

.....

Received 9 December 2010; accepted 11 June 2011

Journal of Field Robotics 28(5), 667–689 (2011) © 2011 Wiley Periodicals, Inc.
View this article online at wileyonlinelibrary.com • DOI: 10.1002/rob.20403

applications are weed detection with high-resolution requirements and hydric stress determination using lower resolution.

Typically, maps are built by stitching a set of georeferenced images through mosaicking procedures. The high-precision positioning of the UAVs and the capability of performing hovering while delivering high-resolution cameras (i.e., color or multispectral cameras) make them the most suitable choice for performing mosaics. Nevertheless, the most relevant drawback of this solution is the limited autonomy of the mini aerial vehicles. This limitation bounds the area that can be covered by an individual robot. Consequently, in order to cope with the typical size of crop fields, either multiple flights or a group of robots is required.

In this paper, a feasible solution for performing aerial imaging applied to PA by using a fleet of UAVs is presented. This paper provides a unified view of the modules and the field work that we have been carrying out in multirobot task planning, path planning, and UAV control.

1.1. The Approach Proposed and Paper Outline

The system provides a decoupled solution, because the methodology employed can be easily divided into two steps. Figure 1 is the global system flowchart. Initially, the target workspace is defined in the crop field by using georeferenced points. It is subsequently provided to the area-partitioning step. The resulting subareas are discretized according to the required spatial resolution. Path planning is then executed for each subarea. Finally, individual flight plans are sent to the quadrotors in order to be performed. The main issues of the proposed solution are as follows:

- **Task subdivision and allocation.** This is a classical problem in multirobot coordination. Given a global task T_0 and R robots, the problem is how to partition T_0 into R nonoverlapping subtasks and how to assign subtasks to the robots for their execution. As a contribution of this

work, the proposed tool simultaneously provides solutions for the two mentioned problems in a distributed way applied to UAV restrictions: each robot is aware of its own characteristics and status but does not know anything about those of its teammates. In this approach, the original task T_0 is the area to be surveyed, which is divided into subareas through a negotiation process in which each robot claims covering as much area as possible. This solution is reviewed more deeply in Section 2.

- **Path planning.** Once the subareas are assigned, each robot is in charge of accomplishing its covering mission by following a path. This path typically consists of a set of waypoints that have to be computed. A waypoint is a position where a sample (i.e., an image) has to be taken. To solve this problem, coverage path planning (CPP) techniques are used. CPP is a subfield of motion planning that addresses the problem of determining the complete coverage path for a robot in the free workspace. CPP algorithms for UAVs are few. In this work, a practical implementation was successfully carried out. The path planning issue is studied in detail in Section 3.
- **Aerial vehicles' control.** Although accuracy in controlling the position and velocity under wind disturbances usually is the most demanded feature of flight control systems (FCS) for UAVs' outdoor operation, attitude stabilization is essential, especially in quadrotors. Along these lines, a new control law called backstepping+FST controller has been developed in prior work. It considers all the nonlinearities of the system (i.e., Coriolis, gyroscopic forces, etc.), improving on the attitude stabilization in comparison to classic proportional-integral-derivative (PID) algorithms. The proposed algorithm introduces a desired angular acceleration function within the control law that takes into consideration the effects of the angular rate variation and their impacts in terms of maneuverability (abrupt angular rotation at high speeds) and disturbance rejection (fast control response). Control performance has

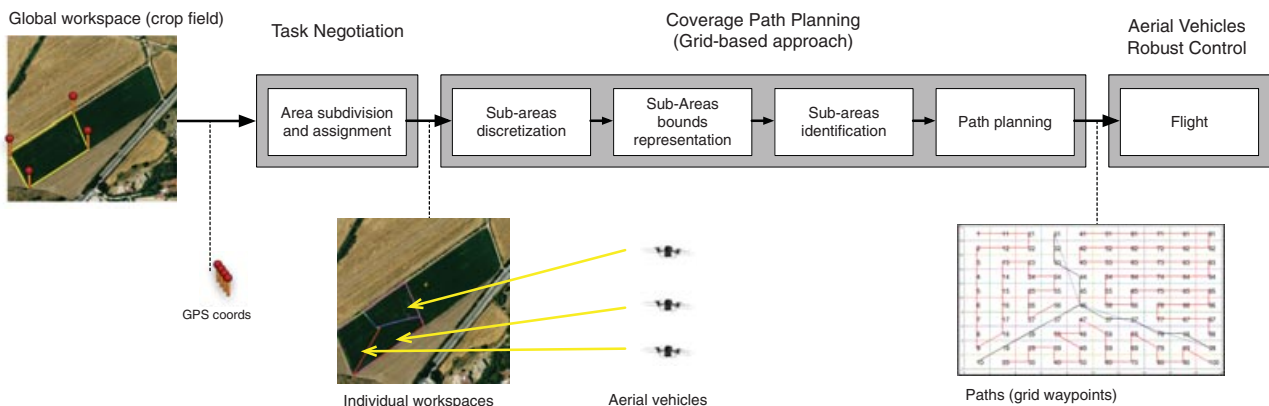


Figure 1. Overall system flowchart.

turned out to be excellent, and it is simple enough to be embedded into the onboard computer of the Hummingbird quadrotor used. The flight control solution is discussed in Section 4 and Appendix A.

After overviewing the motivation and main contributions of the paper, the rest of this section is devoted to reviewing the literature related to each of the fields previously mentioned. Then Sections 2–4 describe in detail the main technological issues that compose the overall system. Section 5 summarizes the experiments performed during test campaigns on real crop fields, providing a unified vision of the three subproblems and their application to the context of remote sensing in precision agriculture. Finally, Section 6 discusses the results and experience gained.

1.2. Related Work

Multi-UAV systems is a relatively young field of research (Ollero & Maza, 2007). Owing to their complexity, the majority of the works reviewed concentrate on one topic, either planning (Gancet, Hattenberger, Alami, & Lacroix, 2005a, 2005b; Lacroix, Alami, Lemaire, Hattenberger, & Gancet, 2007; Lemaire, Alami, & Lacroix, 2004), perception (Tisdale, Ryan, Kim, Tornqvist, & Hedrick, 2008), or navigation (Chen, Dawson, Salah, & Burg, 2006), it being difficult to find a paper that provides a unified view from the theoretical foundations to practical experimentation. This paper is intended to provide such a view, highlighting the need for integration of different knowledge areas. In this section, the most relevant works that are related to the topics involved in this paper are briefly summarized.

1.2.1. Related Work in Task Subdivision and Allocation

Cooperative applications can be roughly divided into two classes (Parker, 2003): tight cooperation requires a continuous coordination between the robots, and loose cooperation requires coordination at the beginning of the mission for planning a division of labor, or when replanning is required. This work falls in the second class. Market-based (Dias & Stentz, 2003) and auction (Gerkey & Mataric, 2002) techniques are commonly used in this class of problems. One of the most popular protocols for task assignment is the *Contract Net* (Smith, 1980), and many algorithms (Botelho & Alami, 1999; Dias & Stentz, 2003; Gerkey & Mataric, 2002; Golfarelli, Maio, & Rizzi, 1997; Stentz & Dias, 1999) have been proposed based on this protocol. In all market and auction-based approaches, each robot has cost and revenue functions that are used to compute the expected gains and losses for performing (sub)tasks. In this work, an instance of the *planning and task decomposition problem* is at hand. This is usually approached in one of two ways: *allocate-then-decompose* and *decompose-then-allocate*. As pointed out by Dias, Zlot, Kalra, and Stentz (2006), “by decoupling the decomposition and allocation problems, these

approaches do not consider the complete solution space and may find highly inefficient solutions.” In the works of Zlot and Stentz (2005, 2006), the trading of whole task decomposition trees is proposed in order to solve the two problems simultaneously. Auctions can be initiated by any agent to (re)allocate subtasks at any time, and the task is partitioned into subtasks by the auctioneer agent. In this way, their approach achieves a dynamic allocation that improves the solution at any reallocation. A drawback of this approach is that an auctioneer must propose a complete partition of the task; thus the task partitioning step is actually centralized.

Another limitation of such approaches is that robot preferences are considered only in the assignment stage, when the robots decide whether to accept (or opt for) a task. A task partitioning algorithm not taking this into account may produce solutions that are not feasible, leading to an incomplete task execution, because no agent will bid for certain task(s). Such features do not suit the needs of an approach that should consider robot capabilities already at the task partitioning stage. This feature is important especially when dealing with heterogeneous vehicles, in either mobility and/or equipment, as in the case of the work described in this paper.

Our negotiation protocol performs a simultaneous task subdivision and allocation in a distributed way, taking into account robot preferences and without the need for exploring whole task decomposition trees in order to find a good allocation.

Negotiations have been widely studied in the context of socioeconomic studies using, among others, game theory (Osborne & Rubinstein, 1994). The main problem with game theory approaches is that the theoretical results obtained refer to simplified models that are not immediately applicable to complex applications. The protocol we propose is based on Rubinstein’s alternate-offers protocol (Rubinstein, 1983). Because such a protocol is based on a unidimensional good, a search mechanism for the best (counter)offer had to be devised for the protocol to be applied in real multidimensional tasks.

1.2.2. Related Work in Multicoverage Path Planning

The problem of finding a robot path that covers a given workspace in an optimal way has been extensively studied by several authors. The schemes proposed to solve the CPP problem are in general oriented to online CPP (Choi, Lee, Baek, & Oh, 2009; Choset, Acar, Rizzi, & Luntz, 2000; Oh, Choi, Park, & Zheng, 2004; Wong & MacDonald, 2003) or offline CPP (Moon & Shim, 2009; Oksanen & Visala, 2009; Weiss-Cohen, Sirotin, & Rave, 2008). Online approaches are usually sensor-based navigation schemes in which the robot has to cover an unknown environment. The typical offline approaches are those in which the robot has to cover an area whose map is available to the path planner. The map gives representation of the area boundaries and the

obstacles therein. The two schemes have also been simultaneously presented, where the robot has an a priori knowledge of the environment but also relies on obstacle avoidance behavior (Luo & Yang, 2008).

The environment is in general mapped in a cellular decomposition or grid, following the taxonomy proposed by Choset (2001). Cellular decomposition approaches always give an exact representation of the robot workspace, where the robot can cover cells with back-and-forth motions. On the other hand, grid-based approaches give an approximated representation of the workspace. The cellular decomposition approaches normally take root from the trapezoidal decomposition approach proposed by Latombe (1991). In addition to the Boustrophedon cellular decomposition, an improved Boustrophedon cellular decomposition with critical points can be found, in which cells are defined by using critical points of Morse functions, bioinspired (e.g., ant colony, genetic algorithms), and also by topological navigation with natural landmarks (Choset et al., 2000; Wong & MacDonald, 2003). Grid-based solutions are probably the most common, and the methods employed fall into spanning trees, neural networks, genetic algorithms, and general heuristic searches (Choi et al., 2009; Oh et al., 2004; Weiss-Cohen et al., 2008).

The majority of the works reviewed were developed for ground robots, and their extension to other types of vehicles is not discussed. Additionally, most of them have presented only simulation results with little experimental testing under real platforms. The requirements for aerial vehicle coverage applied to remote sensing in a farmland environment are different from those for ground robots. First, an aerial vehicle almost always is able to move in any direction without damaging the crop and, depending on the flight altitude and spatial resolution required, to deal with wider areas and therefore augmented cell dimensions. Furthermore, because not all regions are suitable for take-off or landing with aerial robots, the trajectory must ensure starting and ending points in places that fulfill all the requirements, such as safety margins, sufficient space for operation, pick up and drop ability, and accessibility.

Maza and Ollero (2007) present a polygonal area decomposition applied to inspection with a team of aerial robots. The area is divided into subareas by using a sweep-line approach, and then the subareas are assigned to the robots based on their relative capabilities. Each individual robot computes the waypoints needed to perform a back-and-forth pattern with a minimum number of turns. If a robot turns out to be inactive, the algorithm is computed again. However, the solution proposed considers just convex areas without obstacles. Moreover, such an approach is mainly focused on the robot assignment problem rather than the CPP problem.

In Moon and Shim (2009), a study of two algorithms to address area coverage with UAVs for crop dusting purposes is presented. The first algorithm is denoted as a grid

point-based algorithm and the second a modified Boustrophedon algorithm. The first one proposes an approach to discretize an area through points. Then a procedure that selects points inside the sampled set is employed to obtain a coverage trajectory. The resulting area coverage path is reproduced as a spiral. The approaches proposed by the authors are mainly dedicated to area decomposition and sampling. Although the first approach presents a simple way to sample an area, the solution provided to compute the coverage trajectory is not subject to any constraints. Moreover, the trajectory obtained is reproduced as a spiral from outside to inside, which can be a problem in large areas if the UAV runs out of fuel. The second approach is based on a well-known exact cell composition method that employs simple back-and-forth motions to cover the decomposed subareas. In any case, the provided results are referred only to simulations. Finally, the authors also mention the use of multirobot systems; however no results were provided.

