Behavioral Control of Unmanned Aerial Vehicle Manipulator Systems

K. Baizid[1] · **G. Giglio**[2] · **F. Pierri**[2] · **M. A. Trujillo**[3] · **G. Antonelli**[4] · **F. Caccavale**[2] · **A. Viguria**[3] · **S. Chiaverini**[4] · **A. Ollero**[5]

Abstract In this paper a behavioral control framework is developed to control an Unmanned Aerial Vehicle-Manipulator (UAVM) system, composed by a multirotor aerial vehicle equipped with a robotic arm. The goal is to ensure vehicle-arm coordination and manage complex multi-task missions, where different behaviors must be encompassed in a clear and meaningful way. In detail, a control scheme, based on the Null Space-based Behavioral (NSB) paradigm, is proposed to handle the coordination between the arm and vehicle motion. To this aim, a set of basic functionalities (elementary behaviors) are designed and combined in a given priority order, in order to attain more complex tasks (compound behaviors). A supervisor is in charge of switching between the compound behaviors according to the mission needs and the sensory feedback. The method is validated on a real testbed, consisting of a multirotor aircraft with an attached 6 Degree of Freedoms manipulator, developed within the EU-funded project ARCAS (Aerial Robotics Cooperative Assembly System). At the the best of authors knowledge, this is the first time that an UAVM system is experimentally tested in the execution of complex multi-task missions. The results show that, by properly designing a set of compound behaviors and a supervisor, vehicle-arm coordination in complex missions can be effectively managed.

Keywords Aerial manipulation · Behavioral control · Unmanned Aerial Vehicles

1 Introduction

Research interest in aerial robotic systems has grown dramatically in recent years hand in hand with the number of applications involving Unmanned Aerial Vehicles (UAVs), such as surveillance and remote monitoring (Doitsidis et al, 2012), cooperative transportation (Maza et al, 2010), rescue missions (Maza and Ollero, 2011) and monitoring of hostile environments (Merino et al, 2012).

A novel application field for UAVs is aerial manipulation. Therefore, several mechanical structures have been considered: in Pounds et al (2011) a highly compliant gripper, composed by four fingers, is proposed, while several light-weight low-complexity grippers are tested in Mellinger et al (2011). Very recently, in order to extend manipulation capabilities, Unmanned Aerial Vehicle-Manipulator (UAVM) systems have been introduced, namely research platforms combining a multi-DOFs manipulator with a vertical take off and landing UAV. The presence of the manipulator often leads to a kinematically redundant system able to be adapted and reconfigured according to task requirements; however, the manipulator's motion generates reaction forces on the UAV that can have destabilizing effects (Kondak et al, 2013). The interaction between the arm and the vehicle have been tackled in Huber et al (2013) and Kondak et al (2014), where the dynamic coupling of an helicopter with a 7-DOFs robotic manipulator is analyzed, while in Antonelli and Cataldi (2014) an

[1]K. Baizid is with Institut Mines-Télécom, Mines-Douai, Lille, France. E-mail: khelifa.baizid@mines-douai.fr ·
[2]F. Caccavale, G. Giglio and F. Pierri are with University of Basilicata, Potenza, Italy. E-mail: {fabrizio.caccavale,gerardo.giglio,francesco.pierri}@unibas.it ·
[3]M.A. Trujillo and A. Vigura are with Center for Advanced Aerospace Technologies (CATEC), Sevilla, Spain. E-mail: {aviguria,matrujillo}@catec.aero ·
[4] G. Antonelli and S. Chiaverini are with University of Cassino and Southern Lazio, Cassino, Italy. E-mail: {antonelli,chiaverini}@unicas.it ·
[5]A. Ollero is with University of Seville, Sevilla, Spain. E-mail: aollero@us.es

