding state (i.e. current task). This functionality of concatenation of missions, with their respective state, provides the system a high-level reactive behavior which allows the UAV (or agent in the swarm) to perform complicated high-level missions with a high level of flexibility and autonomy.
- Event handling: For improving the event handling capabilities of the system, new communication methods with the Perception Manager (see Section IV-B.3) have been implemented in the GMP. In these new methods, the GMP is in charge of processing *mission adaptation events* (e.g. *go for picking*, *go for releasing*) coming from the Perception Manager and generating the new mission based on the received event.

2) Agent Mission Planner: The Agent Mission Planner (AMP) is the component of the mission planning system responsible for scheduling the different tasks that compose the mission sent by the GMP, acting as an interface between a high-level mission received by the GMP and the execution of the actions needed for the accomplishment of such mission. In this section, we extend the work presented in [17], in order to obtain a more specialized behavior of the AMP.

In the next paragraphs the new functionalities implemented in the AMP are described in detail.

- Dynamic mission point re-planning: This functionality allows the AMP to generate obstacle-free points when the actual mission point lies in an obstacle. This behavior is activated when the AMP receives a message of *goal in obstacle* from the Path Planning system (see Section IV-A.3). For generating the obstacle-free point, the AMP generates random points in an iterative manner, located on a circumference of a predefined safety radius, until one of the generated safety points is obstacle-free.
- Centralization of communications: Based on the new components implemented in the global architecture (see Figure 2), new communication features have been included in the AMP:

– Task monitoring:

  ◦ Communication with the GMP: In this case, the communication with the GMP is extended, with the purpose of obtaining a more robust tracking of the task identifier by the GMP. This capability facilitates the mission concatenation feature explained in Section IV-A.1.

  ◦ Communication with the Path Planning system: In this communication resides most of the navigation capabilities of our system. Once a task of *MoveToPoint* is scheduled by the AMP, a *goal* message is sent to the Path Planning system which generates the obstacle-free trajectory to reach the commanded mission point. Once the UAV reaches the current mission point, a *goal reached* message is sent by the Path Planning system to the AMP, which triggers the scheduling of the next task in the mission.

    As the layout of the area to be explored is unknown before the mission starts, and the 2D map provided by the 2D SLAM component (see Section IV-B.1) is being built during navigation, some of the predefined mission points can be detected as lying in an obstacle. This circumstance is perceived by the Path Planner component, which sends the corresponding *goal in obstacle* message to the AMP, which reacts according to the aforementioned dynamic mission point re-planning capability.

– Action handling:

  ◦ Communication with the Motor System: This communication has been implemented for object interaction purposes, and its objective is twofold. First, interacting with the Visual Servoing component for achieving high accuracy in the object interaction tasks. For this, the AMP receives a *target locked* message from the Visual Servoing component when the object is at the desired position in the image (see Figure 6 in Section V). Second, sending the appropriate command to the corresponding servo for releasing the carried object. This means that when the object of interest (e.g. the container for releasing an object that is being carried by the UAV) is at the desired distance to the UAV, the AMP is the responsible of sending the release action.

3) Path Planner: The path planner component relies both on the use of accurate sensors and a robust mapping algorithm. The proposed path planning algorithm is based on an existent ROS (Robot Operating System) [19] navigation package [20], meant for differential-drive and holonomic-wheels robots.

This package is widely used in ground robots navigation and it is robust in the navigation trough static and dynamic obstacles. In this work, the aforementioned navigation package has been adapted for multi-rotor UAV navigation. To the authors knowledge, this is the first time this package is used for multi-rotor UAV navigation. We consider it a suitable candidate for our purposes, due to its robustness against dynamic obstacles and ease of integration.

The path planner component receives a 2D occupancy grid map as an input (see Section IV-B.1), as well as a mission point (in world frame of reference), generated by the AMP. Based on the 2D occupancy grid map and an inflation radius parameter, a 2D cost map is generated, in which cost values are propagated out of occupied cells. There can be *lethal*, *inscribed*, *possibly circumscribed*, *freespace* or *unknown* costs in the cost map. The cost map generator is out of the scope of this paper, and can be reviewed in [20].

The optimum obstacle-free path is computed based on the 2D cost map. In order to provide 3D navigation capabilities, a new functionality has been implemented on top of the original package, which only provides 2D navigation capabilities. Thus, this 2D path is converted into a 3D path, in order to enable 3D navigation. Due to the constraints of the mission and the complexity of the environment, the remaining coordinate of the path, in this case the altitude, is configured as a constant value. An example of the local 2D cost map is provided in Figure 5b around the UAV.

4) Yaw Planner: The yaw planner, which defines the orientation of each waypoint of the path, follows a simple policy:

- Middle waypoint: Orientation is set to a constant value of 0, $\pi/2$, π or $3\pi/2$, depending on the direction vector at each time step.
- Last waypoint: Orientation is derived from the Mission Planning directives.