Jiao, Wang, Chen, and Li (2010) also report aerial CPP. The authors propose an exact cell decomposition method to break down a polygonal area into subareas by employing a recursive greedy algorithm. Each subarea is covered with back-and-forth motions along the vertical direction of each convex subarea span. The shortest path through the cells to be covered is determined through an undirected graph, in order to reduce the number of turns. This work does not consider obstacles, and it is assumed that the aerial vehicle flies just over a convex polygon area. Additionally, is not clear what type of aerial vehicle is intended to be used in this approach. The proposed method was tested only in simulation.

Therefore, the analysis of the literature has shown that multirobot CPP using UAVs is not mature yet, and contributions to enhance vehicle autonomy, efficiency, and robustness are required. In fact, no reports have shown that it had been put into practice in real agricultural scenarios. Therefore, according to this necessity, this work proposes that a grid-based approach, due to an exact cell decomposition [such as a trapezoidal decomposition (Latombe 1991)], is inefficient for aerial coverage, mainly because the dimensions of the samples acquired have to be homogeneous. It should also be highlighted that the proposed approach deals with both regular and irregular polygonal shapes, including nonconvex shapes. An effort has been made to minimize the number of turns that the aerial robot has to perform so as to decrease the time required to perform the mission and energy consumption. Finally, this approach also ensures nonoverlapped paths with predefined takeoff and landing positions.

1.2.3. Related Work in Aerial Vehicle Control

Within the technological challenges of precision agriculture, the most important of concern to flight controllers are precision and robustness (Oetomo, Billingsley, &

Reid, 2009). UAVs were studied in the context of agricultural applications in Schmale, Dingus, and Reinholtz (2008). However, most of them include consumer-off-the-shelf basic linear position controllers (e.g., PID) that lack high-position accuracy when the vehicle maneuvers at high speeds including in strong wind disturbances (Bouaballah, 2007).

In this work, the aerial vehicle control goal is to apply a novel controller that ensures robustness and reliability within the framework of crop field aerial sampling. Issues such as maneuvering at speeds up to 30 km/h including wind-speed disturbances about 10 m/s are experimentally analyzed. A novel nonlinear control methodology, initially developed in prior work (Colorado, 2010; Colorado, Barrientos, Martinez, Lafaverge, & Valente, 2010), is now deployed into the onboard computer and applied to a new quadrotor model. As a first attempt, an integral backstepping methodology was implemented to achieve attitude stabilization (Olfati-Saber, 2001). However, after several control tests, reliable attitude stabilization was not achieved when the quadrotor speeded up. Thus, we integrated a new term within the original backstepping control law called the *desired angular acceleration function* that considers the effects of angular rates and accelerations. A brief introduction of this controller, called backstepping+FST, is presented in Section 4, and experimental results of precise vineyard area coverage are consigned to Section 5.

2. TASK SUBDIVISION AND ALLOCATION

2.1. Problem Statement and Assumptions

In this work the objective is to partition an area into sub-areas.¹ This is an instance of a more general task partitioning problem that we have generalized in prior work. We refer the interested reader to Rossi, Aldama, and Barrientos (2009) for a detailed explanation.

In the context of aerial surveying, we consider as task the object to be divided, i.e., the area, and we are interested in partitioning a given target area and assigning sub-areas to the agents. A task $T^{(x)}$, where x is a k -dimensional vector of parameters, has to be divided into R subtasks: $T^{(x)} = \{T_1, \dots, T_R\}$, R being the number of agents. Each subtask T_i , $i = 1..R$, can be described by a set of parameters x_i , $T^{(x)} = \{T_1^{(x_1)}, \dots, T_R^{(x_R)}\}$.

In general, a good subdivision is such that there is minimum overlapping between subtasks (ideally null) and such that the union of the subtasks covers the original task.

That is,

$$T_i \cap T_j = \emptyset, \quad \forall i, j = 1 \dots R \quad \text{and} \quad \bigcup_{i=1}^R T_i = T, \quad (1)$$

where the operators $\cap$, $\cup$ are to be defined according to the meaning of the task. In the case of areas, their geometric meaning is straightforward.

2.2. Evaluation of a Task

During the negotiation, each robot has to evaluate the cost and reward of a given task. To this aim, it has to take into account its internal parameters to evaluate the cost of executing the task, the start-up cost (for instance, to reach the execution site), specific constraints (e.g., forbidden zones, turn angles, sensors), and the general reward associated with the task, expressed as a reward function g . Given the complete task T_0 , the evaluation function g_i of subtask S_i for agent i takes the form of a weighted sum of terms such as dimension of the task (e.g., area, length, number of targets), distance from the initial position of the robot to the task execution starting point, and penalties for overlapping subtasks and for the part of S_i outside T_0 (Table I). The last two factors are important as they balance the importance of avoiding overlapping and not going outside mission limits. In this application, we used the following formulation, where α , β , γ , ϕ are mission-dependent parameters:

$$g_i(S_i) = \alpha \cdot \dim(S_i) - \beta \cdot \dim(S_i \setminus T_0) - \gamma \cdot \sum_{j \neq i} g(S_i \cap S_j) - \phi \cdot \text{dist}(\text{pos}_i, \text{site}_i). \quad (2)$$

2.3. Task Negotiation

A given task T_0 can be executed by a team of R robots after performing a suitable subdivision of the task and assigning the subtasks to the robots. Our aim is to perform these two actions simultaneously and in a distributed way. In our system, the number of subtasks is determined by the number of robots willing to participate in the negotiation.

Table I. Factors taken into account when evaluating a (sub)task, which can change according to the specific application.

Factor	Description
$\dim(S_i)$	Area of the task
$\text{dist}(\text{pos}_i, \text{site}_i)$	Distance to task initial point
$\dim(S_i \setminus T_0)$	Area of the part of the task exceeding the target task
$\forall_{j \neq i}: \dim(S_i \cap S_j)$	Tasks overlapping

¹In many cases crop fields have a rectangular or trapezoidal shape. However, more irregular shapes can be found, e.g., in vineyards, one of the target cultivations of our work. This motivates the need for providing a general algorithm that can work for *any* area shape and not be limited to simple and regular shapes.

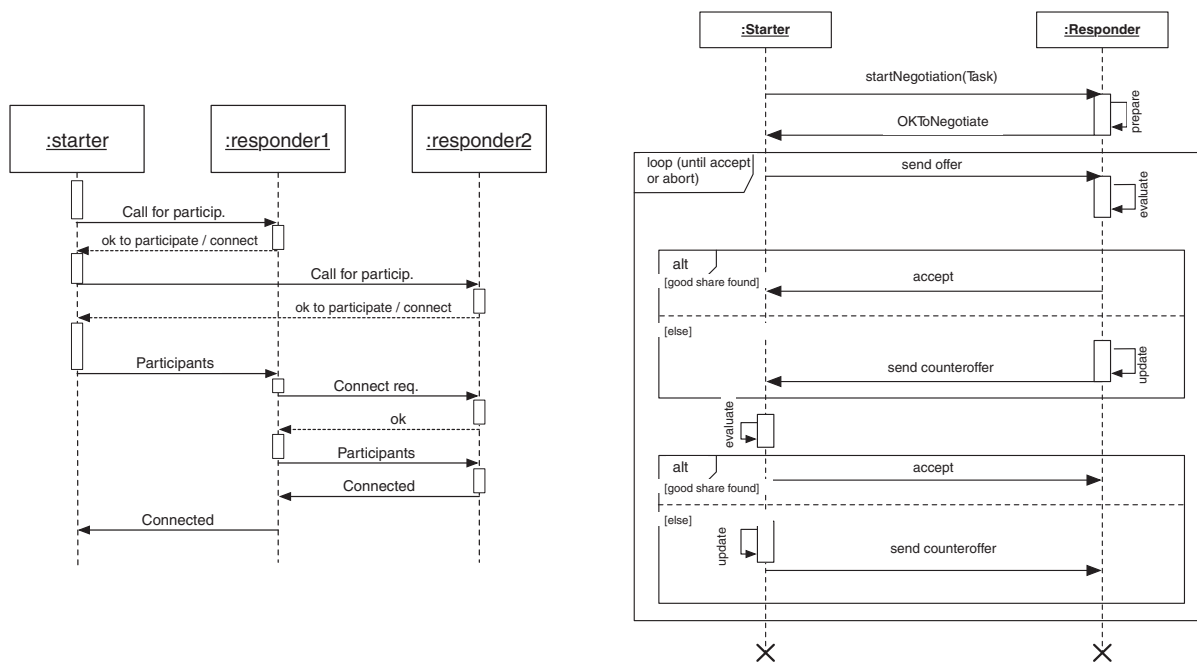


Figure 2. Connection protocol (left) and negotiation loop (right). The negotiation loop can be interrupted at any moment due to an abort message of one part or a timeout when waiting for an offer (not shown in the sequence diagram for clarity).

We assume that the robots are willing to perform as much as they can of the given task (hence maximizing their reward), the only limitations being their available resources (endurance, computation power, battery consumption, etc.). Thus, in a negotiation, each agent will try to maximize its reward by (i) trying to get a subtask as big as possible and (ii) minimizing overlapping with other agents' tasks.

The negotiation protocol is an extension of the *alternate-offers* protocol proposed by Rubinstein (1983). The advantage of such a protocol with respect to the contract net protocol is that in the contract net the auctioneer agent has to split the task into subtasks before placing the subtasks in the market, whereas in Rubinstein's protocol the splitting is done during the negotiation. Additionally, each agent needs just to know whether the others agree with the share proposed for itself and nothing else.

In the *alternate-offers* protocol, each agent starts proposing the biggest possible share for itself and reduces it until the counterpart finds it acceptable. In this way a good near-optimal solution, although not optimum in general, can be achieved. The responder can agree with a subdivision, or disagree with it, and in this case it has to propose a counteroffer. The protocol assumes a negotiation cost called *discount factor*, by which it decreases its offers at each round, starting by claiming the whole good. Such a protocol has interesting theoretical properties: it guarantees a termination and can forecast the final agreement, which will be a perfect equilibrium in the sense of game theory. An extension of the protocol has been necessary because it is not imme-

diately applicable to the multidimensional case. A detailed description of the process is outside the scope of this paper. We refer the reader to Rossi et al. (2009) for a more complete explanation.

Figure 2 (left) shows the connection protocol. Agents are invited by the *starter* agent in turn, and the list of agents willing to participate is sent, in turn, to all participating agents in order to establish direct connections. In case of a lost connection during the connection stage and during negotiation, timeouts are used to interrupt the process. In case an interruption occurs, the protocol is restarted from scratch. Figure 2 (right) shows the communication protocol. Only two agents are depicted for clarity. The *starter* agent starts the negotiation, sending a request to the teammates (only one is depicted). In case the *responder* accepts to participate, a bargain loop is entered where the two parties alternately receive an offer, evaluate it, and decide whether to accept it, and in this case formulate a counteroffer. Note the search step performed at each new offer received and generated. The loop can be interrupted in any moment if one of the parties decides to unilaterally abort the negotiation in case a timeout occurs while waiting for a message or in case the mission is interrupted by the command and control station. When an agreement has been reached, the result is a subdivision of the original task and at the same time an assignment of the subtasks.

When there are three or more agents, the protocol can be extended in negotiation rounds in which, in turn, each agent claims a part of the task. After each round, if all the

agents are satisfied the negotiation ends; otherwise another round is started. To avoid biasing the process in favor of the first agents, a new starter is chosen at each new round, in rotation. In this case, there is no theoretical result that guarantees convergence of the negotiation. However, numerical simulations with up to seven agents have demonstrated that an agreement is always found. Current work on this topic is devoted to a characterization of the convergence time as a function of the number of agents.

2.4. Communication and Computational Cost Issues

A final remark must be made regarding the communication topology implied by the algorithm. In this work, we have assumed that all-to-all communications are available in order for the negotiation to take place. All-to-all communications are not strictly required, because in the negotiation rounds it is sufficient to establish a communication ring. Communications shall be available only during the negotiation itself and are not needed during the execution of the tasks the robots have agreed upon.

Bandwidth restrictions are not significant, as the amount of information exchanged is not a critical issue for our protocol. As an example, a polygon of 12 points is described by 24 floating point numbers. Assuming 4 bytes per number, a total of 96 bytes (plus headers) are sent at each offer. Taking into account the number of agents (three to five) and the typical number of negotiation rounds needed (30–50), a total amount on the order of kilobytes for the whole process can be estimated.

Computational costs are centered on the proposal evaluation and counterproposal update steps. Such costs depend on the application at hand. In this case each evaluation consists of computing polygon intersections and area calculations. A whole negotiation takes the order of seconds. In any case, we underline that in a distributed approach such as the one we propose, each agent performs such computation only for its share, whereas in approaches that need to compute a complete solution, this cost must be multiplied by the number of robots of the team or by the number of subtasks that form the complete task.

3. PATH PLANNING

3.1. Problem Statement and Assumptions

The multi-CPP problem is formalized by assuming that there is a top-level procedure that handles the area division and the robot assignments. A cost-efficient multirobot CPP should result in a coverage path for every robot, such that the union of all paths is equivalent to the overall workspace coverage and the total coverage time to completion is minimized.