adaptive scheme aimed at compensating the manipulator's mass and the interaction caused by its movements is proposed. In Fumagalli et al (2014), authors present the design, modeling and control of an aerial manipulator consisting of a quadrotor helicopter endowed with a robotic manipulator based on a 3-DOFs delta structure and a 3-DOFs end-effector based on a Cardan gimbal. In Kim et al (2013) the dynamic equation of a quadrotor equipped with a 2-DOFs manipulator is derived, and used in an adaptive controller, whose effectiveness is experimentally validated. In Orsag et al (2013a) a Lyapunov-based model reference adaptive control is proposed, while in Orsag et al (2013b) the control of a light-weight prototype 3-arm manipulator, each with 2 DOFs is considered. In Lippiello and Ruggiero (2012a) the dynamic model of the whole UAVM system is devised and a Cartesian impedance control is developed, in such a way to face the contact forces and external disturbances. Through a hierarchical control architecture, the motion control of the end-effector of an UAVM has been tackled in Arleo et al (2013), that has been extended in Caccavale et al (2014) via an adaptive term for compensating the model uncertainties. In Lippiello and Ruggiero (2012b), the intrinsic redundancy given by the robotic arm mounted on the UAV is exploited by resorting to a prioritized task-sequencing algorithm.

To achieve complex tasks, an accurate coordination of the vehicle and arm motion must be achieved. Moreover, the UAVM system usually needs to perform several motion tasks simultaneously. To this aim, a behavior-based approach (Arkin, 1998) can be appealing, since it allows to navigate autonomously in complex and unknown environments by using sensors to obtain information of the environment without the need of an accurate off-line planning. To exploit the redundancy of the robotic systems, the null space based behavioral (NSB) approach (Antonelli et al, 2008), based on the inverse kinematics technique, has been widely applied for grounded mobile robots (Antonelli et al, 2010), and recently extended for aerial manipulation systems in Baizid et al (2014); Antonelli et al (2014a) and Antonelli et al (2014b). More in detail, in Baizid et al (2014) two NSB approaches for obstacle avoidance of multi-UAVMs systems involved in transportation tasks are proposed. In Antonelli et al (2014a,b), a Control software Architecture for cooperative multi-UAVM (CAVIS) is developed.

The use of multi-priority control, both at kinematic or dynamic level, is not new. A good state of the art may be found, e.g., in Chiaverini et al (2008). By limiting our attention to floating-base manipulation, it is worth mentioning the underwater case study, where recently some experimental results have been obtained in a grasping operation (Simetti et al, 2013) with a system characterized by 13 degrees of freedom taking into account several prioritized tasks run by means of proper activating functions. In a sense, the humanoid case is also a floating-base manipulation task with a limited mobility of the torso when the robot is not walking. Within this framework, multipriority control has been addressed, among the others, by Escande et al (2014) within a nonlinear programming theory and Ott et al (2015) by resorting to model-based operational space techniques. It is worth noticing that both Escande et al (2014) and Simetti et al (2013) also address the important aspect of inequality constraints, i.e., control variables that need to be kept in a range of values instead of an exact one.

This work is part of the EU-funded ARCAS (Aerial Robotics Cooperative Assembly System) project (ARCAS, 2011), aimed at developing one of the first cooperative free-flying robot systems for assembly and structure construction. A first important step toward this goal is to ensure proper vehicle-manipulator coordination when the system is involved in complex missions. A major contribution of this paper is to tackle the vehicle-manipulator coordination problem for an underactuated quadrotor helicopter equipped with a manipulator via a behavioral control framework. Although the behavioral control approach has been already applied to other floating (or mobile) base robotic systems, aerial manipulators are novel robotic platforms, still not well investigated, which involve very different challenging issues with respect to other robotic systems, such as, e.g., underactuation. Moreover, the proposed control approach is experimentally validated on a prototype platform. At the authors' best knowledge, this is the first time in which the use of aerial manipulators for the execution of non-trivial missions is demonstrated in practice. The control algorithm includes two layers. In the upper level a kinematic inversion algorithm is aimed at computing the reference values for the actuated motion variables of the UAVM system (i.e., the position and yaw angle for the vehicle and the joint positions for the manipulator). At this layer, in order to achieve multiple goals, the redundancy of the UAVM system is exploited by resorting to the task-priority Null Space Behavioral approach. To the purpose, a set of complex tasks, called *compound behaviors*, are determined *a priori* by the user, on the basis of the mission needs. Each compound behavior is composed by a set of *elementary behaviors*, arranged in a given priority order. Then, during the mission, a Supervisor is in charge of selecting the appropriate compound behavior to be activated, according to sensory feedback on the UAVM and environment state. The second layer is a motion controller aimed at tracking the reference output of the upper layer. For validation purposes, a prototype, developed within the ARCAS project, available at the CATEC (Centro Avanzado de Tecnologas Aeroespaciales) research center in Seville, has been used. It is composed of an eight rotor aircraft in coaxial configuration equipped with a 6-DOFs manipulator.

