A Vision Based Aerial Robot solution for the IARC 2014 by the Technical University of Madrid

J. Pestana, J. L. Sanchez-Lopez, R. Suarez-Fernandez, J.-F. Collumeau, P. Campoy
Computer Vision Group, Centre for Automation and Robotics,
Technical University of Madrid (Spain)
J. Martin-Cristobal, M. Molina, J. De Lope, D. Maravall
Department of Artificial Intelligence, Technical University of Madrid (Spain)

ABSTRACT

The IARC competitions aim at making the state of the art in UAV progress. The 2014 challenge deals mainly with GPS/Laser denied navigation, Robot-Robot interaction and Obstacle avoidance in the setting of a ground robot herding problem. We present in this paper a drone which will take part in this competition. The platform and hardware it is composed of and the software we designed are introduced. This software has three main components: the visual information acquisition, the mapping algorithm and the Aritificial Intelligence mission planner. A statement of the safety measures integrated in the drone and of our efforts to ensure field testing in conditions as close as possible to the challenge's is also included.

INTRODUCTION

Statement of the Problem

The IARC 2014 competition aims for competitors to develop solutions for some of the major unsolved problems in UAV systems: GPS/Laser denied navigation, Robot-Robot interaction and Obstacle avoidance. In order to test all of these challenges an indoor arena was created where no static objects are within range. Inside this arena 10 iRobot Create are programmed to navigate randomly and 4 iRobots are carrying poles as dynamic obstacles. The challenge of this scenario is to locate and "herd" at least seven of the iRobots, which are moving pseudo-randomly, toward one end of the arena. The UAV has to interact with them, by almost touching them, while avoiding the dynamic obstacles in less than 10 minutes.

Conceptual solution to solve the problem

In order to overcome the problem of aerial navigation in a GPS/Laser denied environment, we plan on using visual odometry combined with visual Self-Localization And Mapping (SLAM). The identification of targets and threats inside the arena will be done by computer vision means, providing information to an artificial intelligence module in charge of the tactical decisions needed to successfully complete the challenge.

The system will be validated both in simulated environments and in a replica arena designed following the guidelines and informations available in the IARC mission 7-a rules, ensuring test scenario conditions close to the competition's.

The proposed system architecture has six main modules, as can be seen in figure 1, that are explained later.

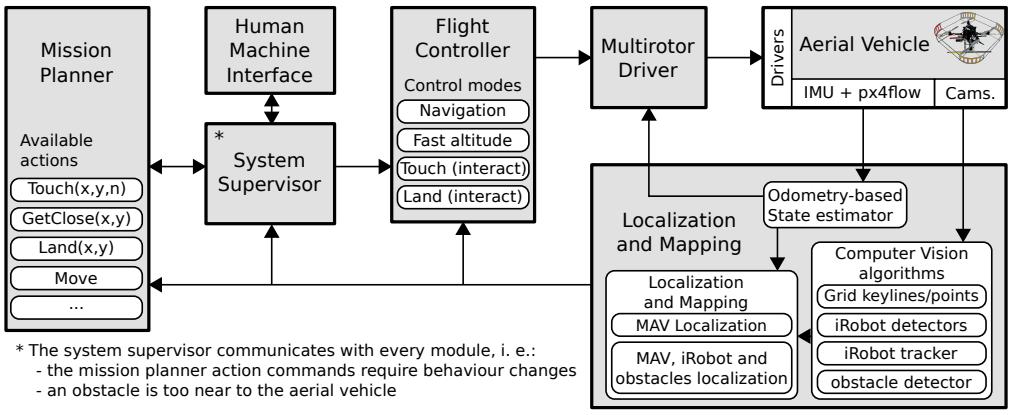


Figure 1: Overall System Architecture and main communication channels between its modules.

Yearly Milestones

We planed the IARC competition as a two-year project, aiming to complete both parts of the 7th mission by the year 2015. The milestones to be achieved by the year 2014 competition are the following:

- Set-up the Aerial Vehicle's hardware and a functional software architecture,
- Develop the simulated environment and a replica arena for field testing,
- Develop a system architecture allowing the team to participate in the IARC while meeting the necessary safety requirements. Performance should be sufficient to complete the guidance of at least four ground target robots and thereby allow the team to finish the competition without being disqualified.