Let us consider an area $A \subset \mathbb{R}^2$ decomposed in a finite set of regular cells $C = \{c_1, \dots, c_n\}$ such that

$$A \approx \bigcup_{c_i \in C} c_i. \quad (3)$$

Let S be a finite set of subareas and L a finite set of line segments that divide A . Therefore $A = S \cup L$, where

$$S = \bigcup_{s_i \in S} s_i \quad \text{and} \quad L = \bigcup_{l_i \in L} l_i. \quad (4)$$

In an aerial robotic-based coverage mission, the following constraints must be considered subject to the vehicle characteristics:

- N_t : Number of turns, i.e., number of times the vehicle rotates around the z axis (yaw angle movements)
- N_r : Number of times the vehicle covers a previously covered cell (revisited cells)
- t : Coverage time to completion of a single subarea s
- $T = \sum_{t_i \in T} t_i$: Coverage time to completion of the areas S

Let $t_i^* = \min\{t_i\}$ be the optimal time for covering area s_i . Then the optimal time to completion of S , T^* is achieved when all subarea coverage takes the minimum time:

$$T^* \Leftrightarrow t^* \forall s \in S. \quad (5)$$

Let us consider a coverage path P to be optimal if it implies the minimum number of turns N_t and of revisited cells N_r . As explained later in this section, an algorithm that guarantees that no cell is visited twice has been adopted, i.e., that $N_r = 0$. Thus, $P^* \Leftrightarrow \min\{N_t\}$. Clearly, t_i^* is achieved with the minimum number of turns. Consequently, $t_i^* \Leftrightarrow P^*$. Then

$$T^* \Leftrightarrow P^* \forall s \in S, \quad (6)$$

3.2. Area Sampling

The workspace is split through an approximate cellular decomposition. Following the taxonomy proposed by Choset (2001), the workspace is sampled like a regular grid. This grid-based representation with optimal dispersion is obtained by dividing the space into cubes and placing a point in the center of each cube. Therefore it can be defined as a kind of Sukharev grid.

The centroid of each cell is considered a waypoint, and the cell base size (i.e., width and height) matches the size of the image to be acquired. It should be pointed out that each aerial robot has to fly at a certain constant height in order to ensure a specific resolution and the image sensor field of view (FOV) has to be enough to cover each cell (see Figure 3). For this reason, the flight altitude has to be set depending on the camera carried by the robot, with the positive side effect of reducing the danger of collisions between vehicles. Although an approximate cell composition has the

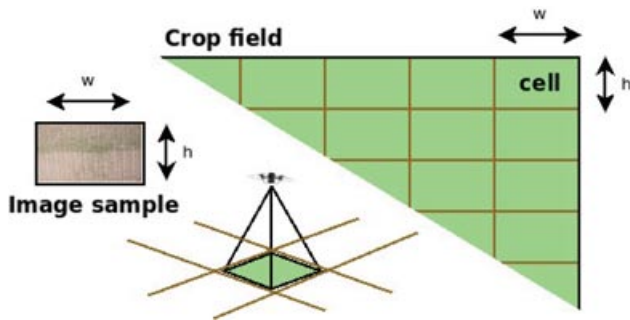


Figure 3. Area sampling schematic. The required resolution and the FOV of the cameras determine cell size.

drawback that in general border cells may be partially occupied cells that must be visited, an exact cell decomposition such as the trapezoidal one is inefficient for aerial sampling because we are interested in acquiring an equally sized set of images.

3.2.1. Border Representation

Because the area partitioning algorithm works in a continuous space, borders (line segments) that define the subareas in the grid-based environment must be recomputed. In the grid environment, we represent lines using the Bresenham's line algorithm (BLA) (Bresenham, 1965), which belongs to the family of line drawing algorithms used in computer graphics. The procedure to approximate line segments in discrete graphical data structures such as rectangular grids of pixels is denoted as *rasterization*, whose goal is to find the best approximation of a line segment given the inherent limitations of a raster environment, considering a set of constraints to be optimized (e.g., pixel proximity with the ideal line, continuity, and uniformity) (see Figure 4).

3.2.2. Subarea Identification

The procedure to individuate the subareas in the grid-based environment is a recursive flood-fill algorithm. The algorithm picks an empty cell (i.e., a cell not marked as occupied) and floods in four direction while there are empty cells, and each cell flooded is marked as occupied. When

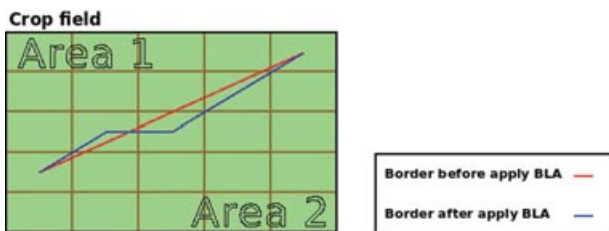


Figure 4. Illustration of the solution based on rasterization.

the flood cannot go further, the algorithm is restarted. This procedure is repeated until all nodes of the grid are marked as occupied.

3.3. Coverage Path Planning

The CPP algorithm is performed for each subarea. We have developed an algorithm based on a wavefront planner (LaValle, 2006) that works by propagating a wave front from the goal cell through all free grid cells bypassing all obstacles. A distance transform is applied over the grid by employing a breadth-first search (BFS) on the graph induced by the neighborhood adjacency of cells. Hence, the coverage path can be easily found from any starting point within the environment to the goal cell by choosing the nearest neighbor cell in gradient ascendant order, opposite to the conventional method.

The variables of interest in CPP are N_t and N_r (see Section 3.1): a good path has a minimum number of turns and avoids visiting previously covered cells. Aerial robots have a certain endurance at a constant velocity, which is considered a constraint (not a cost). Therefore, given an area, the cost to minimize is the coverage time, which fully depends on the number of turns and the revisited cells.

Because the number of cells in the grid-based workspace is known in advance, a simple way to ensure coverage completeness without revisited nodes is to use the deep-limited search (DLS), limiting the search depth to the maximum number of vertexes. During the gradient tracking, the algorithm can find more than one neighbor to choose from, with the same potential weight. A backtracking algorithm builds a tree with all coverage path candidates and retrieves the one with the minimum number of turns.

4. AERIAL VEHICLE CONTROL

The last issue to be faced after the area has been partitioned and assigned and paths have been computed is the control of the vehicles.

Although nonlinear controllers have been used for some time for controlling quadrotors, poor analysis has been conducted on specifically improving attitude control while the aircraft is maneuvering at moderate speeds and performing aggressive changes in orientation. To improve on this, we have adopted the Frenet-Serret (FS) formulation as a set point function of the aircraft angular velocity and acceleration. A complete and detailed description of the six-dimensional algebra operators, notation, and basic concepts of rigid body dynamics applied to quadrotors modeling is outside the scope of this paper. The reader is referred to Section 3.2 of Colorado (2010)² and previous work

²Available for download at http://oa.upm.es/3493/2/TESIS_MASTER_JULIAN_COLORADO_B.pdf.

in Colorado et al. (2010) for a better understanding about backstepping+FST derivation.

In summary, the backstepping+FST method focuses on controlling the torque components of τ_{cm} : roll τ_ϕ , pitch τ_θ , and yaw τ_ψ . This section briefly shows roll-control derivation for τ_ϕ . The first step is to define the roll tracking error e_1 , and its dynamics (derivative with respect to time):

$$\begin{aligned} e_1 &= \phi^d - \phi, \\ \dot{e}_1 &= \dot{\phi}^d - \omega_x. \end{aligned} \quad (7)$$

A Lyapunov function³ positive definite, $V(e_1) = e_1^2/2$, is used for stabilizing the tracking error e_1 . The virtual control law for stabilizing the angular speed tracking error is then defined as

$$\omega_x^d = c_1 e_1 + \dot{\phi}^d + \lambda_1 \int e_1, \quad (8)$$

with $e_2 = \omega_x^d - \omega_x$. Including e_2 in Eq. (8) and differentiating with respect to time (note that $\omega_x = \dot{\phi}$),

$$\dot{e}_2 = c_1 \dot{e}_1 + \ddot{\phi}^d + \lambda_1 e_1 + \ddot{\phi}. \quad (9)$$

Replacing $\dot{e}_1 = -c_1 e_1 - \lambda_1 \int e_1 + e_2$ in Eq. (9) and with $J_{x-cm,T}$ the scalar component of the x -axis moment of inertia of the vehicle,

$$\dot{e}_2 = c_1 \left(-c_1 e_1 - \lambda_1 \int e_1 + e_2 \right) + \ddot{\phi}^d + \lambda_1 e_1 + J_{x-cm,T}^{-1} u_\phi. \quad (10)$$

Note in Eq. (10) that $u_\phi = \tau_\phi$. Solving u_ϕ using $\dot{e}_2 = -e_1 - \lambda_2 e_2$, the control law is

$$\begin{aligned} u_\phi &= J_{x-cm,T} \left[e_1 (c_1^2 - \lambda_1 - 1) + e_2 (-c_1 - \lambda_2) \right. \\ &\quad \left. + c_1 \lambda_1 \int e_1 - \ddot{\phi}^d \right]. \end{aligned} \quad (11)$$

Likewise, for pitch and yaw stabilization,

$$\begin{aligned} u_\theta &= J_{y-cm,T} \left[e_3 (c_2^2 - \lambda_3 - 1) \right. \\ &\quad \left. + e_4 (-c_2 - \lambda_4) + c_2 \lambda_3 \int e_3 - \ddot{\theta}^d \right], \\ u_\psi &= J_{z-cm,T} \left[e_5 (c_3^2 - \lambda_5 - 1) \right. \\ &\quad \left. + e_6 (-c_3 - \lambda_6) + c_3 \lambda_5 \int e_5 - \ddot{\psi}^d \right]. \end{aligned} \quad (12)$$

Finally the terms $\ddot{\phi}^d$, $\ddot{\theta}^d$, and $\ddot{\psi}^d$ in Eqs. (11) and (12), respectively, are replaced by the desired angular acceleration functions obtained using the FS formulation in Eqs. (13). The term $\omega_{\{i\}}^{\{R\}}$ is a $\mathbb{R}^{3 \times 3}$ resultant rotation matrix that describes the vehicle's trajectory orientation in terms of

the FS frame transformation (see Colorado et al., 2010, for more details):

$$\begin{aligned} \ddot{\phi}^d &= \text{atan2}(\omega_{\{i\}23}^{\{R\}}, \omega_{\{i\}33}^{\{R\}}), \\ \ddot{\theta}^d &= \text{atan2} \left(\omega_{\{i\}13}^{\{R\}}, \sqrt{(\omega_{\{i\}23}^{\{R\}})^2 + (\omega_{\{i\}33}^{\{R\}})^2} \right), \\ \ddot{\psi}^d &= \text{atan2}(\omega_{\{i\}12}^{\{R\}}, \omega_{\{i\}11}^{\{R\}}). \end{aligned} \quad (13)$$

Figure 5 shows the modeling and control workflow.⁴ The system-control module is composed of a cascade-mode configuration of altitude, position, and attitude controllers coded onboard the vehicles. Appendix A shows a comparison between backstepping+FST and a PID control.

The next section details experimental results conducted on both Hummingbirds and AR100 quadrotors. The analyzed data will show that our assumption of improving tracking speed is achieved by the introduction of the angular backstepping control law. Those functions consider the effects of the vehicle linear speed and acceleration, thus ensuring accurate and fast response to abrupt angular rate change, making attitude and position control more reliable when the quadrotor is maneuvering at high speeds (including strong wind disturbances).

5. EXPERIMENTAL RESULTS

In this section, the results of missions performed in a vineyard farming site using the aforementioned system are presented. The goal of the experiments is twofold:

1. To provide a useful field report in terms of using this system as a practical approach for aerial sampling in precision agriculture involving a complex scenario (vineyard). Because vineyards are normally located in hilly regions, their shape can be quite irregular. In this case, area partitioning might not be trivial, and a sophisticated area partitioning algorithm like the one we are adopting provides a useful tool.
2. To demonstrate the simplicity and robustness of the system architecture in terms of flight control and accuracy of crop field sampling.

First, we illustrate three different examples of the results of the task partitioning and path planning steps on vineyards, in order to provide numerical results and illustrate their outcome on different instances. The fields are located in Belmonte de Tajo, southeast of Madrid, Spain.⁵ Figure 6 shows the three vineyards and the results of the

³Lyapunov stability analysis of the backstepping+FST controller is detailed in the cited work (Colorado, 2010, Section 4.3.5).

⁴Aerodynamics, dynamics modeling, and backstepping+FST control derivation can be found in Colorado et al. (2010) and Colorado (2010) with more detail.

⁵The authors gratefully acknowledge the courtesy of Bodegas Andres Morate (<http://www.andresmorate.com/>) for providing their vineyards for experimentation.