A preliminary version of this paper has been presented in Baizid et al (2015). Here, we include further details regarding the motion controller and an accurate description of the supervisor, not addressed in Baizid et al (2015). In detail, the finite state supervisors for the two case studies are introduced and consistency and completeness analyses are performed. Preliminary experiments regarding the extension of the proposed scheme to multi-UAVMs systems are reported in Muscio et al (2016).

The paper is organized as follows; in Section 2 there is an overview on kinematic modeling of the UAVM system; in Section 3 the proposed behavioral control approach is detailed, including the description of elementary and compound behaviors and the supervisor; in Section 4 the motion controller acting on the ARCAS prototype is presented; in Section 5 the experimental setup is described and the experimental results are reported and discussed; finally in Section 6 some conclusions are given underlying the open issues for future work.

2 Kinematic model of UAVM

Consider a system composed by a quadrotor vehicle equipped with a n_M-DOFs robotic arm; let $\mathcal{F}_V$ be the coordinate frame attached to the center of mass of the vehicle's body and $\mathcal{F}_E$ be the coordinate frame attached to the manipulator's end-effector (Fig. 1). The position and orientation of $\mathcal{F}_V$ with respect to a common inertial reference frame, are given the (3×1) vector $\boldsymbol{p}_V$ and the (3×3) rotation matrix $\boldsymbol{R}_V(\boldsymbol{\phi}_V)$, respectively, where $\boldsymbol{\phi}_V = [\psi_V \, \theta_V \, \varphi_V]^{\mathrm{T}}$ is the triple of ZYX yaw-pitch-roll angles. The position $(\boldsymbol{p}_E)$ and the orientation $(\boldsymbol{R}_E)$ of the frame $\mathcal{F}_E$ with respect to the inertial frame can be written as:

$$\begin{cases} \boldsymbol{p}_E = \boldsymbol{p}_V + \boldsymbol{R}_V \boldsymbol{p}_{E,V}^V \\ \boldsymbol{R}_E = \boldsymbol{R}_V \boldsymbol{R}_E^V, \end{cases} \tag{1}$$

where $\boldsymbol{p}_{E,V}^V$ and $\boldsymbol{R}_E^V$ denote, respectively, the relative position and orientation of $\mathcal{F}_E$ with respect to $\mathcal{F}_V$, expressed in frame $\mathcal{F}_V$. The linear, $\dot{\boldsymbol{p}}_E$, and angular, $\boldsymbol{\omega}_E$, velocities of $\mathcal{F}_E$ can be expressed, by differentiating (1),

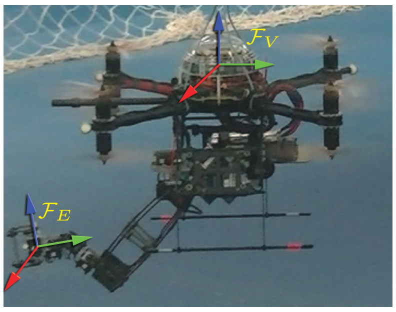


Fig. 1 UAVM system available at CATEC in Sevilla, with the end-effector and vehicle frames.