AERIAL VEHICLE

The AscTec Pelican Quadrotor made by Ascending Technologies is used as the main platform. The vehicle structure, motors, and rotors are used without modification, nevertheless, a new landing gear was fabricated along with camera mounts.

The AscTec Autopilot, onboard our robot, is a precise and multifunctional research tool mounted on the UAV. This flight control unit contains all necessary sensors to function as an IMU, it also has two onboard ARM7 microprocessors and various communication interfaces. As explained in the communications section, in our design, the Autopilot receives commands from the Onboard computer as: pitch, roll, yaw speed and total thrust references.

Propulsion and Lift System

The air vehicle chosen for the IARC 2014 is a mid-sized quadrotor. These vehicles obtain the lift force from four fixed-pitch propeller blades. This simplifies the design and maintenance of the vehicle. As a safety advantage these rotors are smaller than an equivalent helicopter rotor.

Guidance, Navigation, and Control

All the modules of the our architecture, depicted in Fig. 1, are explained in this section.

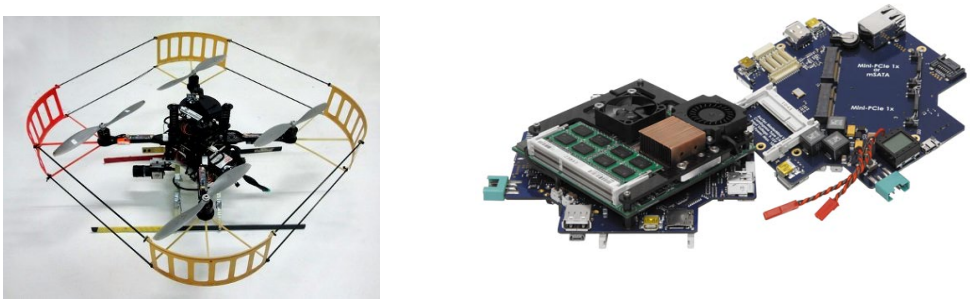


Figure 2: (left) AscTec Pelican Quadrotor (right) AscTec Mastermind Onboard Computer

Localization and Mapping
The Localization and Mapping module is in charge of localizing the drone in the Arena, as well as finding and identifying Target Ground Robots and Moving Obstacles, with enough precision and speed to allow the drone to complete the mission.

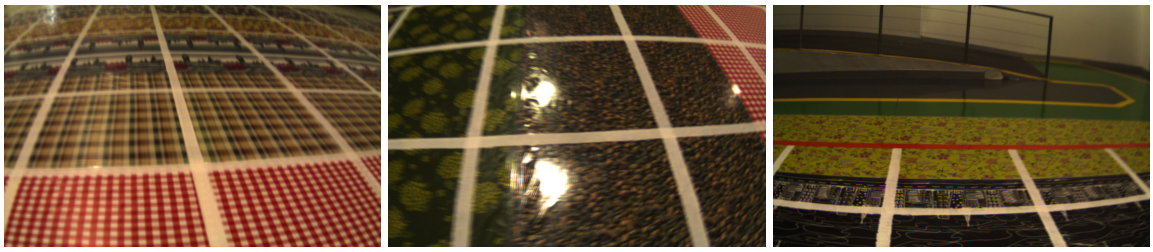
It is composed of the following four main modules:

- Odometry State Estimator, which fuses the measurements received from some of the onboard drone sensors (IMU, Optical Flow and Altitude) to calculate an estimate of the state of the drone, using its module and a standard Extended Kalman Filter (EKF);
- Computer Vision Algorithms (explained later), which processes the pictures taken by the cameras to extract high-level features that can be used easily;
- Drone Localization and Arena Mapping, which takes the grid landmarks given by the CV Module and the estimated pose of the drone given by the Odometry State Estimator to create a local map of the Arena and to locate the drone in this map. Using the information of the key lines (coloured borders of the grid) given by the CV Module and the previous local map, the drone can be located in the Arena;
- Target Ground Robots and Moving Obstacles Localization, which estimates the pose and location of the target ground robots and moving obstacles using the information given by the CV Module.