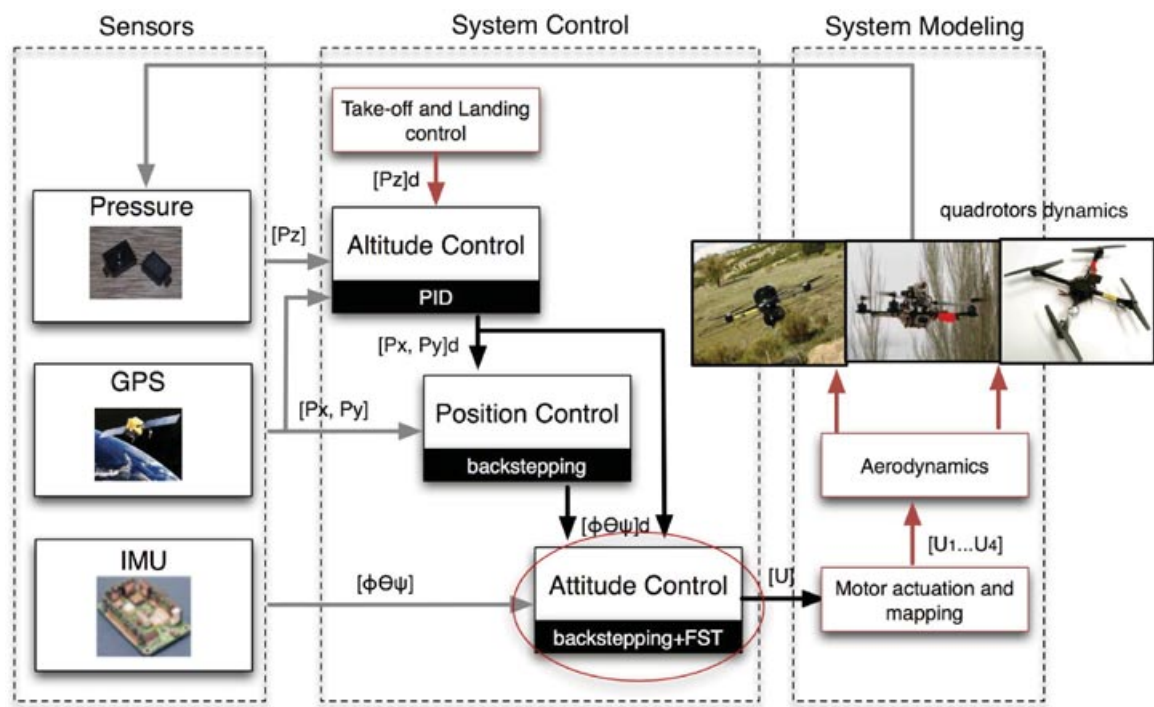


Figure 5. Modeling and control workflow.

area partitioning (left) and the corresponding paths generated (right). Table II summarizes the results in terms of negotiation round taken, elapsed time for the path planning step, number of turns, and quality of service obtained (see below for a definition).

The vineyard-field chosen for the field tests was Field 1. Its location is depicted in Figure 7, with the area to be covered delimited by a thick line. This crop field scenario has an approximate size of 327 × 195 m.

To evaluate the experimental results according to the goals previously mentioned, we introduce a set of metric variables. These metrics are conceived in order to analyze the performance of the vehicles *during* and *after* the mission. Tables III and IV summarize these metrics: T_M is the total mission time, including deployment and setup time; T_L is the setup time; T_C is the total flight time; T_E is the time the vehicles are actually flying between waypoints; $T_{\bar{E}}$ is the time the UAVs spent hovering over a waypoint; and P is the length of the coverage path.

Table II. Summary of the results for task partitioning and path planning.

Field	Negotiation rounds	Total path planning time (s)	Total no. of turns	Quality of service (%)
1	36	0.19	37	72
2	43	0.22	42	81
3	49	0.31	36	80

Table III. Metrics *during* mission flight.

Variable	Description
P_E	Position error tracking
H_E	Altitude error tracking
V_W	Wind disturbance

Table IV. Metrics *after* mission flight.

Variable	Description
T_M	Mission time of completion
T_L	Grounded time
T_C	Coverage time of completion
T_E	Effective time of coverage
$T_{\bar{E}}$	Noneffective time of coverage
P	Coverage path

Finally, because subarea borders will not be sampled, an end user-oriented metric called the *quality of service* index is measured, which indicates the percentage of the field actually surveyed. An upper bound (U_B) of unvisited cells can be estimated for the security strips as follows. For simplicity, let the area be discretized in an $N \times M$ grid, $N < M$. The maximum number of border lines is equal to

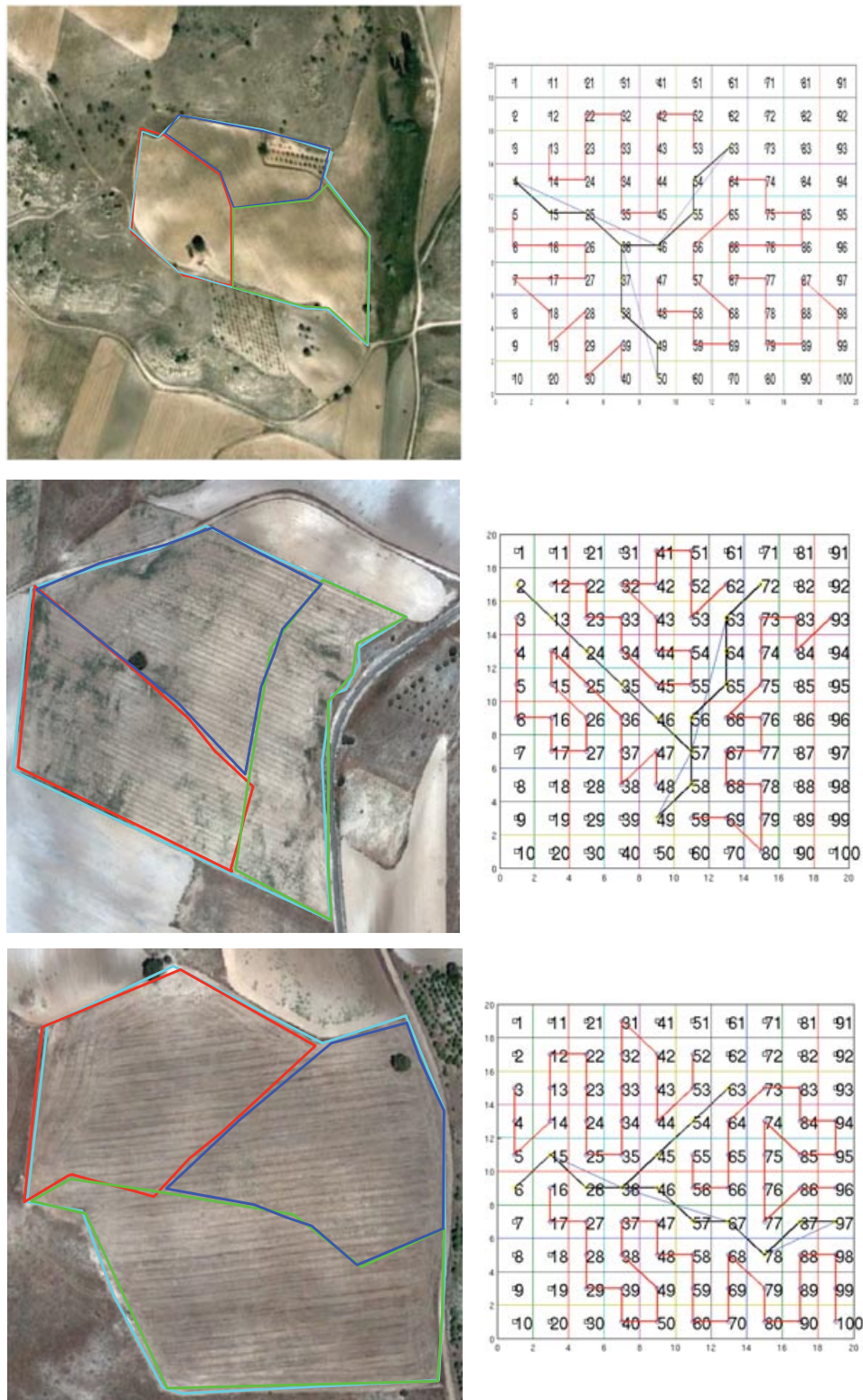


Figure 6. Area subdivision and multi-CPP on different vineyard parcels. Left: Orthophoto illustrating the subdivided area. Right: Discretization and area coverage paths.



Figure 7. The vineyard crop field object of the field tests. Its location is 40°11'24.68"N, 3°28'56.09"W.



Figure 8. The aerial vehicles flying over the agricultural scenarios.

the number of robots R . The maximum length of a line is equal to the diagonal. In a discrete world, the length in cells of the diagonal is equal to M ; hence,

$$U_B = \frac{R \cdot M}{N \cdot M} = \frac{R}{N}. \quad (14)$$

We refer to the value $U_B/(N \cdot M)(\%)$ as a *quality of service* index. In our experiments, $N = M = 10$; $R = 3$, and a *maximum* of 30% of the cells will be security zone and therefore will not be visited (actual values range from 20% to 30%; see Table II). Note that in crop field sampling, 100% sampling is not needed. Conventional sampling methods currently used are based on far lower sampling rates.

5.1. Platform Overview

Experiments were carried out using a team composed of three quadrotors: (i) two Hummingbird quadrotor platforms⁶ and (ii) the AR100 platform.⁷ Prior to the description of the experimental setup, this section briefly

introduces some description of the platforms employed. Both platforms are shown in Figure 8.

The mainframe of the Hummingbird is composed of an AutoPilot board that includes (i) a LPC2146 ARM high-level (HL) processor, (ii) a triple-axial compass for measuring the vehicle's heading, (iii) an inertial measurement unit (IMU) and pressure sensor from attitude and altitude measurements, (iv) a GPS unit for geolocalization, and (v) low-level X-BLDC brushless motor controllers. Using the Software Development Kit (SDK) provided by Asctech, our backstepping+FST runs onboard the HL processor using a sample rate of 30 samples per second. The GPS position loop is set at 1 Hz, whereas the IMU loop is set at 100 Hz (the maximum transfer rate of the IMU data can be set to 1 kHz using the serial interface). All data generated by the IMU are sent to the ground PC station via wireless data link (XBee 2.4-GHz module) for data analyses and user visualization purposes.

The AR100 also provides a three-axis gyroscope, three-axis accelerometer, three-axis magnetometer, GPS, and a barometer. Moreover, it has a servo mounted in the mainframe, which enables a commercial zoom digital camera to be tilted up to 100 deg. This instrumentation ensures

⁶Ascending Technologies GmbH, <http://www.asctec.de/>.

⁷AirRobot GmbH & Co. <http://www.airrobot.de/>.

Table V. Aerial fleet parameters.

	AR100	Hummingbird
Diameter (m)	1.0	0.63
Weight (kg)	1.32	0.6
Payload (kg)	0.25	0.2
Autonomy (min)	20	25
Wind load (m/s)	8	8
Max. velocity (km/h)	50	50

platform stabilization, manual control, and waypoint navigation, as well the dispatch of the commands from a software mission planner set in the ground station, which enables the operation of the robot at a higher level.

A summary of the technical data of the two platforms appears in Table V.

The *ground station* is composed of the communication devices and a graphical user interface (GUI) for real-time information of the mission. For the AR-100 quadrotor, a graphical interface was developed to fulfill the needs of a friendly-user experience. On the other hand, the Hummingbird already provides a GUI that enables some basic functionalities for mission supervision. Figure 9 shows the base station for the experimental setup and the GUI of both quadrotor platforms (left for the AR-100, right for both Hummibirds).