as

$$\begin{cases} \dot{\boldsymbol{p}}_E = \dot{\boldsymbol{p}}_V - \boldsymbol{S}(\boldsymbol{R}_V \boldsymbol{p}_{E,V}^V)\boldsymbol{\omega}_V + \boldsymbol{R}_V \dot{\boldsymbol{p}}_{E,V}^V \\ \boldsymbol{\omega}_E = \boldsymbol{\omega}_V + \boldsymbol{R}_V \boldsymbol{\omega}_{E,V}^V, \end{cases} \tag{2}$$

where $\boldsymbol{S}(\cdot)$ is the (3×3) skew-symmetric matrix operator performing the cross product (Siciliano et al, 2009) and $\boldsymbol{\omega}_{E,V}^V = \boldsymbol{R}_V{}^{\mathrm{T}}(\boldsymbol{\omega}_E - \boldsymbol{\omega}_V)$ is the relative angular velocity between $\mathcal{F}_E$ and $\mathcal{F}_V$ expressed in the frame $\mathcal{F}_V$.

Let $\boldsymbol{q}$ be the $(n_M \times 1)$ vector of manipulator joint positions, $\boldsymbol{p}_{E,V}^V(\boldsymbol{q})$ and $\boldsymbol{R}_E^V(\boldsymbol{q})$ represent the direct kinematics equations of the manipulator with respect to frame $\mathcal{F}_V$. Thus, the (6×1) end-effector's generalized velocity relative to $\mathcal{F}_V$,

$$\boldsymbol{v}_{E,V}^V = \begin{bmatrix} \dot{\boldsymbol{p}}_{E,V}^V \\ \boldsymbol{\omega}_{E,V}^V \end{bmatrix},$$

can be expressed in terms of the joint velocities $\dot{\boldsymbol{q}}$ via the manipulator Jacobian, $\boldsymbol{J}_{E,V}^V$, namely

$$\boldsymbol{v}_{E,V}^V = \boldsymbol{J}_{E,V}^V(\boldsymbol{q})\dot{\boldsymbol{q}}. \tag{3}$$

Based on (2) and (3), the generalized end-effector velocity, $\boldsymbol{v}_E = \left[\dot{\boldsymbol{p}}_E{}^{\mathrm{T}} \boldsymbol{\omega}_E{}^{\mathrm{T}}\right]^{\mathrm{T}}$, can be written as

$$\boldsymbol{v}_E = \boldsymbol{G}_V^T(\boldsymbol{R}_V, \boldsymbol{q})\boldsymbol{v}_V + \boldsymbol{J}_{E,V}(\boldsymbol{R}_V, \boldsymbol{q})\dot{\boldsymbol{q}}, \tag{4}$$

where $\boldsymbol{v}_V = \left[\dot{\boldsymbol{p}}_V{}^{\mathrm{T}} \boldsymbol{\omega}_V{}^{\mathrm{T}}\right]^{\mathrm{T}}$,

$$\boldsymbol{G}_V = \begin{bmatrix} \boldsymbol{I}_3 & \boldsymbol{O}_3 \\ \boldsymbol{S}(\boldsymbol{R}_V \boldsymbol{p}_{E,V}^V) & \boldsymbol{I}_3 \end{bmatrix}, \quad \boldsymbol{J}_{E,V} = \begin{bmatrix} \boldsymbol{R}_V & \boldsymbol{O}_3 \\ \boldsymbol{O}_3 & \boldsymbol{R}_V \end{bmatrix} \boldsymbol{J}_{E,V}^V,$$

while $\boldsymbol{I}_3$ and $\boldsymbol{O}_3$ represent the (3×3) identity and null matrices, respectively.