B. SITUATION AWARENESS SYSTEM

1) 2D Localization and Mapping: A state-of-art 2D-SLAM algorithm [21] has been integrated into our system, due to its robust scan matching and low computational cost as compared to more sophisticated 3D-SLAM approaches.

It combines a robust scan matching technique using a LIDAR system with a 3D attitude estimation system based on inertial sensing. By using a fast approximation of map gradients and a multi-resolution grid, reliable localization and mapping capabilities in a variety of environments are provided [21]. The map is encoded into a 2D occupancy grid map, including *occupied*, *non-occupied* and *non-explored* cell types.

The 2D-SLAM component can be used without odometry information as well as on platforms that exhibit roll/pitch motion. As an example, a 2D occupancy grid map of a real indoor scenario is depicted in Figures 6a and 6c.

2) Multi-sensor Fusion State Estimator: In all our previous architectures [3], [4], [16], [17] the solutions proposed to the localization problem consisted of a SLAM algorithm fusing the position measurements from ArUco visual markers [5] with the orientations from the IMU and ground speeds provided by the optical flow sensor. The main drawbacks of these architectures were first, that the UAV could only explore predefined areas marked with ArUco visual markers and second, that a well illuminated and textured floor was needed for the optical flow sensor to provide accurate measurements.

To overcome the limitations of our previous systems, in which the localization problem was addressed by a SLAM algorithm based on visual markers (e.g. ArUco) and Optical Flow measurements, in this paper a more versatile localization approach based on the combination of three components is proposed. The first component consists of the 2D SLAM algorithm explained in section IV-B.1. The second component is based on a Flight Altitude State Estimator (see section IV-B.2.a), providing accurate information of the flight altitude as well as the elevation of the ground. The third component consists of a complete Robot State Estimator (see section IV-B.2.b), in which the 2D SLAM data is fused with the measurements coming from the IMU and the Flight Altitude State Estimator.

a) Flight Altitude State Estimator: In this paper a modular multi-sensor fusion technique to estimate the flight altitude as well as the elevation of the ground below the UAV is proposed, based on an Extended Kalman Filter algorithm. It consists of a kinematic process model without any robot inputs and a process noise. The proposed component fuses the measurements from the distance sensor (i.e. the measurements from a laser altimeter), IMU (accelerometer and gyroscope) and barometer. The outputs of the Flight Altitude State Estimator component are: vertical linear position, velocity and acceleration, attitude and angular velocities of the UAV w.r.t. the world frame of reference. Additionally, the elevation of the ground (ground object height) w.r.t. the world frame of reference is provided as an output. Furthermore, the Flight Altitude State Estimator auto calibrates the barometer and the accelerometer sensors for estimating the bias of the sensors.

b) Robot State Estimator: A ROS package [22] was used for the robot state estimation. The key motivation of using this package was the ease of integration of an arbitrary number of sensors. The inputs to the Robot state Estimator component are the IMU measurements, the 2D estimated pose of the robot coming from the 2D SLAM component and the estimated flight altitude of the robot provided by the Flight Altitude State Estimator. The outputs provided by the Robot State Estimator are the pose, velocity, and linear acceleration of the robot w.r.t. world frame of reference.

3) Perception Manager: The Perception Manager manages and centralizes all the visual recognition events that can occur during the execution of the current mission. For this purpose it includes the functionalities that are listed below:

- Object state management: One of the main functionalities of the Perception Manager is the management of the states of the objects which the UAV can interact with. The number and characteristics of these objects (i.e. size and color), as well as their initial state (see Table I) are provided to the Perception Manager in a configuration file, which is loaded at the beginning of the mission.

  During the execution of the current mission some perception events can occur (e.g. *object recognized*, *picked* and *released*). When a perception event occurs, the Perception Manager actuates by updating the objects states and its own internal state. The possible internal states of the Perception Manager have been considered based on the type of missions to be performed in the IMAV 2016 competition. These internal states are: $\{$*Exploring, Going For Picking Item, Going For Releasing Item*$\}$

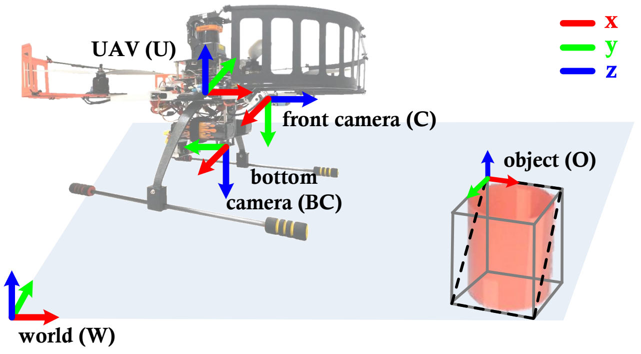


Fig. 3: Relevant coordinate frames of our proposed solution. Dotted line in the object represents the sloped plane used for pose estimation purposes.