In terms of communication architecture (see Figure 10), the base station computer receives and send data to AR100 through a RS-232 downlink and uplink that operates between 2.3 and 2.5 GHz. The quadrotor can also be teleoperated through a radio control (RC) system that relies on nine available channels and operates at 35 MHz. Additionally, a video transmission downlink at 688 MHz is also set to receive video images acquired by the digital camera. For the

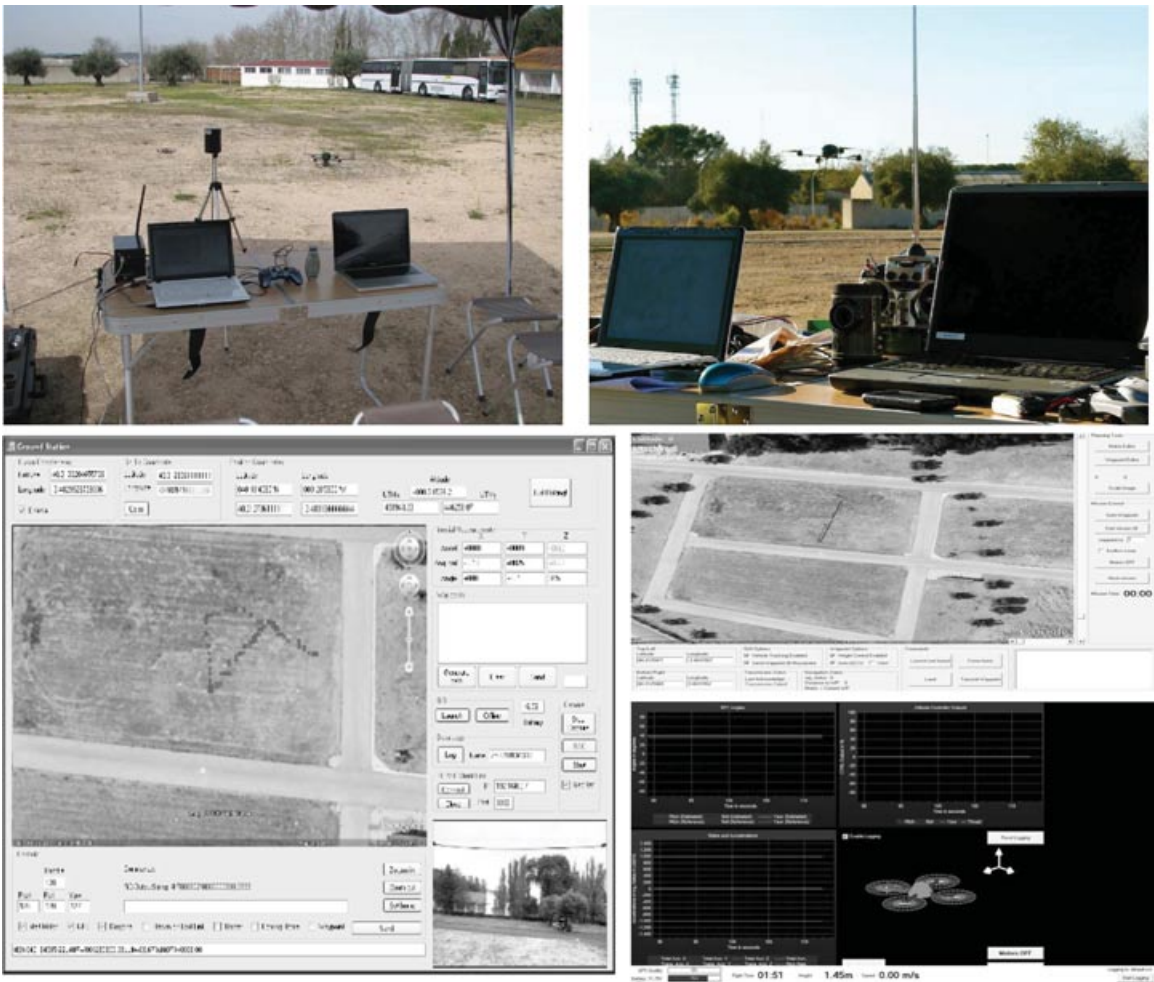


Figure 9. Ground-PC station and the GUI provided to the user.

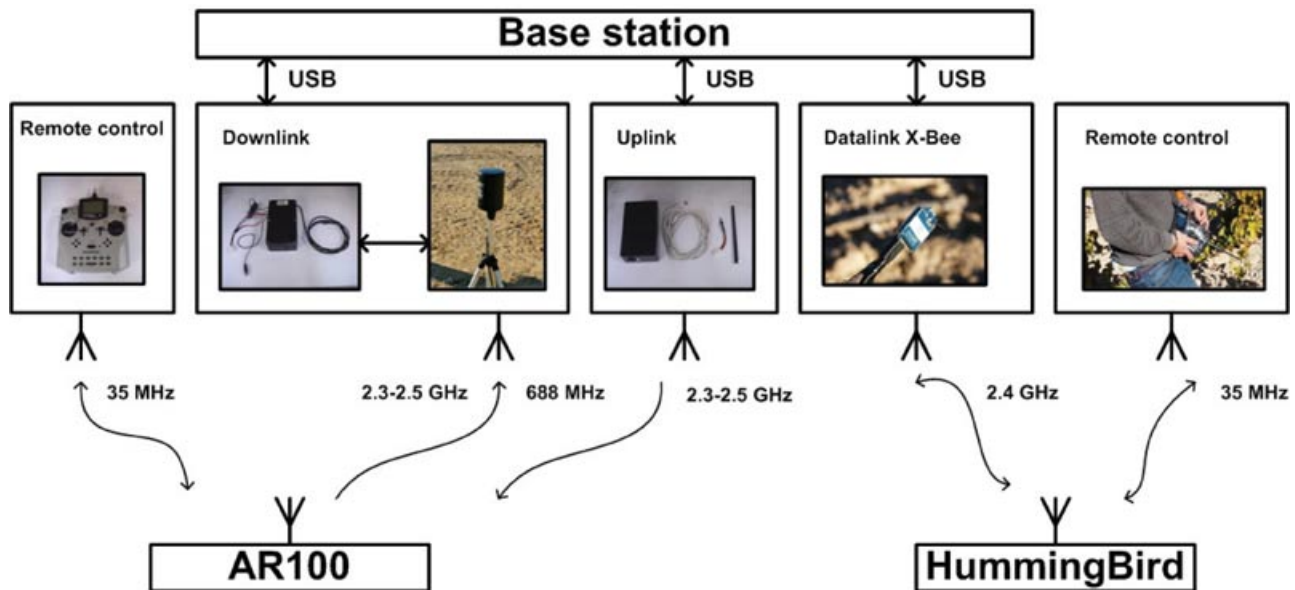


Figure 10. Air-ground communications architecture.

Hummingbird, the vehicle sends and receives data through the base station through a X-Bee Datalink that operates at 2.4 GHz, as previously mentioned. In terms of user supervision it is important to highlight that the mission can be manually aborted using the GUI. For this reason, two procedures can be followed: (i) the quadrotor has to remain in hovering maneuver waiting for manual flight or (ii) the quadrotor will return to the home point (beginning of the mission) autonomously.

5.2. Experimental Setup

Recalling the workflow of Figure 1, the steps followed in order to performed a survey mission are as follows:

1. The operator defines the target mission field by selecting the area's vertices on a georeferenced image.
2. The vertices are passed to the negotiation algorithm, which produces the area subdivision and assignment (at this stage, the characteristics of the robots employed such as endurance and the cameras' FOV are hard coded in the workflow).
3. The subareas are discretized and then the areas' bounds are computed, taking into account the required resolution and camera parameters.
4. Cells containing obstacles (e.g., water well, irrigation systems) and subareas bounding cells are marked as no-fly zone.
5. Coverage paths for each subarea are calculated.
6. Flight plans (list of waypoints) are sent to the robots.

7. Mission is simulated (optional step using a Matlab-based simulator depicted in Figure 11).
8. Mission starts.

Given the limited sensing and processing capabilities of the robots employed, direct perception for avoiding collisions during flight is not feasible. On the other hand, communication of the positions is not feasible because vehicles do not have direct communication during the mission, and indirect communication through a base station is not recommended due to a high risk of radio interference.

Thus, with simplicity and effectiveness in mind, collision avoidance is achieved through two basic principles:

- Different vehicles fly at different altitudes.
- Subarea bounds define a security zone: vehicles are not allowed to enter it.

5.3. The Vineyard Field Experiments

5.3.1. Experimental Setup and Preflight Phases

Experiments were performed in a vineyard field, characterized by its irregular layout and its variable altitude profile. The target of this experiment was to verify the system under PA common conditions, so adverse situations were set up: rough terrain irregularities with different slopes, uneven perimeter, and different wind conditions could be found depending on the subarea and the altitude considered. It should be also highlighted that the perimeter configures a nonconvex area, so as to validate the approach presented.

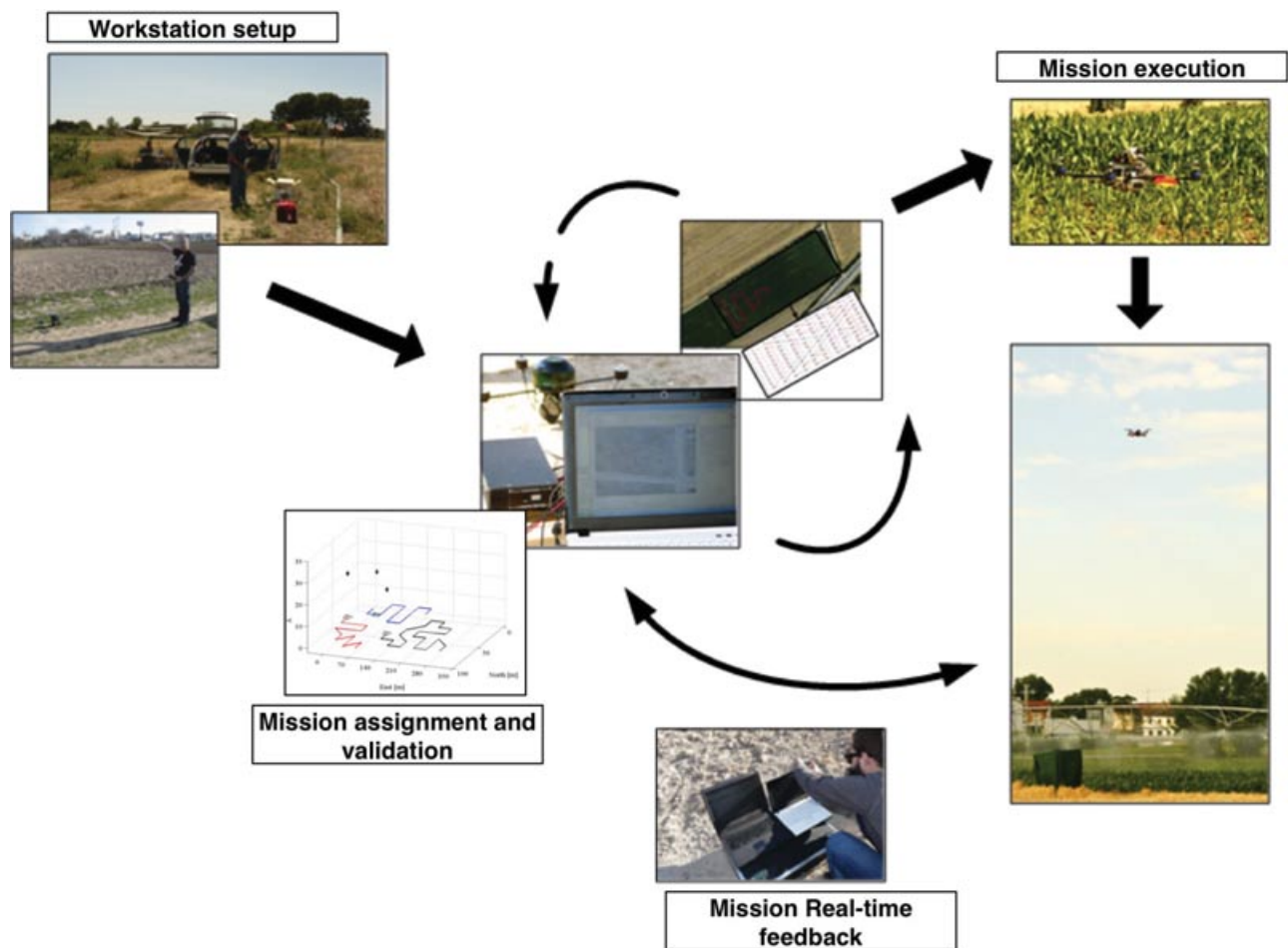


Figure 11. Steps performed for a real mission.

This complex environment was inscribed in a rectangle [see Figure 12(a)]. Therefore, the workspace for this experiment was defined as a 63,765-m² rectangular area (see Figure 7), where general parameters are set up as follows:

- Field size: 330 × 200 m
- Grid size: 10 × 10 cells
- Cell size: 32.7 × 20 m

According to the workflow, the first step in the process is to divide the workspace into as many adjacent areas as quadrotors available. Figure 6 (top, left) shows the three subareas resulting from this step. Next, flight plans are generated [Figure 6 (top, right)]. Table VI summarizes the results of this step. When a mission has been simulated and the parameters for the CPP have been validated [Figure 12(b)], configuration data are sent to the corresponding drones. The strings containing the waypoints for each mission are loaded into the scheduler of the drones, waiting for the mission start signal.

Table VI. Details of the results obtained from multi-CPP on the vineyard parcel chosen for the field tests.

	Area 1	Area 2	Area 3
Turns	9	8	20
Computation time (s)	0.15	0.02	0.02

Note that the actual target workspace consisted of 65 of the 10 × 10 cells of the area, 12 of which would not be sampled because they formed the subarea boundaries [cf. Figure 6 (top)]. Thus, the *quality of service* in this experiment has been 72% (18% of the area was not sampled).

5.3.2. Flight Results

During the experiments, flight parameters were monitored and logged. Results allowed us to compare not only the performance of each individual drone, but also the global behavior of the system when facing hard conditions.