Computer Vision Algorithms
The information provided by computer vision means is of two natures. First, visual landmarks need to be provided for the mapping of the arena and the localization of the uav inside it. Target- and threat-related information is also needed in order to successfully achieve the competition's objectives.

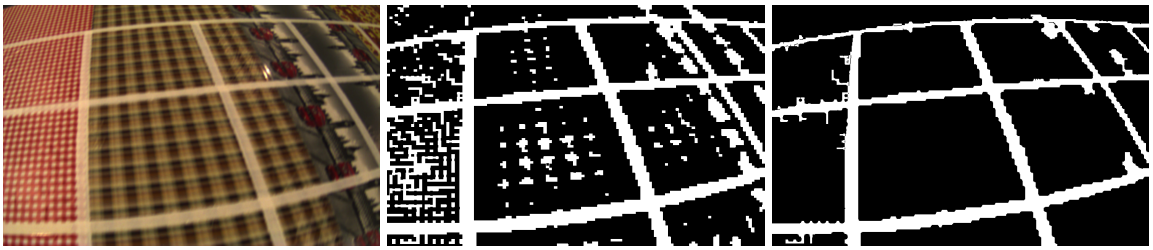
The general appearance of pictures taken in our test arena is as follows: a grid of homogeneous lines caracterized by a high luminance value are drawn on a non-uniform background in which appearance may be locally similar to the grid lines. Possible light reflections on the floor may add high luminance regions to the pictures. In addition, at most one key line may appear in the pictures. Examples can be seen in figure 3.

The visual landmarks we seek are: the points where the vertical and horizontal lines of the grid intersect, and the key lines. In order to extract these landmarks, we propose the following algorithm:

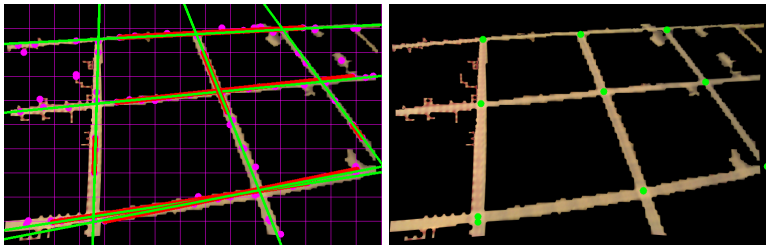


(a) A non-uniform background (b) Light reflections (c) A red keyline

Figure 3: Some pictures of our testing arena taken from our UAV.



(a) Input (b) Threshold mask (c) Floodfilled grid



(d) 2-pass RANSAC (e) Grid intersections

Figure 4: Illustration of the grid intersection extraction.

1. Segment the picture using an adaptive threshold. As the luminance is the most salient characteristic of the lines, we use the Y channel of the YCrCb colorspace;
2. Refine segmentation by floodfilling the grid: which removes of high-luminance blobs;
3. Compute a distance map of the obtained mask.
4. Divide the distance map into subregions and apply RANSAC [4] to identify possible lines in the picture.
5. Use RANSAC again, using the line models by descending vote order.
6. Grid intersections are then computed knowing the equations of the retained lines;
7. Compute the mean color of the neighborhoud of each couple of local maxima having generated one of the retained lines in order to identify the keylines.

This process is illustrated in Fig. 4. In Subfig. 4(d), green lines are the result of the two-pass RANSAC. Red lines link the two generating points of each green line, which are used for identifying key lines.

In addition to the static components of the arena described in the previous paragraph, iRobots may appear in the pictures. We need to localize them, identify the category they belong to and know

the direction in which they are moving in order to provide the information needed for decision-making. The robots are split in two categories:

- *Targets* are robots to be herded; their top plate is red- or green-colored;
- *Threats* are robots carrying poles.

We plan on localizing robots using a Viola-Jones classifier. This classifier has shown in the past excellent real-time results in localizing faces and pedestrians [9], and can be extended to objects [2]. However false positives may be produced, especially if we try to detect more than one robot in a picture. In order to get rid of those outliers, a confirmation step is needed.