TABLE I: Example of the initial state of the objects in a Search & Release Mission

Object \ State	Recognized	Picked	Released
Item A	0	1	0
Bucket A	0	0	0
Item B	0	1	0
Bucket B	0	0	0

- Object mapping: The Perception Manager provides a higher level of consciousness over the Visual Recognition component (see Section IV-C.1). It has the knowledge of the UAV pose w.r.t. to the world frame of reference, and in addition, it incorporates a proprioceptive knowledge of the UAV (i.e. the position of all the sensors and actuators w.r.t. the UAV frame of reference). Using this information, the Perception Manager is capable of providing a higher level of understanding to the received perception event. For example, for a visual recognition event, this consists of converting the pose of the object that has been detected (in the frame of reference of the camera) to the frame of reference of the world (see Figure 3).

  The 4×4 homogeneous transformation utilized inside the Perception Manager is computed by Eq. 1

$$ {}^{W}\boldsymbol{T}_{O} = {}^{W}\boldsymbol{T}_{U} \cdot {}^{U}\boldsymbol{T}_{C} \cdot {}^{C}\boldsymbol{T}_{O} \qquad (1)$$

  where ${}^{W}\boldsymbol{T}_{O}$ is the global transformation from world to the object frame of reference. ${}^{W}\boldsymbol{T}_{U}$ is the transformation from world to the UAV frame of reference, which is given by the Robot State Estimator component. ${}^{U}\boldsymbol{T}_{C}$ represents a rigid transformation from the UAV to the front camera reference frame, and ${}^{C}\boldsymbol{T}_{O}$ provides the transformation from the camera to the object frame of reference (see Figure 3).

- Object position filtering: Another important functionality of the Perception Manager is the capability of filtering out some of the perception events coming from the Visual Recognition component. This filtering is performed in two stages:
  - Based on the object altitude position: the Perception Manager can reject some perceptions if the requirements in terms of the z coordinate (w.r.t. the world reference frame) of the object are not satisfied (i.e. $0 \leq z_{bucket} \leq 0.3$ m and 0.6 m $\leq z_{item} \leq 1.0$ m).
  - Based on accumulated buffer: for this purpose the Perception Manager has a buffer of N position measurements. Once the N_{th} measurement is obtained, the filter actuates iteratively by computing the centroid of the buffer, rejecting the 20% of the measurements that are farthest from the centroid. Once this process is finished, the centroid of the remaining measurements is taken as the definitive position of the recognized object, which corresponding state is updated.
- Mission adaptation event: When a perception event is received, the Perception Manager evaluates the current state of the objects (see Table I) and taking into account the received event, it generates a *mission adaptation event* to the GMP (e.g. if Item A is picked and Bucket A is recognized, then a *Go For Releasing Item A* event is generated).

4) Altimeter Based Moving Object Perceptor: As stated in Section II, landing maneuver has to be usually performed at the end of an entire mission, which imposes several limitations related to the time needed for performing such maneuver. In addition, current battery technology imposes high constraints to the power onboard and to the time available to perform the landing maneuver. To overcome these issues, a power-efficient and fast landing approach based on altimeter measurements is proposed.

As presented on the IMAV 2016 rules, the landing platform performs a periodic trajectory. This knowledge is used for our proposed landing approach. Taking into account n periods of the landing platform, its velocity can be estimated. Our approach relies on the time the landing platform is underneath the UAV, which can be estimated using altimeter measurements. Once this time is estimated, and assuming the size of the object as known, the speed of the landing platform can be determined. Additionally, the time to fall from the moment in which the landing command is sent has been experimentally measured.

Thus, based on the estimated speed and the time-to-fall, the precise instant at which the landing command has to be sent is computed.

C. FEATURE EXTRACTION SYSTEM

1) Visual Recognizer: This component of the architecture is responsible for the object recognition in terms of the detection and classification of the objects of interest in the images captured by the RGB cameras mounted onboard the UAV, and the 3D location of those objects w.r.t. the frame of reference of the camera. All the algorithms developed within this component have been implemented with the objective of minimizing the computational cost tackling with the real-time requirements of a fully autonomous UAV.

For achieving the mentioned objectives, the Visual Recognizer component is mainly composed of two modules:

- Visual Object Detector: The objective of this module is twofold. First, localizing the object of interest in the image plane by providing the corresponding ROI (Region of Interest) in the image, and second, computing the 3D location of the object by providing the relative pose w.r.t. to the camera frame of reference. For achieving these objectives, three main functionalities have been implemented within the Visual Object Detector:
  - Color Image Processing: This submodule is in charge of processing the images using color information. For the purpose of detecting the objects proposed in the IMAV 2016 (red and blue objects) the HSV color space has been utilized for segmenting the image, whose optimal ranges for each channel have been derived experimentally.
  - Shape Image processor: This submodule takes into account the information related to the 3D geometry of the corresponding object to be detected. For solving the object detection challenge within the IMAV competition, cylindrical objects were modeled by a 3D sloped plane assuming that the UAV is always flying at higher altitude than the objects position (see Figure 3).
  - Pose Estimator: This submodule provides the Visual Object Detector the functionality of computing the 3D location of the detected object w.r.t. to the frame of reference of the camera. For achieving this, an iterative Perspective-*n*-Point (P*n*P) algorithm based on minimization of the reprojection error using Levenberg-Marquardt over RANSAC filtering is used. The implementation provided in [23] was utilized achieving accurate pose estimation at real-time frame rates.
- Visual Object Classifier: This module is in charge of classifying the detected objects among the predefined four categories established by the IMAV 2016 competition. For this purpose, prior knowledge in terms of color and size of the objects was utilized (see Section II). Thus, for solving the classification task, the Visual Object Classifier takes as input the ROI in the image provided by the Visual Object Detector and utilizes the decision criteria presented in Table II.