By expressing the attitude of the vehicle and the end-effector in terms of yaw-pitch-roll angles, $\boldsymbol{\phi}_V$ and $\boldsymbol{\phi}_E$, the differential kinematics (4) can be rearranged in terms of the operational-space vectors, $\boldsymbol{x}_V = \left[\boldsymbol{p}_V^{\mathrm{T}} \; \boldsymbol{\phi}_V^{\mathrm{T}}\right]^{\mathrm{T}}$ and $\boldsymbol{x}_E = \left[\boldsymbol{p}_E^{\mathrm{T}} \; \boldsymbol{\phi}_E^{\mathrm{T}}\right]^{\mathrm{T}}$, as

$$\dot{\boldsymbol{x}}_E = \bar{\boldsymbol{T}}^{-1}(\boldsymbol{\phi}_E)\left[\boldsymbol{G}_V^T(\boldsymbol{q}, \boldsymbol{\phi}_V)\bar{\boldsymbol{T}}(\boldsymbol{\phi}_V)\dot{\boldsymbol{x}}_V + \boldsymbol{J}_{E,V}(\boldsymbol{\phi}_V, \boldsymbol{q})\dot{\boldsymbol{q}}\right],$$

(5)

where the relationship between the generalized velocities and the operational-space vectors, namely $\dot{\boldsymbol{x}}_E = \bar{\boldsymbol{T}}^{-1}(\boldsymbol{\phi}_E)\boldsymbol{v}_E$ and $\boldsymbol{v}_V = \bar{\boldsymbol{T}}(\boldsymbol{\phi}_V)\dot{\boldsymbol{x}}_V$, have been exploited and

$$\bar{\boldsymbol{T}}(\boldsymbol{\phi}_*) = \begin{bmatrix} \boldsymbol{I}_3 & \boldsymbol{O}_3 \\ \boldsymbol{O}_3 & \boldsymbol{T}(\boldsymbol{\phi}_*) \end{bmatrix}, \qquad * = V, E,$$

while $\boldsymbol{T}(\boldsymbol{\phi}_*)$ is the transformation matrix between the angular velocity $\boldsymbol{\omega}_*$ and the time derivative of the Euler angles $\dot{\boldsymbol{\phi}}_*$, namely $\boldsymbol{\omega}_* = \boldsymbol{T}(\boldsymbol{\phi}_*)\dot{\boldsymbol{\phi}}_*$ (Siciliano et al, 2009).

Equation (5) can be expressed in compact form as

$$\dot{\boldsymbol{x}}_E = \boldsymbol{J}(\boldsymbol{\phi}_E, \boldsymbol{\phi}_V, \boldsymbol{q})\dot{\boldsymbol{\zeta}}\,, \tag{6}$$

where

$$\boldsymbol{J} = \left[\bar{\boldsymbol{T}}^{-1}(\boldsymbol{\phi}_E)\boldsymbol{G}_V^T\bar{\boldsymbol{T}}(\boldsymbol{\phi}_V) \quad \bar{\boldsymbol{T}}^{-1}(\boldsymbol{\phi}_E)\boldsymbol{J}_{E,V}\right], \tag{7}$$

and $\boldsymbol{\zeta}$ is the vector of the motion variables given by

$$\boldsymbol{\zeta} = \begin{bmatrix} \boldsymbol{x}_V \\ \boldsymbol{q} \end{bmatrix}.$$

In case of a quadrotor-arm system, the vehicle is an under-actuated system, since only 4 independent control inputs are available against the 6 degrees of freedom. Usually, for the quadrotor helicopters, the position, $\boldsymbol{p}_V$, and the yaw angle, ψ_V, are the controlled variables, while pitch, θ_V, and roll, φ_V, angles are used as intermediate control inputs for position control (Kendoul et al, 2008). Therefore, it is worth defining the controlled, $\boldsymbol{\zeta}_c$, and uncontrolled, $\boldsymbol{\zeta}_u$, motion variables as

$$\boldsymbol{\zeta}_c = \begin{bmatrix} \boldsymbol{p}_V \\ \psi_V \\ \boldsymbol{q} \end{bmatrix}, \quad \boldsymbol{\zeta}_u = \begin{bmatrix} \theta_V \\ \phi_V \end{bmatrix}. \tag{8}$$