We plan on using basic shape information (e.g. circularity and ratio of principal axes) associated with color information (related to the presence of a top plate) to identify targets.

Threats will be identified using a conjunction of robot and pole shape information. The latter could be easily obtained using Hough transform [3].

In order to obtain the direction of a robot's movement, we will track it in several consecutive frames using a tracking algorithm. The CamShift algorithm [1] is a likely candidate.

Drone Localization and Arena Mapping

This module uses the grid landmarks given by the CV Module and the estimated pose of the drone given by the Odometry State Estimator to create a local map of the Arena and to locate the drone in this map. It can be divided in two submodules: local localization and mapping, and global localization.

The *local localization and mapping* uses the estimated state of the drone given by the state estimator and the grid landmarks obtained by the CV Module. It runs a Simultaneous Localization and Mapping (SLAM) algorithm to generate a local map of the arena composed by the grid landmarks and simultaneously locate the drone in this map. An EKF-SLAM (see [8]) algorithm in the 2D map has been explored as the solution of this part. Other SLAM algorithms that optimize graphs, like the one developed in [6], will be examined in order to improve the results obtained so far.

The *global localization* submodule utilizes the local map, the local drone pose, and the key lines given by the CV module, to locate the drone in the real Arena map. A particle filter including a model of the Arena will be used to achieve this task (see [8]).

The local map is not complete at the beginning of the mission, neither is the global positionning working. Hence the first task to be performed after the take-off and stabilization of the drone is to perform an exploration task aiming at finding a key line.

Target Ground Robots and Moving Obstacles Localization

In the case of the target ground robots and moving obstacles localization task the pose and location is estimated using the information given by the CV module in the form of local coordinates for each individual robot. With this information a local map of the position of the robots will be created to estimate their position at each instance and plan the "herding" task according to importance. To localize and track the robots a Particle Filter (PF) is implemented [8].

Navigation and Flight Control

The Flight Controller module (see Fig. 1) is in charge of navigating the vehicle, and has four operating modes which prepare the robotic system to execute the action commands sent by the Mission Planner.

- Navigation Mode: this mode uses the feedback from the Localization and Mapping modules. It is used to navigate at 2-3 m altitude to desired positions in the map. A obstacle-free trajectory is calculated by the trajectory planner.
- Fast Altitude Mode: the altitude is controlled by means of a fast predictive controller. The other degrees of freedom of the vehicle are commanded on open-loop to approximately stay in place. This mode is used when the vehicle is too near the floor where the sensing elements of the architecture might fail.
- Visual Servoing Modes: such operating modes are used for actions that require the interaction with an iRobot. These modes directly utilize the feedback from the Visual Tracking of the iRobot with which the aerial robot is going to interact. The estimated position of the MAV in the map is unused. However the System Monitor might decide to interrupt the action due to safety concerns, based on feedback from the rest of the architecture. Related modes:
  - Vertical Touching Mode: the MAV is commanded to touch an iRobot from above.
  - Ground-Level Touching Mode: the MAV is commanded to land in front of the iRobot.

In addition to the Flight Controller operating modes, the Multirotor Driver has several flying modes: take-off, hover, flying, land and emergency. These flying modes were designed to ease the realization of experiments during testing and also as safety measures.

System Supervisor
The system supervisor is in charge of the following tasks:

- During start-up: it does not allow the software architecture to start the flight until all modules are online. Each module broadcasts a ROS message stating whether the module is started (processing information) or not (idle). The supervisor checks this message to know if any module is not online, or frozen.
- During mission execution: it acts as an interface between the Human Machine Interface (HMI) and the rest of the architecture.
- During mission execution: it acts as an interface between the Mission Planner and the rest of the architecture. It ensures that changing the operating modes of the aerial robot is done in the correct step sequences. It also reports the execution status of the current action to the Mission Planner.
- During mission execution: it also monitors some safety measures regarding the rules of the competition. It checks that the vehicle does not move out of the map, and also that there are no obstacles within a predefined safety distance.