The final output provided by the Visual Object Recognizer component consists of the detection and classification in the image plane of the recognized object, with its corresponding 3D position w.r.t. the camera frame of reference (see Figure 4)

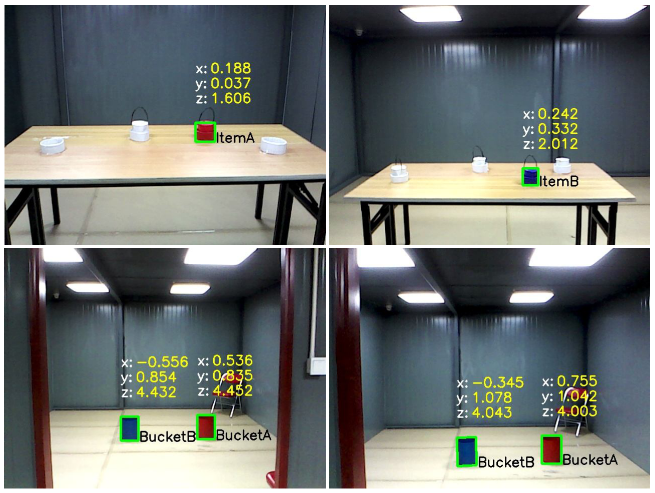


Fig. 4: Visual Object recognition during participation in the IMAV 2016 competition. (Top row) Recognition of Item objects. (Bottom row) Recognition of Bucket objects. The relative pose of the object with respect to the camera is computed using a Perspective-*n*-Point (P*n*P) algorithm. In this Figure, z represents the depth.

TABLE II: Criteria utilized for object classification by the Visual Object Classifier

Color \ ROI aspect ratio	1.0	0.833
Red	ItemA	BucketA
Blue	ItemB	BucketB

D. *MOTOR SYSTEM*

In our implementation, the motor system comprises two components: the motion controller and the visual servoing controller, which are described below.

1) Motion controller: The motion controller consists of a general-purpose configurable controller capable of operating as a velocity, position and trajectory controller simultaneously. Our architecture makes use of the original implementation described in [24].

2) Visual servoing controller: Our solution provides object-relative UAV positioning capabilities based on visual servoing. Given a desired position of the UAV with respect to the object of interest, visual feedback is used to estimate the position error. UAV position estimations in world coordinates are used in conjunction with the estimated position error in order to obtain the references for the motion (position) controller in world coordinates.

Estimated position errors correspond to position errors of the camera with respect to the object of interest. A simple, rigid transformation between the camera and the UAV coordinate systems can be used to obtain the position error of the UAV with respect to the object of interest.

Visual feedback is provided to the controller in the form of estimations of the location of the object of interest in the image plane. These estimations are computed by the Visual Recognition component and delivered as a regions of interest (ROIs). Given these ROIs, Eq. 2 can be used to estimate position errors of the camera with respect to the object of interest. In this equation, ΔX, ΔY and ΔZ represent the position errors of the on-board bottom-looking camera with respect to the object of interest in meters, (x_{ref}, y_{ref}) are the image coordinates of the desired positions of the center of the ROI, (x, y) are the current ROI's 2D coordinates, d_{ref} is the desired hovering distance above the object of interest, fx, fy are the camera's focal lengths in pixels, O_a is the object area, measured in squared meters and A_{im} is the area of the ROI (width $\times$ height).

$$\begin{bmatrix}\Delta X\\ \Delta Y\\ \Delta Z\end{bmatrix} = \begin{bmatrix}\frac{(x_{ref}-x)d_{ref}}{f_x}\\ \frac{(y_{ref}-y)d_{ref}}{f_y}\\ d_{ref}-\sqrt{\frac{f_x f_y O_a}{A_{im}}}\end{bmatrix} \tag{2}$$

The desired positions of the center of the ROI are obtained empirically. These positions correspond to the 2D coordinates of the ROI's center when the UAV is located in the desired position with respect to the object of interest.

As a safety measure, if estimated position errors are larger than 2 m in any direction no updates in the position reference are given to the motion controller. The same measure is applied if the distance between the last computed position reference and the current one is larger than 2 m in any direction.