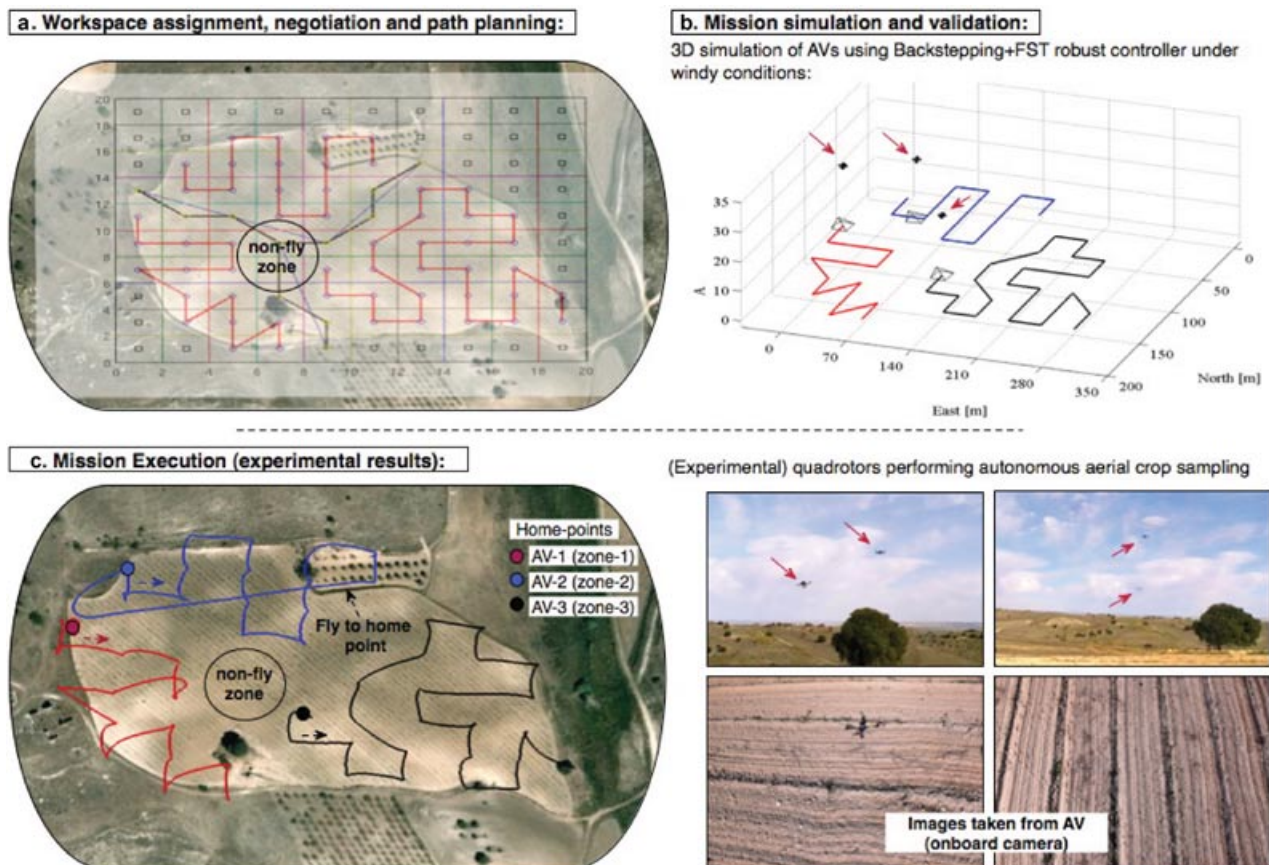


Figure 12. Detailed experimental workflow results of three quadrotors performing vineyard field sampling and coverage. Labels AV-1 and AV-2 correspond to the Hummingbird UAVs, AV-3 to the AR100 UAV.

Figure 12(c) shows the results of one test mission flight. In the following, we analyze the outcome of this test.

The main requirement that aerial vehicles must accomplish for a reliable imaging survey is to have an accurate three-dimensional (3D) positioning and be able to perform smooth trajectories. This is important because the vehicles have to acquire a set of georeferenced images that will be postprocessed by an image mosaicking procedure in order to build a map from the crop field surface. Owing to its relevance on this mission, the first parameter analyzed is altitude. Altitude accuracy plays an important role in imaging surveys, because the image samples must be taken from the ground at a determinate constant altitude. The relationship between the camera FOV and the aerial vehicle height above the ground is given by $\tau_d/\tau_h = l_d/l$, where τ_d , τ_h , l_d , and l stand, respectively, for cell dimension, aerial vehicle height, image dimension, and focal length of the camera.

In Figure 13 (right) the altitude evolution for each quadrotor during the flight is shown. On the left, the corresponding Cartesian paths are plotted. As expected, the flat-

ter surface provides better results (see plot for quadrotor 1, Figure 13, right, top). Altitude control also has a direct effect on battery evolution, reducing peaks and maintaining a more stable discharging process. The mean altitude had a maximum error of 3.35%, as shown in Table VII, which is acceptable for an outdoor mission in which the platforms are constantly subject to prevailing winds.

Errors in altitude location are not directly related to accuracy in navigation over the XY Cartesian plane. Figure 14 shows the results obtained in position tracking. Compared to the paths shown in Figure 14 (bottom), where real paths are superimposed on the theoretical paths on workspace assignment, maximum errors are related to sudden variations in the quadrotor's yaw axis (orientation). The best example comes from the tracking error on zone 1. In this flight, note an error increase of almost six times in four moments ($t_A = 0.05$ min, $t_B = 0.65$ min, $t_C = 1.2$ min, $t_D = 1.4$ min) when the quadrotor faces acute angles. This is also confirmed by the relation between the number of turns performed and the average position error. The subarea with more turns (zone 3), and consequently with

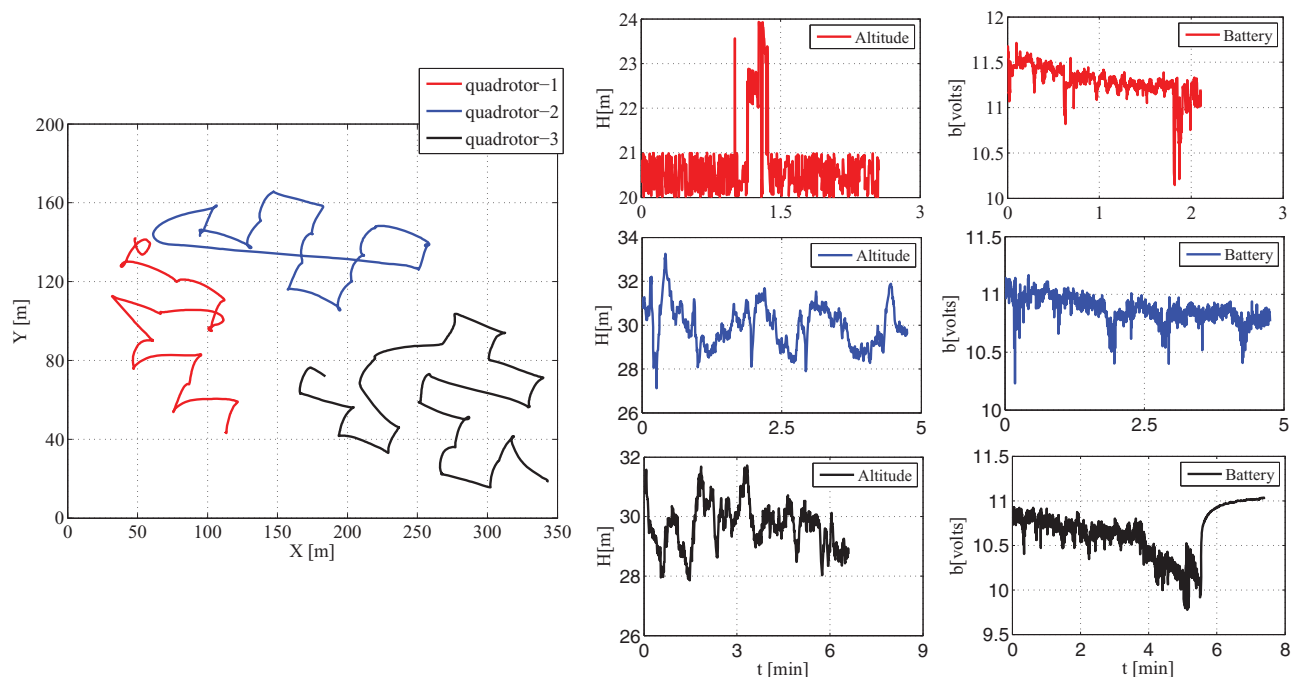


Figure 13. (Experimental) Cartesian path (left) and altitude tracking and power consumption during flight (right).

more yaw drifts, presents a position error greater than the area with fewer turns (zone 2) (see Table VII). Table VII also permits a first comparison between our backstepping+FST controller and the commercial PID-based controller of the AR100 UAV. The P_E error for UAVs 1 and 2, adopting our controller, is smaller than the error for UAV 3. UAVs 1 and 3 have a similar error, but note that UAV 1 was flying at a higher speed. This comparison is done on different paths and different areas. Appendix A illustrates a more fair comparison between the two controllers under equal conditions.

As far as flight speed accuracy is concerned, preliminary tests have been done through simulations, with the purpose of helping the design and tuning process of the AV controllers. During the simulations the vehicles were subject to wind speeds of 10 m/s in both the x axis and the y axis. The vehicle dynamics during the simulations performed well. However, real performance was not as good as expected. The reason for this behavior is the winding streams the vehicles are exposed to. Such winds are typical in hilly zones and can achieve peaks much greater than 10 m/s.⁸ For the purpose of our mission this drawback can affect a system in two ways: when a vehicle is hovering over a waypoint, because the drone must maintain its

Table VII. Summary of the metrics measured during the flights; flight took place on March 21, 2011, 15:20 hours, Spanish time.

UAV no.	h_{ref} (m)	h (m)	h_E (%)	V_{ref} (m/s)	V (m/s)	V_w^a (m/s)	P_E^b (%)
1	20	20.67	3.35	5	3.22	13	2.3
2	30	28.95	3.50	5	2.97	13	1.5
3	30	28.69	4.30	5	2.41	13	2.5

^aWind speed measured at base station.

^bError accurately measured using Vincenty (1975) formula.

position to provide a clean and accurate image, and when flying between waypoints, because the vehicle shall perform a smooth trajectory in order to optimize flight time and power consumption.

Figure 15 shows a comparison between the simulated and measured velocity profiles generated by the controllers during the test mission.

Another important detail in a coverage mission is the overall time of completion. Overall completion time can be decomposed into coverage time and setup time. Moreover, coverage time can be decomposed into effective and non-effective time.

The coverage time is the neat time devoted to the purpose of the mission. However, to accomplish such a mission, there is an often underestimated and seldom

⁸On one occasion, a strong wind stream lifted one vehicle to >100-m altitude. The vehicle eventually landed some 500 m away from the base station.

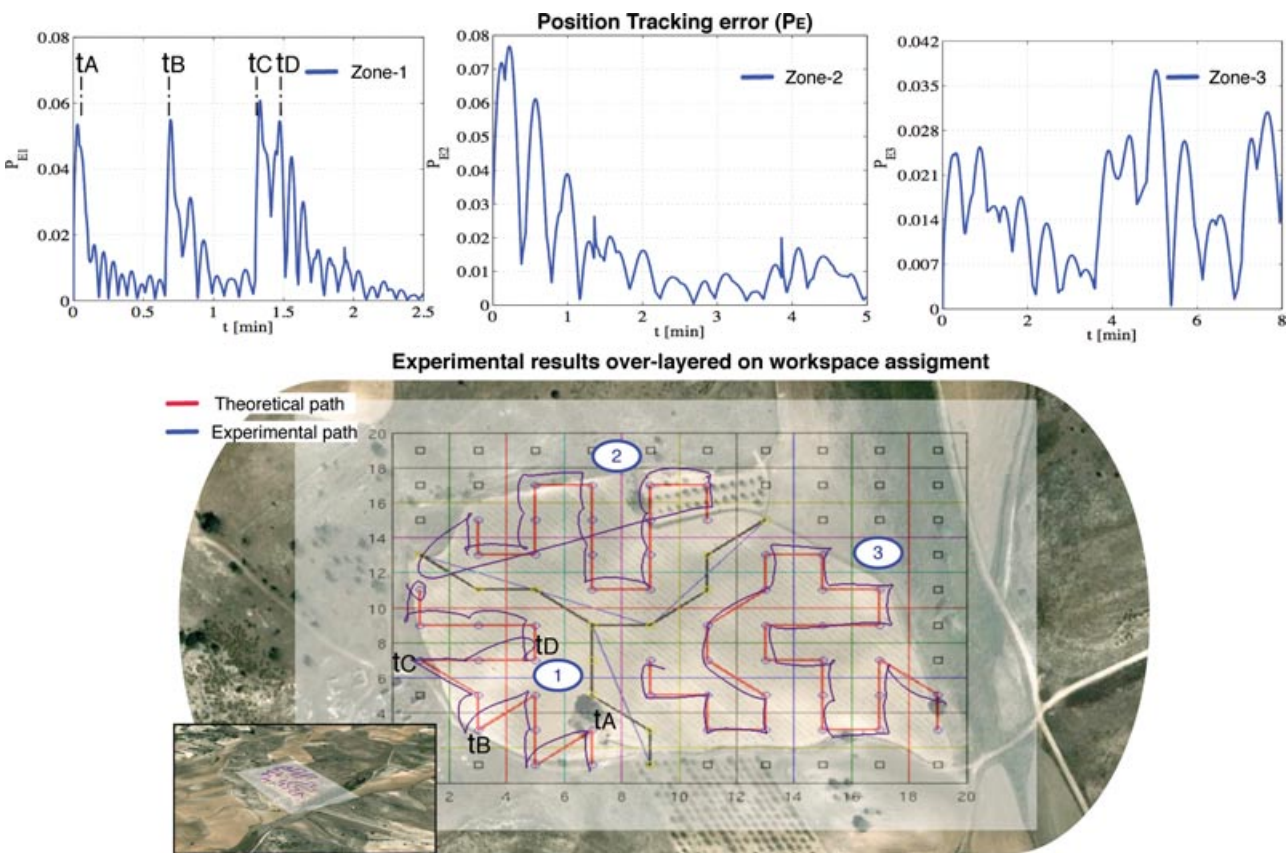


Figure 14. Position tracking during flights. Labels t_A , t_B , t_C , and t_D refer to peaks of the tracking error.