Thus, the differential kinematics (6) can be rearranged as

$$\dot{\boldsymbol{x}}_E = \boldsymbol{J}_c(\boldsymbol{\zeta})\dot{\boldsymbol{\zeta}}_c + \boldsymbol{J}_u(\boldsymbol{\zeta})\dot{\boldsymbol{\zeta}}_u, \tag{9}$$

where $\boldsymbol{J}_c$ and $\boldsymbol{J}_u$ are the controlled and uncontrolled Jacobian matrices, obtained from $\boldsymbol{J}$ by selecting the columns referred to the controlled and uncontrolled variables, respectively.

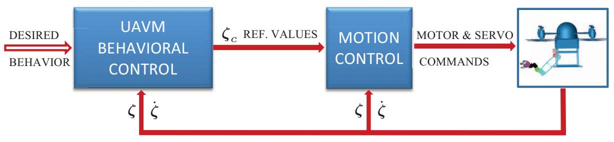


Fig. 2 Block scheme of the proposed controller.

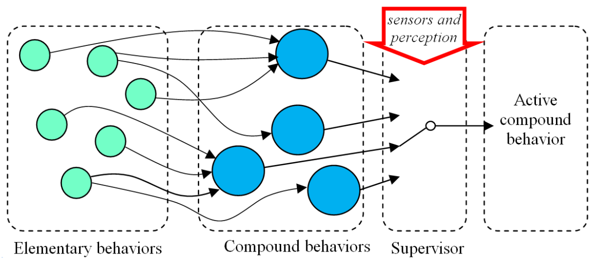


Fig. 3 Sketch illustrating the relationship between elementary, compound behaviors and supervisor.

3 Behavioral Control for Vehicle Manipulator System

The proposed control scheme is a two-layer kinematic behavioral control (Fig. 2) aimed at coordinating the motion of the vehicle and the manipulator. The upper layer is in charge of computing the reference trajectories for the manipulator joints as well as for the controlled variables of the vehicle (i.e., position and yaw angle). At this layer, since the vehicle manipulator system is kinematically redundant, the redundancy can be exploited to fulfill multiple tasks by adopting a task-priority algorithm, based on the NSB control approach (Antonelli et al, 2009, 2010). The second layer is a motion controller (detailed in Section 4) that is aimed at ensuring the tracking of the reference values computed by the upper layer.

The first layer can be seen as a three-level scheme (Fig. 3), including:

– *Elementary behaviors*, which are the atomic task functions to be controlled at the kinematic level;
– *Compound behaviors*, which are combinations of elementary behaviors, arranged in a priority order;
– *Supervisor*, which is in charge of switching between the defined compound behaviors on the basis of the UAVM and the environment state.

3.1 Elementary behaviors

An elementary task, or *elementary behavior*, of the system is encoded by a task variable $\boldsymbol{\sigma} \in \mathbb{R}^m$, which is function of the system's configuration, $\boldsymbol{\zeta}$, i.e.,

$$\boldsymbol{\sigma} = \boldsymbol{f}(\boldsymbol{\zeta}). \tag{10}$$

The configuration-dependent task Jacobian matrix $\boldsymbol{J}_\sigma \in \mathbb{R}^{m\times(6+n_M)}$ is defined via the differential relationship

$$\dot{\boldsymbol{\sigma}} = \frac{\partial \boldsymbol{f}(\boldsymbol{\zeta})}{\partial \boldsymbol{\zeta}}\dot{\boldsymbol{\zeta}} = \boldsymbol{J}_\sigma(\boldsymbol{\zeta})\dot{\boldsymbol{\zeta}} = \boldsymbol{J}_{\sigma,c}(\boldsymbol{\zeta})\dot{\boldsymbol{\zeta}}_c + \boldsymbol{J}_{\sigma,u}(\boldsymbol{\zeta})\dot{\boldsymbol{\zeta}}_u\,, \tag{11}$$

where $\boldsymbol{J}_{\sigma,c}$ and $\boldsymbol{J}_{\sigma,u}$ are the task Jacobians referred to the controlled and uncontrolled motion variables, respectively.