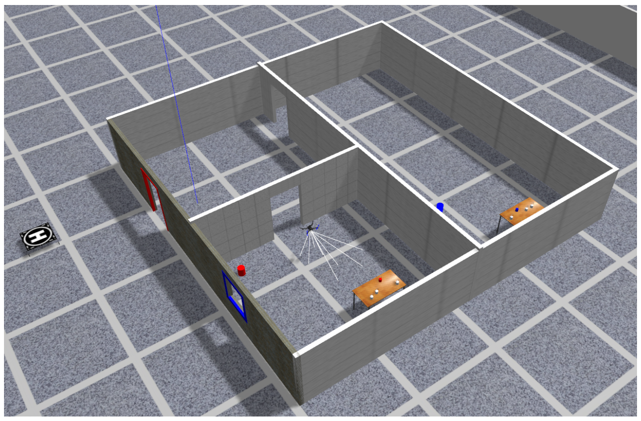
Additionally, a buffer accumulating the last n position errors has been implemented and can be used as an indicator of the positioning stability in order to automatically trigger specific actions involving interaction with the object of interest (e.g. pick up, release).

V. EXPERIMENTS AND RESULTS

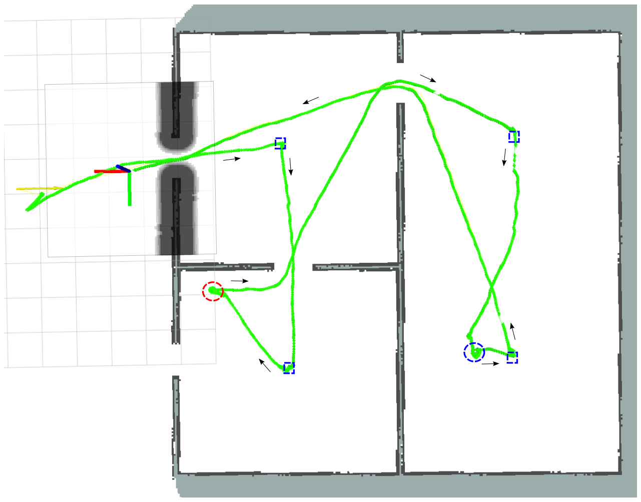
The aim of this section is to present the experiments that have been designed for evaluating the capabilities of the proposed hardware and software systems in order to solve the different high-level missions designed in the IMAV 2016 Competition. This evaluation involves the validation of three main blocks: The proposed framework and hardware configuration, the designed software components, and the different developed algorithms. For a complete evaluation of the two latter blocks, a detailed simulation setup has been designed. Additionally, for a thorough evaluation of the three main blocks as a whole, several experiments consisting on real flights in unstructured indoor environments with different layouts have been conducted.

A. *Experimental Setup*

In this section, the proposed experimental setup for simulated and real flights is described. All the developed software components have been implemented within the Aerostack architecture [16] using C++, under the standard C++ 11, and using ROS (Robot Operating System) [19] as the communication framework between them. The Operative System utilized for running the different processes in both simulated and real flights is Ubuntu 14.04 LTS.



(a) Gazebo simulation environment of a 13 m × 13 m area.



(b) Trajectory and 2D map generated during the execution of the simulated flight. Cost map is depicted around the UAV.

Fig. 5: Simulated flight experiment. (a) Gazebo environment designed for reproducing the IMAV 2016 competition layout. (b) 2D occupancy grid map and trajectory generated by the UAV (green color). The colored squares depict the mission points automatically generated by the GMP, while the colored circles represent the Buckets.

1) Simulated Flights: A simulation environment has been created using Gazebo robot simulator [25] in combination with PX4 Software-In-The-Loop (SITL). The designed simulation using SITL allows a realistic emulation of different sensors that are utilized by several algorithms, such as the readings received from lasers and cameras.

Following the guidelines proposed in IMAV 2016 Competition rules, and with the purpose of obtaining a complete validation of the proposed system, several simulated scenarios of dimensions 13 m × 13 m (see Figure 5a) have been designed incorporating variability between each other in terms of layout, location of obstacles and objects to be found (i.e. Buckets and Items), etc.

2) Real Flights: A real indoor environment with dimensions 9 m × 11 m has been designed, in which the Mission Zone consists of a rectangular area of 9 m × 6 m. With the aim of testing the proposed system in a similar scenario to the real one designed by the IMAV 2016 competition, a replica of the main façade has been built, in which the dimensions of the possible entries (i.e. door and window), as well as the distance between them, are exactly the same as those stated in the IMAV 2016 rules.

Using the Mission Zone presented in Figure 6, several real flights have been conducted in order to perform a thorough validation of the proposed system in real indoor scenarios. Inside the Mission Zone, several layouts have been designed, modifying the number and dimensions of the rooms and the location of obstacles (i.e. tables, chairs, etc.) and objects to be found.

B. Results and Discussion

The results presented in this section demonstrate the accomplishment of most of the proposed missions in IMAV 2016 competition in a fully autonomous and concatenated manner. The missions in which the presented experiments and results are focused are the following: take-off, enter the Mission Zone (via the door or window), explore the unknown layout of the Mission Zone in order to recognize the Buckets, perform the release operation of Items, exit the Mission Zone (via the door or window), and land (in either the ground or the moving platform). For achieving the release missions, the two Items are preloaded in the arms provided by our UAV.