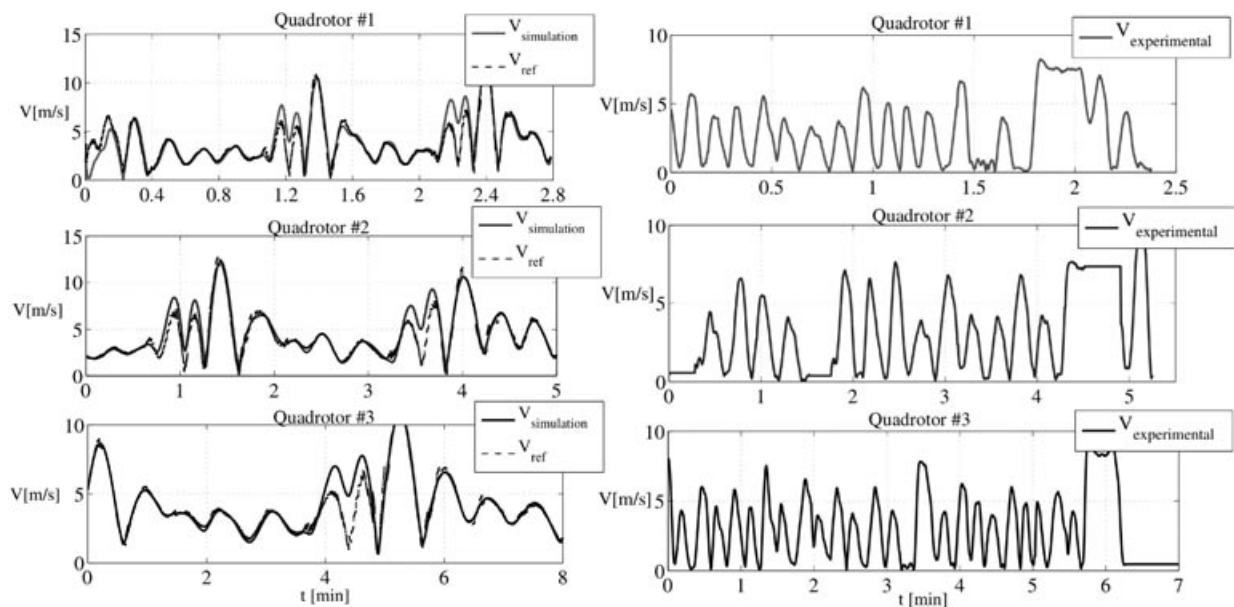


Figure 15. Simulation (left) vs. experimental results (right) of velocity tracking.

mentioned factor, the human efforts for mission setup. Although the AVs are full autonomous platforms, each one must have an experienced pilot and/or a ground base operator who supervises the mission. In this sense, we are adopting a configuration similar to the one proposed in Murphy and Burke (2008), in which a pilot is in charge of supervising (and, if necessary, teleoperating) the UAVs and a *mission specialist* is in charge of supervising the mission. The typical and safer modality is to have two persons in charge for each drone. The pilot is responsible for supervising the mission from the ground and switches from the automatic to manual mode in case there is some system failure or emergency. The base-station operator is in charge of supervising the mission at the highest level (i.e., monitoring all the navigation data in real time). Although there is no specific security legislation applied to mini aerial vehicles, this measure is mandatory for safe experimentation.

The mission presented here was carried out by three pilots and two base-station operators. Table VIII shows the time dedicated before and after the mission with the AVs. This includes preliminary flights and communication tests (e.g., ensuring that the corresponding ground stations are correctly receiving and sending data). It is interesting to observe that the setup time increases with the area of coverage (i.e., geographic area).

As mentioned before, the coverage time is composed of an effective and a noneffective time. The effective time is the absolute time the aerial vehicle is moving between waypoints until the final coverage trajectory is performed. The noneffective time is the overall time hovering over the waypoints. The results obtained also show that both effective and noneffective times increase with the number of waypoints per subarea.

The results of the field tests performed can be summarized as follows:

- The accuracy of the navigation in the X - Y plane is affected by the kinds of turns.
- The coverage time increases with the number of turns and the number of waypoints (however, note that the number of turns is not necessarily related to the number of waypoints).
- Depending on the region where the mission will take place, peak wind stream velocities must be quantified and taken into account for good speed tracking. Simulation and tuning of the controllers shall take these into account.
- Around 50% of the overall mission time was devoted to system assessment and setup.
- Even if highly autonomous UAVs can be currently obtained off the shelf, efforts are still needed as far as usability and safety for their effective use for commercial applications, especially when multiple UAVs are to be employed.

Table VIII. Summary of the metrics measured *after* the flight.

UAV no.	T_E (s)	$T_{\bar{E}}$ (s)	T_C (s)	T_L (s)	T_M (s)	P^a (m)	Battery ^b (%)
1	141	42	183	282.76	465.76	373.74	7.64
2	151	45	196.43	81.85	278.28	339.75	8.399
3	213	75	288.02	313.25	601.27	650.52	12.39

^aInformation obtained from a georeferenced image with a precision of ± 3 m.

^bRelative battery consumption.

6. DISCUSSION AND CONCLUSIONS

In this paper, a practical solution for performing aerial imaging applied to precision agriculture by using a fleet of UAVs has been presented. This paper provides a unified view of the work we have been carrying out in different fields: multirobot task planning, path planning, and UAV control.

A simple and efficient strategy for the coverage of a crop field using aerial robots for data collection has been presented. The approach proposed includes two steps: in the first step a given area is subdivided by the team robots and each robot is assigned a subarea. In this phase the robots are aware of each other and carry out this task cooperatively. The second phase handles the workspace sampling and the coverage planning. In this phase the robots are unaware of each other and parallel path planners compute the coverage path for each robot. Finally, a robust flight controller drives the AVs through the mission.

The area partitioning and assignment distributed algorithm is capable of taking into account the characteristics of different vehicles, a feature useful in the case of heterogeneous fleets. The methodology employed for path planning guarantees finding an optimal solution if one exists, providing coverage paths with minimal turns to the aerial robots, ensuring that cells are not visited twice, and including fixed and known obstacle⁹ avoidance.

The planning step introduces security boundaries between subareas assigned to the vehicles. These should be carefully optimized in order to ensure suitable quality of service and at the same time guarantee the security and safety of the platforms. A way to get rid of the safety strips that are mainly responsible for reducing the area covered would be to take into account the cell dimensions in relation to the AVs' position accuracy and the probability of having two or more than two AVs flying at the same time in adjacent cells, analyzing the flight plans prior to the

⁹For instance, irrigation systems or other obstacles such as machinery.

beginning of the mission. In other words, a mission validation step, using the optional simulation tool, could at the same time eliminate the need for no-fly security strips and ensure collision-free missions. Alternatively, achieving 100% coverage will involve the integration of temporal considerations in the planner in order to always respect a safety area around the UAVs, for instance using a validating plan-merging operation (Gravot & Alami, 2001).

As shown by the field tests, reducing the number of turns is extremely important because turns affect both mission time and, more important, the precision of the 3D positioning. We can also conclude that, contrary to what was expected, position error is less affected by the speed of the vehicle.

As far as control is concerned, it was possible to assess the performances of the backstepping+FST technique adopted by the Hummingbird robots (in substitution for their original one) in comparison with the AR100 controller. The backstepping+FST technique has shown better robustness against large disturbances as model uncertainties were canceled thanks to integral action and the incorporation of FS theory. Precision during tracking was also improved. This feature clearly plays an important role in *open* fields, where the vehicles are exposed to rapidly changing conditions.

The field tests have also shown that local conditions (in this case, the different slopes) must be carefully taken into account because they can generate wind streams far more rapid than the maximum wind velocities used in simulations and measured at the base station. This is especially true in wide areas. This aspect did not appear in other field tests (e.g., cornfields, tests not reported here) due to the simpler orography of the terrain.

Although not applied yet, orography information can be incorporated in the planning stage, by adding preference terms to the cost model of the vehicles during the negotiations. Information regarding particular terrain-dependent wind conditions in given regions would be treated in the same way as in no-fly zones: vehicles would be denied to fly over regions where expected winds would overcome their limitations.

In general the operation protocol workflow has worked well, providing a smooth tool from area definition to mission execution. The decoupled area partitioning and path planning steps allow for different partitioning to be generated and examined by the operator prior to mission execution. Also, after path planning has been performed, the mission simulator allows simulating and validating the mission beforehand.

However, we would like to point out that even with a well-structured operations protocol and with highly autonomous robots (capable of taking off, performing the mission, and landing back to base autonomously), the resources needed for deploying the robot team and setting up the whole system play an important role. As far as human factors, with three UAVs at least four persons are needed

for a safe mission: one backup pilot for each vehicle plus one mission supervisor at the base station. As far as mission time is concerned, deployment time was about 50% of total time for a single mission.

This is only partially due to the fact that the system is not currently thought out for the end user, and its operation has to be done by technical staff. Much work will be needed in order to provide a simple and safe system to be operated directly by farmers, especially in terms of a unified, friendly GUI that provides task definition and mission execution monitoring by untrained persons. For this reason, our view is that a multi-UAV system as a farming tool such as the one presented here would be better exploited commercially by a company providing services (system plus technical staff) to different farmers in the same area.

Finally, it is important to remark upon the unified vision provided by this work. In the literature a number of papers can be found that regard either task planning path planning or UAV control, but few works provide a comprehensive analysis of a whole system, from the theoretical study of the individual technical issues to a complete field experimentation.

7. APPENDIX A: BACKSTEPPING+FST CONTROL

To assess the assumption of improving flight control using the backstepping+FST, this appendix shows simulation tests related to the accuracy of the aforementioned controller against a PID control in terms of disturbance rejection (strong wind). Many commercial autopilots use PID methodology that is capable of controlling the system during certain conditions, such as hovering or flight without the presence of strong perturbations. However, as experimentally demonstrated in Bouaballah (2007) and Olfati-Saber (2001) and also in previous work (Colorado, 2010), PID controllers are capable but not efficient in stabilizing the aircraft (in terms of attitude control) when nonlinearities produced by Coriolis accelerations play a significant role. This is produced when large disturbances cause changes in velocity that result from rotation. PID controllers have proven to be well adapted to the quadrotor when flying near hover, but the backstepping+FST is capable of faster attitude stabilization mainly because of the introduction of the terms $\ddot{\phi}^d$, $\ddot{\theta}^d$, and $\ddot{\psi}^d$ within control laws (11) and (12) respectively, which consider the effects of abrupt angular changes while flying at moderate speeds (FS foundation).

Figure A.1 shows the cornfield scenario. Flying at relatively high altitude (considering the sizes of the AVs), wind strength and gusts play a fundamental role. Hence, one of the main goals, as far as control is concerned, is to analyze how the controller is capable of stabilizing the system when strong wind disturbances are addressed.