Mission Planning
Our architecture for the UAV includes a component for mission planning to generate tactical goals [5]. The mission planner generates long-term goals to achieve in the next 5 or 10 seconds. We have designed the mission planner to follow a strategy that we call *select-and-guide*. The UAV first *selects* an appropriate ground robot considering certain characteristics (distance between UAV and ground robots, distance to the green line, etc.). Then, the UAV *guides* the selected robot to the green line by interacting with the robot at specific moments. During the execution of this strategy, the process can fail because, for example, there are obstacles in the path. In this case, the UAV

may decide abandon the selected robot and select another one. The mission planner works as the deliberative component in our architecture following to the hybrid deliberative/reactive paradigm [7]The mission planner uses as input data the *observed world*, i.e., the information generated by the perception system. This includes, for every obstacle, its spatial location and, for every perceived ground robot, its spatial location, speed, orientation and time of cycle (time to the next end of cycle of 20 seconds). The mission planner generates *actions*, i.e., specific goals associated to behaviors that can be understood and achieved by the flight control system. Each reasoning cycle, the mission planner sends a specific action to the flight controller to be executed. The flight controller tries to execute the action and, then, notifies to the mission planner the *result*, i.e., either if the action has been completed or, on the contrary, it has failed due to certain reasons (presence of obstacles, etc.). Some of the actions the mission planner can generate are:

- *Touch(x,y,n).* The UAV touches *n* times the ground robot that is currently located at the point *(x,y)*.
- *GetClose(x, y).* The UAV gets close to the ground robot located at the point *(x, y)* up to a prefixed distance (e.g., 1 meter). This movement is limited to a maximum time of 5 seconds.
- *Watch.* The UAV goes to a position where it can perceive clearly a number of ground robots (e.g., 4 robots).

We have implemented the mission planner as a hierarchical planner. Figure 5 shows a partial view of the hierarchical decomposition of goals and sub-goals. The top level goal, *HerdRobots*, represents the general goal of the UAV, i.e., guide a set of robots to the green line. It can be performed by a set of simpler goals of type *Guide(x, y)*, i.e., guide a particular robot located at *(x, y)* to the green line. In its turn, this goal can be decomposed into specific actions (get close or touch) or other subgoals. Note that this can be a recursive process, i.e., at the bottom level there can be goals that are also present at upper levels in the hierarchy. For this decomposition process we use

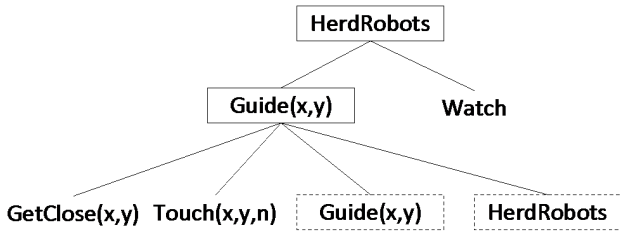


Figure 5: Example of hierarchy for goal-subgoal decomposition.

production rules. The rules include in their premises qualitative attributes. Table 1 shows attributes that we use for a ground robot. The values of these attributes are obtained as an abstraction process from the quantitative input data from the observed world. For example, the attribute *DistanceUAV* obtains the qualitative value *far* when the distance between the ground robot and the UAV is greater than 2 meters.

In order to select the most appropriate ground robot to be guided, the mission planner uses the function called *SelectRobot* which is minimized among feasible ground robots.

Flight Termination System

A kill switch derived from the one supplied by the IARC judges, can interrupt current to the motors by means of electronics. Its signal is sent via Xbee to the vehicle.

Attribute	Values	Description
Out	{*Yes, No*}	The ground robot is out of bounds of the arena when the UAV can reach the robot
ReachGreenLine	{*Yes, No*}	The ground robot will cross the green line by itself without any interaction with the UAV
NearObstacle	{*Yes, No*}	There is at least another robot (ground robot or obstacle) very close to the robot (distance less than 1 m)
DistanceUAV	{*Close, Far*}	Distance between the UAV and the ground robot (the distance is far when it is greater than 2 meters)
Direction	{*N, NE, E, SE, S, SW, W, NW*}	Estimated direction of the robot when the UAV reaches it (represented as cardinal points)
NearReverse	{*Yes, No*}	The robot is near the end of the cycle of 20 seconds (when it changes the direction 180 degrees)
FeasibleGuiding	{*Yes, No*}	There is enough available time to guide the robot, according to an estimation of the actions to be done

Table 1: QUALITATIVE ATTRIBUTES OF THE IROBOTS USED BY THE MISSION PLANNER.