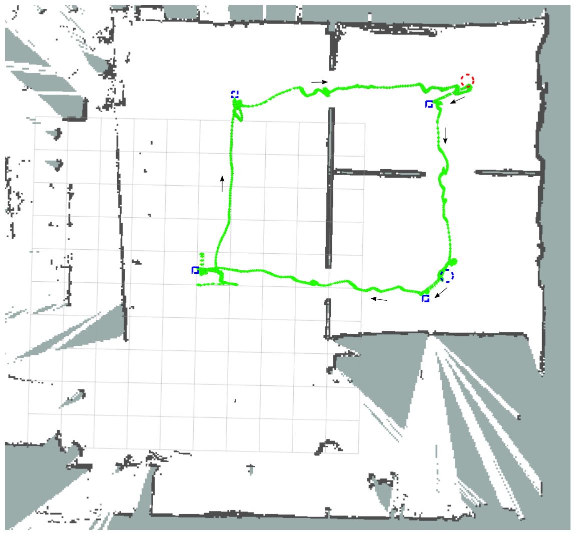
Figure 5 shows the results obtained during the execution of a simulation flight that involves the accomplishment of all the mentioned missions consecutively. As can be noticed in Figure 5b the UAV is able to accomplish the entire exploration mission and in addition, throw the onboard Items into the recognized Buckets. For achieving the latter objective, a mission adaptation for the release missions is performed by the GMP, recovering the previous exploration mission when the release missions are finished.

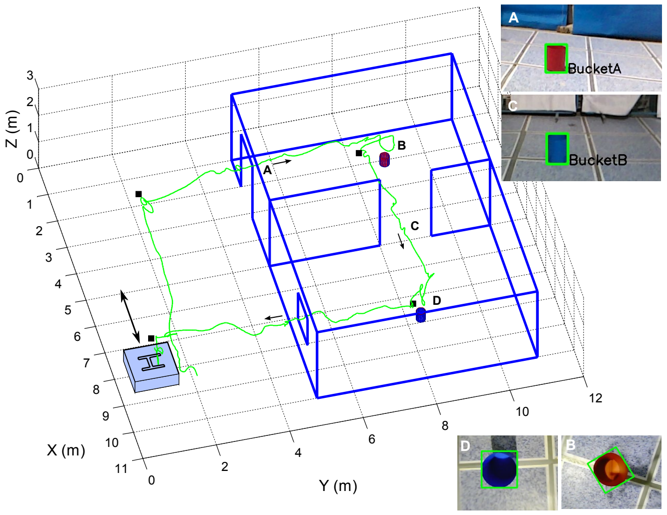
Figure 6 shows the trajectory performed by the UAV during the execution of several real indoor flights in two different scenarios. As can be observed in Figure 6, the UAV is able to navigate performing obstacle avoidance and exploring the unknown layout, recognizing both Buckets and throwing the onboard Items into the corresponding Buckets.

In the first experiment scenario presented in Figures 6a and 6b, the UAV is capable of autonomously exploring the unknown environment searching for the two Buckets, recognizing them, performing the corresponding release operations, and executing the autonomous landing maneuver on a moving platform at the final commanded mission point.

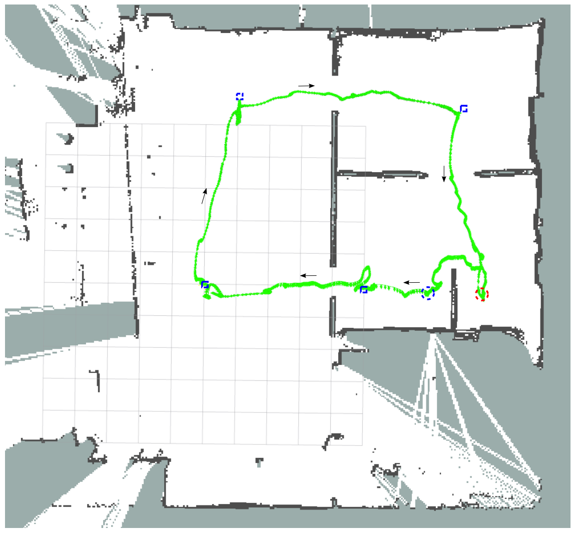
Figures 6c and 6d illustrate the trajectory followed by the UAV in a more complex scenario, where the number of obstacles is increased, and the two Buckets are located very close to obstacles, hampering the release maneuvers. In this scenario, taking-off and landing missions were performed at ground level.