The kinematic control problem for the UAVM system can be formulated as to find reference values for the controlled variables, $\boldsymbol{\zeta}_{c,r}$, to be fed to a motion controller in order to ensure that the task variable reaches its desired value, $\boldsymbol{\sigma}_d$. The velocity reference, $\dot{\boldsymbol{\zeta}}_{c,r}$, can be computed via a closed-loop algorithm (Siciliano et al, 2009) as

$$\dot{\boldsymbol{\zeta}}_{c,r} = \boldsymbol{J}^\dagger_{\sigma,c}(\dot{\boldsymbol{\sigma}}_d + \boldsymbol{\Lambda}\tilde{\boldsymbol{\sigma}} - \boldsymbol{J}_{\sigma,u}\dot{\boldsymbol{\zeta}}_u), \tag{12}$$

where $\boldsymbol{J}^\dagger_{\sigma,c} = \boldsymbol{J}^{\mathrm{T}}_{\sigma,c}\left(\boldsymbol{J}_{\sigma,c}\boldsymbol{J}^{\mathrm{T}}_{\sigma,c}\right)^{-1}$ is a right pseudo-inverse of $\boldsymbol{J}_{\sigma,c}$, $\boldsymbol{\Lambda}$ is a constant positive-definite matrix of gains and $\tilde{\boldsymbol{\sigma}} = \boldsymbol{\sigma}_d - \boldsymbol{f}(\boldsymbol{\zeta}_{c,r})$ is the task error.

Because of the pseudoinverse, inverse solution (12) generates the instantaneous minimum norm joint velocities (Siciliano, 1990). Since many motion controllers require the position references for the controlled variables, (12) can be integrated in such a way to obtain $\boldsymbol{\zeta}_{c,r}$. The adoption of a closed-loop inversion, thanks to the presence of the feedback term, allows to avoid drift phenomena of the solution, which can arise due to the numerical integration.

In the following, a set of possible elementary behaviors is provided with a brief description of the corresponding task functions. It is worth noticing that the following elementary behaviors are those used for the experiments (see Section 5) and additional behaviors can be designed according to the planned mission. A more complete list of elementary behaviors can be found in Antonelli et al (2014a).

3.1.1 End-Effector Configuration (EEC)

This elementary behavior is aimed at controlling the end-effector position and orientation simultaneously. This can be achieved by defining the task function

$$\boldsymbol{\sigma}_{EEC} = \begin{bmatrix}\boldsymbol{p}^{\mathrm{T}}_E & \boldsymbol{\phi}^{\mathrm{T}}_E\end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^6.$$

The corresponding task Jacobian is the matrix $\boldsymbol{J}$ defined in (7) (namely $\boldsymbol{J}_{EEC} = \boldsymbol{J} \in \mathbb{R}^{6\times 6+n_M}$).

3.1.2 Vehicle Position (VP)

This elementary behavior allows to control the vehicle motion along a planned trajectory. It is described by the task function $\boldsymbol{\sigma}_{VP} = \boldsymbol{p}_V \in \mathbb{R}^3$ and by the corresponding Jacobian $\boldsymbol{J}_{VP} = [\boldsymbol{I}_3\ \boldsymbol{O}_{3\times 3+n_M}] \in \mathbb{R}^{3\times 6+n_M}$, where the notation $\boldsymbol{O}_{\alpha\times\beta}$ represents the $(\alpha\times\beta)$ null matrix.