PAYLOAD

Sensor Suite

GNC Sensors

The only sensor included in our architecture that can be considered specifically as a Guidance Navigation and Control (GNC) sensor is the PX4flow. The PX4Flow is an optical flow smart camera, with a native resolution of 752 x 480 pixels, which calculates optical flow on a 4x binned and cropped area at 250 Hz (bright, outdoors). In our system this sensor is used to provide measurements of the vehicles altitude and the horizontal navigation speed in body frame coordinates.

Mission Sensors: Target Identification and Threat Avoidance

To obtain information about the environment our vehicle uses five Computer Vision cameras: a down-facing camera, and four horizontal symmetrically distributed cameras. The selected cameras are IDS Imaging UI-1221LE cameras with the 752 x 480 pixels Aptina color WVGA sensor and global shutter.

As explained in the "Guidance, Navigation, and Control" subsection, these sensores will be used to: perform localization and mapping, detect the iRobots, to track the iRobots during interaction operations and to detect the obstacles.

In order to solve the main task in the competition which is to "herd" the mobile robots towards the green line, magnets are needed to change the direction of the robots heading. We have decided to use Rare-Earth Neodymium magnets. The disk shape of the magnet provides good field strength to weight ratio which is better for contact with the robots and manages to comply with our payload limitations. Once the robots heading is as desired, the aerial robot can be commanded to repeatedly land behind the iRobot to herd it as fast as possible towards the exit line.

Communications, and Risk Reduction RFI Solutions

A summary diagram of the communication interfaces present in our architecture is shown in Fig. 6. The Onboard Computer, an AscTec Mastermind, runs the software architecture developed by our

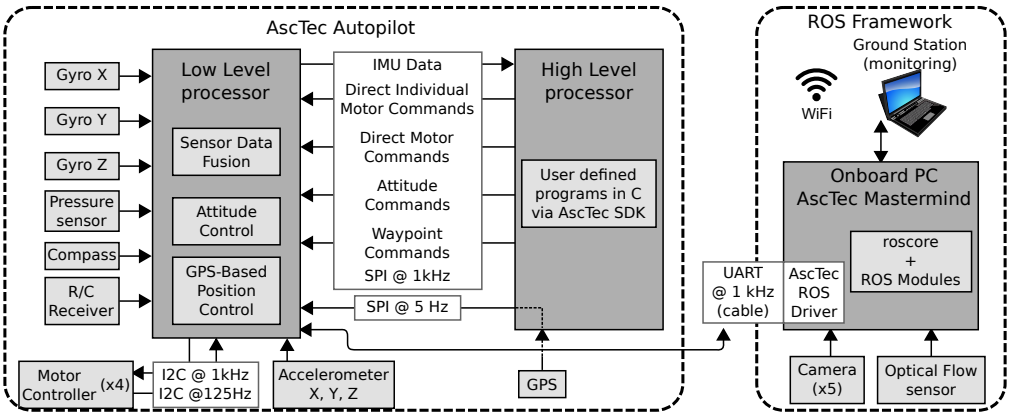


Figure 6: Communications diagram of the team's aerial robot architecture.

team. The communication with the ground station is achieved by means of a 5GHz wifi link. The ground station is only used for monitoring purposes and to start the mission execution. The low level attitude and motor controllers are managed by the AscTec Autopilot board. The Onboard Computer is connected to the Autoutpilot through a UART cabled connection.

OPERATIONS

Flight Preparations

Before any autonomous flight, the correct preparation of the flight guarantees the safety of the drone, the operators and the nearby people. These are the different preparations:

- Drone & Ground Station Preparation: the pre-flight is specified in table 2.
- Correct Mission Definition
- Arena Preparation: verify that there are no dangerous elements on the arena.
- Operators Preparation: both safety pilots and ground station operators must be concentrated and must know their tasks and responsibilities during the flight.
- External people (evaluators and gossips) Preparation: This people has to be notified that an autonomous flight is going to be carried out and that they are not allowed to leave the safety zone. A visual flag is shown from the Ground Station to notify the people that the autonomous mission is running.