For this experiment, the commercial PID-based autopilot of the Hummingbird quadrotor is compared with

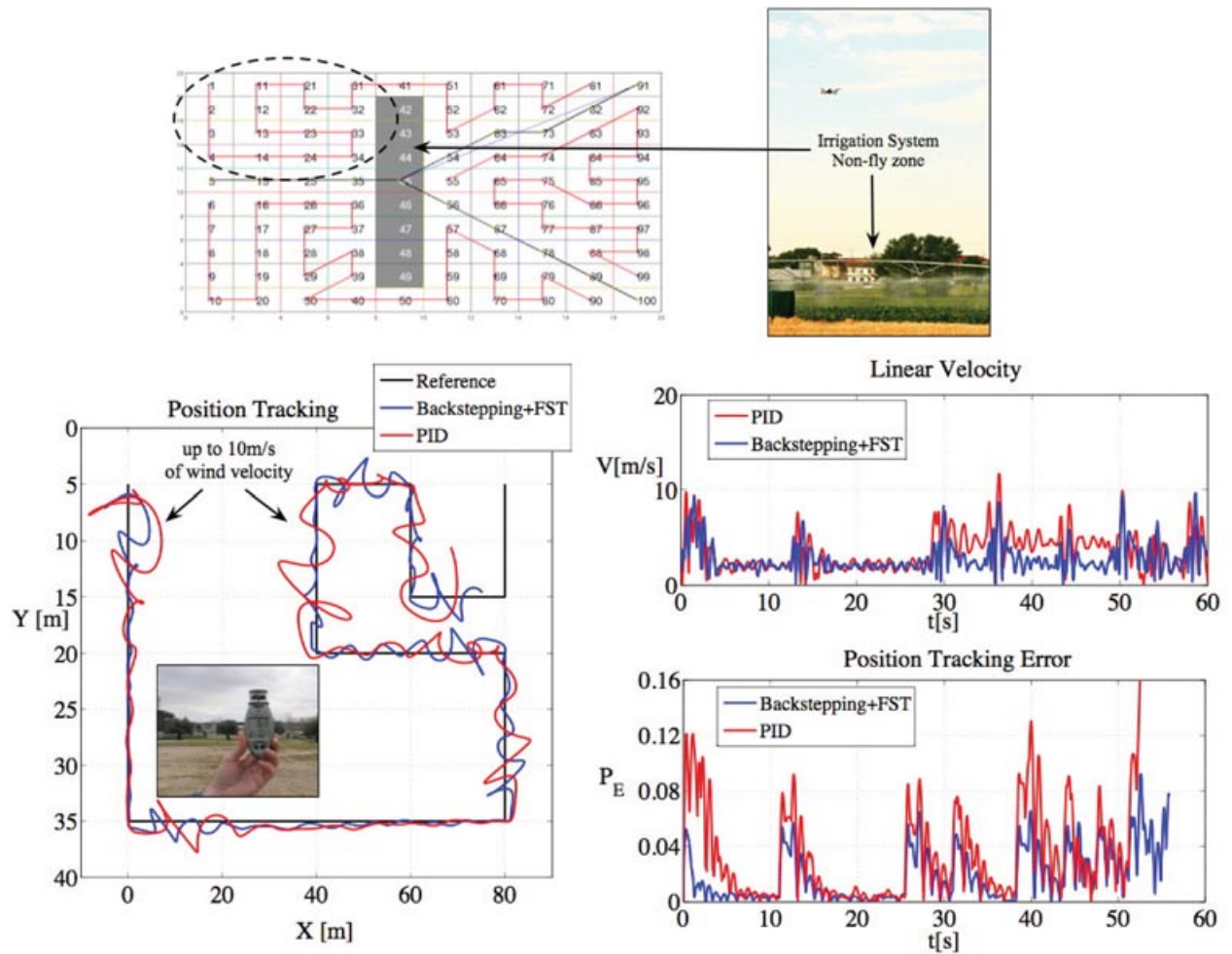


Figure A.1. (Simulation) control performance in terms of position tracking under strong wing perturbations. Backstepping+FST against PID.

Table A.1. Summary of the metrics measured during simulation.

	No.	h_{ref} (m)	h (m)	h_E (%)	V_{ref} (m/s)	V (m/s)	V_w^a (m/s)	P_E^b (%)
PID	1	20	20.04	0.2	2	2.3	10	2.5
	2	20	20.12	0.6	5	4.8	10	8
	3	20	20.2	1	10	9.8	10	23
BS+FST	1	20	20.04	0.2	2	2.05	10	1.2
	2	20	20.08	0.4	5	5.2	10	3.8
	3	20	20.17	0.85	10	10.3	10	6.7

^aWind speed introduced as a force disturbance.

^bAverage tracking error.

the backstepping+FST approach by measuring the position tracking of the trajectory within the circle. In addition, Figure A.1 also details the results of the Hummingbird flying at 20-m altitude and average linear speed of 5 m/s.

On average, the PID obtains maximum position tracking errors up to 8%, backstepping+FST about 3.8% under the presence of strong wing perturbations (wing velocity up to 10m/s). More details are given in Table A.1.

8. APPENDIX B: INDEX TO MULTIMEDIA EXTENSIONS

The video is available as Supporting Information in the on-line version of this article.

Extension	Media type	Description
1	Video	Steps performed for mission setup and experimental flights on the vineyards

The video accompanying this paper illustrates the steps performed during the experiments on the vineyards, the UAVs in flight, and the final result of the mission. The target region is shown, along with its geographical coordinates. Then the task partitioning and allocation process is shown, where three polygons, representing desired regions, evolve during the negotiation. The final agreed partition is shown along with the routes computed by the path planner. Next, the UAVs in action on the field are shown. Finally, the performed trajectories are superimposed on an image of the field to summarize flight results.

ACKNOWLEDGMENT

This work has been partially funded by project Robocity 2030-II (Comunidad de Madrid).

REFERENCES

- Botelho, S., & Alami, R. (1999, May). M+: A scheme for multi-robot cooperation through negotiated task allocation and achievement. In *Proceedings IEEE International Conference on Robotics and Automation (IRCA 99)* (pp. 1234–1239).
- Bouaballah, S. (2007). Design and control of quadrotors with application to autonomous flying. Ph.D. thesis, École Polytechnique Fédérale de Lausanne.
- Bresenham, J. E. (1965). Algorithm for computer control of a digital plotter. *IBM Systems Journal*, 4(1), 25–30.
- Burgos-Artizzu, X. P., Ribeiro, A., Tellaeche, A., Pajares, G., & Fernández-Quintanilla, C. (2010). Analysis of natural images and processing for the extraction of agricultural elements. *Image and Vision Computing*, 28, 138–149.
- Chen, J., Dawson, D., Salah, M., & Burg, T. (2006, June). Multiple UAV navigation with finite sensing zone. Paper presented at the American Control Conference, 2006, Minneapolis, MN.
- Choi, Y.-H., Lee, T.-K., Baek, S.-H., & Oh, S.-Y. (2009, October). Online complete coverage path planning for mobile robots based on linked spiral paths using constrained inverse distance transform. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009 (IROS 2009)*, St. Louis, MO (pp. 5788–5793).
- Choset, H. (2001). Coverage for robotics—A survey of recent results. *Annals of Mathematics and Artificial Intelligence*, 31(1–4), 113–126.
- Choset, H., Acar, E. U., Rizzi, A. A., & Luntz, J. E. (2000, April). Exact cellular decompositions in terms of critical points of Morse functions. In *Proceedings IEEE International Conference on Robotics and Automation (ICRA 00)*, San Francisco, CA (pp. 2270–2277).
- Colorado, J. (2010). Towards Mav autonomous flight: A modeling and control approach. Master's thesis, Universidad Politécnica de Madrid. Retrieved May 15, 2010, <http://oa.upm.es/3493/>.
- Colorado, J., Barrientos, A., Martinez, A., Lafaverge, B., & Valente, J. (2010). Mini-quadrotor attitude control based on hybrid backstepping and Frenet-Serret theory. In *Proceedings IEEE International Conference on Robotics and Automation (ICRA 10)*, Anchorage, AK (pp. 1617–1622).
- Curran, P. (1985). Aerial photography for the assessment of crop condition: A review. *Applied Geography*, 5(4), 347–360.
- Dias, M. B., & Stentz, A. T. (2003). Traderbots: A market-based approach for resource, role, and task allocation in multirobot coordination (Tech. Rep. CMU-RI-TR-03-19). Pittsburgh, PA: Robotics Institute.
- Dias, M. B., Zlot, R., Kalra, N., & Stentz, A. (2006). Market-based multirobot coordination: A survey and analysis. *Proceedings of the IEEE*, 94(7), 1257–1270.
- Gancet, J., Hattenberger, G., Alami, R., & Lacroix, S. (2005a, April). An approach to decision in multi-UAV systems: Architecture and algorithms. In *Proceedings of the ICRA-2005 Workshop on Cooperative Robotics*, Barcelona, Spain.
- Gancet, J., Hattenberger, G., Alami, R., & Lacroix, S. (2005b, August). Task planning and control for a multi-UAV system: Architecture and algorithms. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005 (IROS 2005)*, Edmonton, Alberta, Canada (pp. 1017–1022).
- Gerkey, B. P., & Mataric, M. J. (2002). Sold!: Auction methods for multirobot coordination. *IEEE Transactions on Robotics and Automation*, 18, 5.
- Golfarelli, M., Maio, D., & Rizzi, S. (1997). A task-swap negotiation protocol based on the contract net paradigm (Tech. Rep. 005–97). Università di Bologna, Italy: DEIS, CSITE.
- Gravot, F., & Alami, R. (2001, May). An extension of the planning paradigm for multi-robot coordination. In *Proceedings IEEE International Conference on Robotics and Automation (IRCA 01)*, Seoul, Korea (vol. 3, pp. 2929–2934).
- Jiao, Y.-S., Wang, X.-M., Chen, H., & Li, Y. (2010, June). Research on the coverage path planning of UAVs for polygon areas. In *the 5th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, Taichung, Taiwan (pp. 1467–1472).
- Lacroix, S., Alami, R., Lemaire, T., Hattenberger, G., & Gancet, J. (2007). Decision making in multi-UAVs systems: Architecture and algorithms. In A. Ollero and I. Maza (Eds.), *Multiple heterogeneous unmanned aerial vehicles* (vol. 37 of Springer Tracts in Advanced Robotics, pp. 15–48). Berlin: Springer.

- Latombe, J.-C. (1991). Robot motion planning. Norwell, MA: Kluwer Academic Publishers.
- LaValle, S. M. (2006). Planning algorithms. Cambridge, UK: Cambridge University Press.
- Lemaire, T., Alami, R., & Lacroix, S. (2004, April). A distributed tasks allocation scheme in multi-UAV context. In Proceedings IEEE International Conference on Robotics and Automation (IRCA 04), New Orleans, LA (vol. 4, pp. 3622–3627).
- Luo, C., & Yang, S. (2008). A bioinspired neural network for real-time concurrent map building and complete coverage robot navigation in unknown environments. *IEEE Transactions on Neural Networks* 19(7), 1279–1298.
- Maza, I., & Ollero, A. (2007). Multiple UAV cooperative searching operation using polygon area decomposition and efficient coverage algorithms. In R. Alami, R. Chatila, and H. Asama (Eds.), *Distributed autonomous robotic systems 6* (pp. 221–230). Tokyo: Springer.
- Moon, S.-W., & Shim, D. H.-C. (2009). Study on path planning algorithms for unmanned agricultural helicopters in complex environment. *International Journal of Aeronautical and Space Sciences*, 10(2), 1–11.
- Murphy, R., & Burke, J. (2008). From remote tool to shared roles. *IEEE Robotics & Automation Magazine*, 15(4), 39–49.
- Oetomo, D., Billingsley, J., & Reid, J. F. (2009). Editorial: Agricultural robotics. *Journal of Field Robotics*, 26(6–7), 501–503.
- Oh, J. S., Choi, Y. H., Park, J. B., & Zheng, Y. (2004). Complete coverage navigation of cleaning robots using triangular-cell-based map. *IEEE Transactions on Industrial Electronics*, 51(3), 718–726.
- Oksanen, T., & Visala, A. (2009). Coverage path planning algorithms for agricultural field machines. *Journal of Field Robotics*, 26, 651–668.
- Olfati-Saber, R. (2001). Nonlinear control of underactuated mechanical systems with application to robotics and aerospace vehicles. Ph.D. thesis, Massachusetts Institute of Technology.
- Ollero, A., & Maza, I. (2007). Multiple heterogeneous unmanned aerial vehicles, 1st ed. Berlin: Springer.
- Osborne, M. J., & Rubinstein, A. (1994). A course in game theory. Cambridge, MA: MIT Press.
- Parker, L. (2003). Current research in multirobot systems. *Artificial Life and Robotics*, 7, 1–5.
- Rossi, C., Aldama, L., & Barrientos, A. (2009). Simultaneous task subdivision and allocation for teams of heterogeneous robots. In *ICRA '09. IEEE International Conference on Robotics and Automation*, 2009 (pp. 946–951).
- Rubinstein, A. (1983). Perfect equilibrium in a bargaining model. *Econometrica*, 50, 97–109.
- Schmale, D. G. I., Dingus, B. R., & Reinholtz, C. (2008). Development and application of an autonomous unmanned aerial vehicle for precise aerobiological sampling above agricultural fields. *Journal of Field Robotics*, 25(3), 133–147.
- Smith, R. G. (1980). The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, C29–12, 1104–1113.
- Srinivasan, A. (Ed.) (2006). *Handbook of precision agriculture: Principles and applications*. Danvers, MA: CRC Press.
- Stentz, A., & Dias, M. (1999). A free market architecture for coordinating multiple robots. (Tech. Rep. CMU-RI -TR-99-42). Pittsburgh, PA: Carnegie Mellon University.
- Tisdale, J., Ryan, A., Kim, Z., Tornqvist, D., & Hedrick, J. (2008, June). A multiple UAV system for vision-based search and localization. In *American Control Conference*, 2008, Seattle, WA (pp. 1985–1990).
- Vincenty, T. (1975). Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations. *Survey Review*, 22(176), 88–93.
- Weiss-Cohen, M., Sirotin, I., & Rave, E. (2008, December). Lawn mowing system for known areas. In *2008 International Conference on Computational Intelligence for Modelling Control Automation*, Vienna, Austria (pp. 539–544).
- Wong, S., & MacDonald, B. (2003, October). A topological coverage algorithm for mobile robots. In *Proceedings. 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) Las Vegas, NV* (vol. 2, pp. 1685–1690).
- Zlot, R., & Stentz, A. (2005, April). Complex task allocation for multiple robots. In *Proceedings IEEE International Conference on Robotics and Automation (IRCA 05)*, Barcelona, Spain (pp. 1515–1522).
- Zlot, R., & Stentz, A. (2006). Market-based multirobot coordination for complex tasks. *International Journal of Robotics Research*, 25, 73–101.