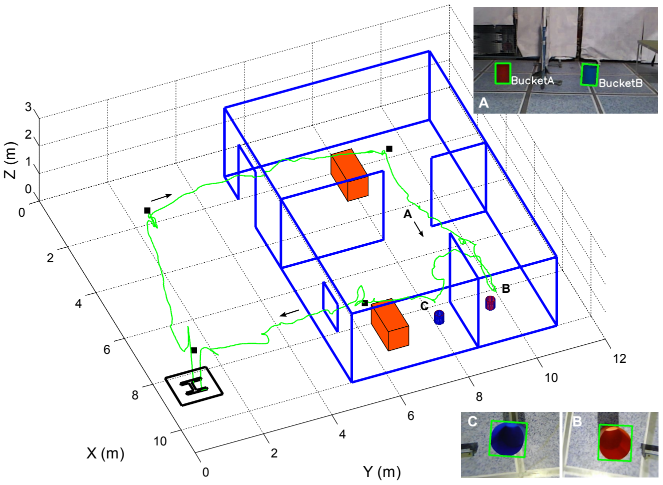
As can be noticed in Figure 6d, once the two Buckets have been recognized by the UAV, the GMP commands the UAV to reach the mission point generated by the estimated position of these objects in the map. Once the UAV reaches the mission point, in which the corresponding Bucket is


(a) 2D map and trajectory performed by the UAV in Scenario 1.



(b) 3D view of the trajectory performed by the UAV in Scenario 1.


(c) 2D map and trajectory performed by the UAV in Scenario 2.



(d) 3D view of the trajectory performed by the UAV in Scenario 2.

Fig. 6: Real flight experiments in two different indoor scenarios. (a), (c) 2D occupancy grid map and trajectory generated by the UAV (green color). (b), (d) 3D view of the trajectory performed by the UAV (green color). The Mission Zone outline is shown in blue color. Obstacles are represented by orange cuboid and the Buckets by their corresponding cylinder. The squares depict the commanded mission points. Images represent the object recognition at the specific point of the trajectory.

located, the next task of performing Visual Servoing is scheduled by the AMP, based on the visual reference given by the Visual Recognizer over the image captured by the bottom camera. For obtaining an accurate release maneuver, the Visual Servoing component commands the UAV at 0.7 m height, which can be well appreciated in Figure 6d at positions [x: 8.5, y: 9.5] for the red Bucket and [x: 8.5, y: 8.0] for the blue one.

We refer the reader to a description video[2], in which the capabilities of the proposed approach are demonstrated, showing the results obtained in simulated and real flights.

[2]A video demonstration of the reported results has been made available at: https://vimeo.com/205775713

VI. CONCLUSIONS AND FUTURE WORK

In this paper, a fully-autonomous quadrotor aerial robot for solving the missions designed in the IMAV 2016 International Indoor Competition has been presented. The proposed system is an effort on developing a fully autonomous UAV with the appropriate capabilities for performing the high-level missions presented in IMAV 2016. For achieving the objectives proposed in the mentioned competition, our system provides a complete hardware configuration and a flexible software architecture. The latter is composed by several components that provide versatile functionalities such as a flexible and dynamic Mission Planning system that allows mission re-planning in real-time and event management that can occur during the mission (e.g. a target object has been recognized), a reactive collision avoidance navigation based on laser information, a low-cost and highly effective object recognition system based on predefined information of shape and color, a complete multi-sensor fusion system for accurate pose estimation and altitude filtering, among others. All these functionalities have been implemented taking into account a multi-robot structure, thus the proposed system is able to manage swarms of UAVs with the mentioned capabilities.

In this work, simulated flights using Gazebo and PX4 Software-In-The-Loop (SITL) have been conducted in order to emulate the conditions in the IMAV 2016 Competition and test the capabilities of the proposed system. In addition, several real flights in an indoor scenario of 9 m $\times$ 11 m with different configurations have been conducted in order to validate the proposed system in real flight conditions. The results presented in this paper, both in simulated and real flights, demonstrate that the proposed aerial robot is capable of performing high-level missions (e.g. Search & Release) in a fully-autonomous manner with the ability to concatenate different kinds of missions in order to manage unexpected events. Furthermore, the proposed system has been validated in IMAV 2016, being the only team capable of performing all the here explained missions in a fully-autonomous and consecutive manner.

In addition, these results demonstrate the versatility and flexibility our solution for being adapted to solve other kinds of high-level missions. The Search & Release missions solved in the designed experiments can be easily adapted to other highly demanded applications such as precision agriculture or packet delivery, which will lead to future experiments.

Future work is lined towards the evaluation of the proposed system using a swarm of UAVs, whose cooperation can be exploited to optimize the execution of several high-level missions. Another scope of research is focused towards the development of more generic visual recognition components based on supervised learning, which in addition can improve the localization and mapping capabilities. Finally, other object interaction approaches (e.g. picking objects) will be addressed.

REFERENCES

[1] L. Jayatilleke and N. Zhang, "Landmark-based localization for unmanned aerial vehicles," in *Systems Conference (SysCon), 2013 IEEE International*. IEEE, 2013, pp. 448–451.

[2] J. Faigl, T. Krajník, J. Chudoba, L. Přeučil, and M. Saska, "Low-cost embedded system for relative localization in robotic swarms," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 993–998.