Checklists

The check-list before an autonomous flight is shown in table 2.

Man/Machine Interface

During experimental mission executions the HMI is used mainly for monitoring purposes. It has a terminal console front-end and a visualization front-end based on ROS rviz. The terminal shows a summarized status of each module, the current action and its execution status, the battery levels and part of the telemetry data. The operator only interacts with the HMI to start the mission by pressing the start key.

PLACE	ACTION
Drone	Check the mechanical structure and parts of the vehicle
Drone	Check all onboard connections and connect battery
Ground Station	Ground Station initialized and working, perform battery check
Ground station	Check Ground Station to Aerial Robot communications
Drone	Safety RC controller, kill-switch and battery checks
Drone	Power the RC controller and the Autopilot and check
Drone	Power the onboard computer
Drone & Ground Station	Communication Drone - Ground Station up and working
Drone	Motors started and commands tested without taking-off
Ground station	Ground Station Human Machine Interface started and running
Drone & Ground Station	Onboard Autonomous Navigation Software started and running
Drone & Ground Station	Autonomous Mission can be started

Table 2: PRE-FLIGHT CHECK-LIST.

During testing, the HMI has a specifically defined set of keys for the current test at hand. It allows to substitute the Mission Planner by an operator, so that experimental tests can be focused on specific behaviours or parts of the architecture.

RISK REDUCTION AND SAFETY

The safety measures that work regardless of the state of the onboard computer are the following. First, the quadrotor has been equipped with propeller protection. Second, the landing gear has been equipped with padding as to protect the on-board sensors and equipment. And third, the safety pilot can take control of the vehicle at any time overriding the whole software architecture implemented in the onboard computer.

The figure 7 presents and analysis of the non-hardware safety features of our architecture.

Modeling and Simulation

The system is validated at different levels before the complete architecture is tested in experimental flights. This section explains the utilization of simulations designed to test the Mission Planner, the simulations designed to test the state estimation and control related modules. Additionally the Computer Vision algorithms are tested on images taken in a replica of the IARC competition flight field.

High-Level Simulation

We validated the mission planner with the help of a computer system that simulates the behavior of the UAV and the ground robots. This computer system is completely simulated, and substitutes all the modules of the architecture except for the Mission Planner. The three main functional blocks of this simulator are:

- *Scene simulator.* This part of the architecture simulates the movements of the ground robots, the obstacles and the UAV.
- *UAV control and perception.* The software architecture includes components that simulate the perception system and the flight control system of the UAV. Basically, this component