[3] J. Pestana, J. L. Sanchez-Lopez, P. De La Puente, A. Carrio, and P. Campoy, "A vision-based quadrotor multi-robot solution for the indoor autonomy challenge of the 2013 international micro air vehicle competition," *Journal of Intelligent & Robotic Systems*, vol. 84, no. 1-4, pp. 601–620, 2016.

[4] J. L. Sanchez-Lopez, J. Pestana, P. de la Puente, and P. Campoy, "A reliable open-source system architecture for the fast designing and prototyping of autonomous multi-uav systems: Simulation and experimentation," *Journal of Intelligent & Robotic Systems*, pp. 1–19, 2015.

[5] R. Munoz-Salinas, "Aruco: a minimal library for augmented reality applications based on opencv," *Universidad de Córdoba*, 2012.

[6] S. Zingg, D. Scaramuzza, S. Weiss, and R. Siegwart, "Mav navigation through indoor corridors using optical flow," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 3361–3368.

[7] V. Lippiello, G. Loianno, and B. Siciliano, "Mav indoor navigation based on a closed-form solution for absolute scale velocity estimation using optical flow and inertial data," in *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*. IEEE, 2011, pp. 3566–3571.

[8] D. K. Kim and T. Chen, "Deep neural network for real-time autonomous indoor navigation," *arXiv preprint arXiv:1511.04668*, 2015.

[9] S. Shen, N. Michael, and V. Kumar, "Autonomous multi-floor indoor navigation with a computationally constrained mav," in *Robotics and automation (ICRA), 2011 IEEE international conference on*. IEEE, 2011, pp. 20–25.

[10] S. Grzonka, G. Grisetti, and W. Burgard, "Towards a navigation system for autonomous indoor flying," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 2878–2883.

[11] S. Grzonka, G. Grisetti, and W. BUrgard, "A fully autonomous indoor quadrotor," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 90–100, 2012.

[12] T. Tomic, K. Schmid, P. Lutz, A. Domel, M. Kassecker, E. Mair, I. L. Grixa, F. Ruess, M. Suppa, and D. Burschka, "Toward a fully autonomous uav: Research platform for indoor and outdoor urban search and rescue," *IEEE robotics & automation magazine*, vol. 19, no. 3, pp. 46–56, 2012.

[13] S. Lange, N. Sünderhauf, P. Neubert, S. Drews, and P. Protzel, "Autonomous corridor flight of a uav using a low-cost and light-weight rgb-d camera," in *Advances in Autonomous Mini Robots*. Springer, 2012, pp. 183–192.

[14] G. Loianno, V. Lippiello, and B. Siciliano, "Fast localization and 3d mapping using an rgb-d sensor," in *Advanced Robotics (ICAR), 2013 16th International Conference on*. IEEE, 2013, pp. 1–6.

[15] S. Shen, N. Michael, and V. Kumar, "3d indoor exploration with a computationally constrained mav," in *Robotics: Science and Systems*, 2011.

[16] J. L. Sanchez-Lopez, R. A. S. Fernández, H. Bavle, C. Sampedro, M. Molina, J. Pestana, and P. Campoy, "Aerostack: An architecture and open-source software framework for aerial robotics," in *Unmanned Aircraft Systems (ICUAS), 2016 International Conference on*. IEEE, 2016, pp. 332–341.

[17] C. Sampedro, H. Bavle, J. L. Sanchez-Lopez, R. A. S. Fernandez, A. Rodriguez-Ramos, M. Molina, and P. Campoy, "A flexible and dynamic mission planning architecture for uav swarm coordination," in *Unmanned Aircraft Systems (ICUAS), 2016 International Conference on*. IEEE, 2016, pp. 355–363.

[18] L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "Pixhawk: A system for autonomous flight using onboard computer vision," in *Robotics and automation (ICRA), 2011 IEEE international conference on*. IEEE, 2011, pp. 2992–2997.

[19] M. Quigley, J. Faust, T. Foote, and J. Leibs, "Ros: an open-source robot operating system."

[20] E. Marder-Eppstein, E. Berger, T. Foote, B. Gerkey, and K. Konolige, "The office marathon: Robust navigation in an indoor office environment," in *International Conference on Robotics and Automation*, 2010.

[21] S. Kohlbrecher, J. Meyer, O. von Stryk, and U. Klingauf, "A flexible and scalable slam system with full 3d motion estimation," in *Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. IEEE, November 2011.

[22] T. Moore and D. Stouch, "A generalized extended kalman filter implementation for the robot operating system," in *Intelligent Autonomous Systems 13*. Springer, 2016, pp. 335–348.

[23] G. Bradski, "The opencv library," *Dr. Dobb's Journal of Software Tools*, 2000.

[24] J. Pestana, I. Mellado-Bataller, J. L. Sanchez-Lopez, C. Fu, I. F. Mondragón, and P. Campoy, "A general purpose configurable controller for indoors and outdoors gps-denied navigation for multirotor unmanned aerial vehicles," *Journal of Intelligent & Robotic Systems*, vol. 73, no. 1-4, pp. 387–400, 2014.

[25] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 3. IEEE, pp. 2149–2154.