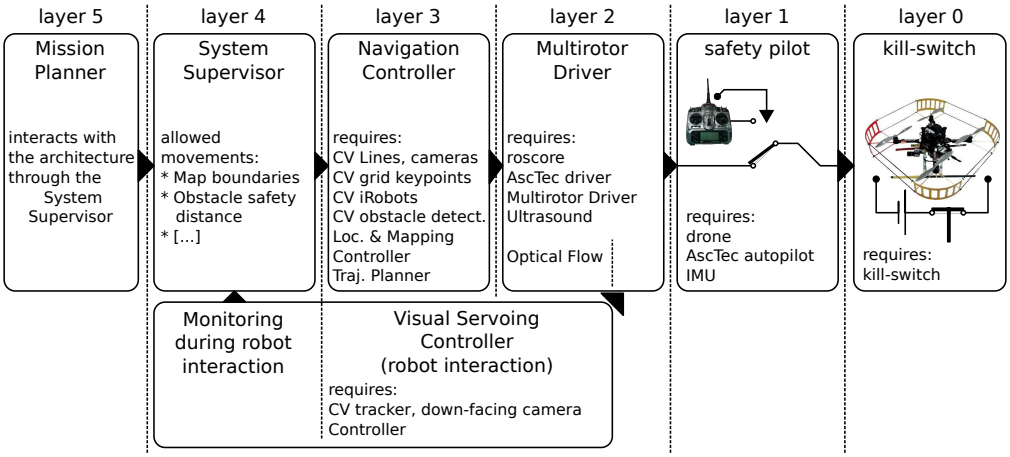


Figure 7: Safety measures and risk workflow diagram of the proposed architecture. The diagram divides the architecture in layers, to understand the functioning requirements and the risk workflow related to the various modules of the architecture. (layer 0) The kill-switch. (layer 1) The safety pilot can override the onboard computer commands. (layer 2) The flying modes provided by the multirotor driver. Also, the visual servoing controller does not use the ground optical flow measurements. (layer 3) The rest of the architecture uses the CV algorithms and the localization and mapping modules. (layer 4) The System Supervisor interfaces the Mission Planner.

Figure 8: (left) Visualization system of the High-Level Simulation testing (center & right) Pictures of the replica arena that has been prepared for the the experimental testing of our solution to IARC

receives as input an action from the mission planner (e.g., touch a particular ground robot) and generates as output a sequence of changes of direction and speed for the UAV.

- *3D Visualization.* The simulator also includes a component for 3D visualization. This component presents to the user a 3D animated view of the robots and the UAV. The visualization also shows the trajectories followed by the robots, which is useful for the user to analyze the movements. Figure 8 shows an example of the screen presented by the visualization tool.

Mid-Level Simulation

A ROS module that emulates the interface of the flying vehicle and its dynamics has been designed and implemented in order to run tests against the modules under development.

For many of the architecture's modules, the simulator can be run against the modules that will be run during experimental mission execution. This kind of tests can be run by the developer alone and allow to find bugs on the implementation without risking the actual aerial vehicle.

The main drawback of this simulator is the lack of a generation of relistic images that could

be processed by the Computer Vision algorithms. Thus, the CV modules must be tested offline separately on data gathered on our replica of the IARC competition field.

Testing - Field Tests

Field testing is necessary to ensure all components of the drone work well together in a real context. In order to prepare efficiently the drone for the IARC challenge, we have created an arena replica based on the informations given in the challenge rules. The arena consists of a grid over an inhomogeneous background. This background was made of various wallpapers showing different colors and patterns in order to provide harsh conditions for the vision-based algorithms. In addition, we possess five iRobots prepared according to the rules for simulating the mission. Figure 8 (center & right) shows our arena replica.

CONCLUSIONS

This paper presents the proposed solution and architecture developed by the team from the "Universidad Politecnica de Madrid" to the IARC2014 competition. The design of this architecture has presented several new challenges to our team with regards to our own prior work: it uses Computer Vision in a mostly unstructured environment, the controller requires to switch between various operating modes and to interact with ground robots, and the architecture includes a fully functional Mission Planner which has to take intelligent decisions.

The authors would like to thank the "Instituto Tecnológico 'La Marañosa' (ITM)" for the provided infrastructure and resources; the "Consejo Superior de Investigaciones Cientificas (CSIC)" of Spain for the JAE-Predoctoral scholarships of two of the authors; and the Spanish Ministry of Science MICYT DPI2010-20751-C02-01 for project funding.

References

[1] Gary R. Bradski. Computer Vision Face Tracking For Use in a Perceptual User Interface. *Intel Technology Journal*, 1998.

[2] L. Dlagnekov. License plate detection using adaboost. Technical report, Computer Science and Engineering Department, San Diego, 2004.

[3] Richard O. Duda and Peter E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Communications of the Association for Computing Machinery*, 15:11–15, January 1972.

[4] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 1981.

[5] Hui-Min Huang, editor. *Autonomy levels for unmanned systems (ALFUS) framework*, volume I: Terminology. NIST Special Publication 1011-I, 2008. Contributed by the Ad Hoc Autonomy Levels for Unmanned Systems Working Group Participants.

[6] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. G2o: A general framework for graph optimization. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3607–3613, May 2011.

[7] Robin Murphy. *Introduction to AI robotics*. MIT press, 2000.

[8] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[9] P. Viola and J. Michael. Robust real-time object detection. In *International Journal of Computer Vision*, 2001.