3.1.3 Vehicle Obstacle Avoidance (VOA)

If an obstacle is present along the planned trajectory of the vehicle, it should be able to fly around it without collision. Let $\boldsymbol{p}_{obs} \in \mathbb{R}^3$ denote the obstacle position, the task function is defined as

$$\sigma_{VOA} = \|\boldsymbol{p}_V - \boldsymbol{p}_{obs}\|^2 \in \mathbb{R},$$

with the corresponding Jacobian

$$\boldsymbol{J}_{VOA} = 2(\boldsymbol{p}_V - \boldsymbol{p}_{obs})^{\mathrm{T}}\,[\boldsymbol{I}_3\ \boldsymbol{O}_{3\times 3+n_M}] \in \mathbb{R}^{6+n_M}.$$

The desired value for σ_{VOA} is the square of a suitable defined safety distance. Clearly, if the obstacle is far from the planned trajectory, this task has a negative effect, since it attracts the vehicle to the sphere at a given distance from the obstacle. Thus, activation of the behavior must be properly handled.

3.1.4 End-effector Obstacle Avoidance (EEOA)

Such elementary behavior is analogous to the VOA, but it takes into account the case in which the obstacle is present along the planned trajectory of the end-effector. The task function is defined as

$$\sigma_{EEOA} = \|\boldsymbol{p}_E - \boldsymbol{p}_{obs}\|^2 \in \mathbb{R},$$

with the corresponding Jacobian

$$\boldsymbol{J}_{VOA} = 2(\boldsymbol{p}_E - \boldsymbol{p}_{obs})^{\mathrm{T}}\,[\boldsymbol{I}_3\ \boldsymbol{O}_3]\,\boldsymbol{J} \in \mathbb{R}^{6+n_M}.$$

As in the VOA behavior, the desired value for σ_{EEOA} is the square of a suitable defined safety distance.

3.1.5 Robot Nominal Configuration (RNC)

Often, a given dexterity of the manipulator can be achieved by controlling its position in the joint space. Therefore, it is required to move to an assigned position m joints (with $m \le n_M$) of the arm. This behavior can be described by the task function

$$\boldsymbol{\sigma}_{RNC} = \boldsymbol{\Pi}\boldsymbol{q} \in \mathbb{R}^m, \tag{13}$$

where $\boldsymbol{\Pi}$ is a $(m\times n_M)$ selection matrix that selects m elements from a vector. The task Jacobian is

$$\boldsymbol{J}_{RNC} = \begin{bmatrix}\boldsymbol{O}_{m\times 6} & \boldsymbol{\Pi}^{\mathrm{T}}\boldsymbol{\Pi}\end{bmatrix} \in \mathbb{R}^{m\times 6+n_M}.$$

3.1.6 Mechanical Joint Limits (MJL)

The manipulator exhibits mechanical limits for the joint mobility, namely each joint is allowed to move in a range. This task avoids the violation of such limits. The system is considered safe if $q_i \in [\underline{q}_i, \overline{q}_i]$ $(\forall i = 1, 2, \dots, n_M)$, where $\underline{q}_i$ and $\overline{q}_i$ are suitably chosen values (software limits) far enough from the lower and upper mechanical limit of the ith joint, respectively. Different task functions have been proposed in literature, in this paper the following choice has been considered (Mansard and Chaumette, 2009): $\boldsymbol{\sigma}_{MJL} = \sum_{i=1}^{n_M} l_i(q_i)$, where

$$l_i(q_i) = \begin{cases} \frac{(\underline{q}_i - q_i)^2}{2n_M}, & \text{if } q_i \le \underline{q}_i, \\ 0, & \text{if } \underline{q}_i < q_i \le \overline{q}_i, \\ \frac{(\overline{q}_i - q_i)^2}{2n_M}, & \text{if } q_i > \overline{q}_i, \end{cases}$$

The task Jacobian is $\boldsymbol{J}_{MJL} = [\boldsymbol{0}_{1\times 6}\ \boldsymbol{J}_l] \in \mathbb{R}^{6+n_M}$ where

$$\boldsymbol{J}_l = \left[\frac{\partial l_1}{\partial q_1}, \frac{\partial l_2}{\partial q_2}, \dots, \frac{\partial l_{n_M}}{\partial q_{n_M}}\right] \in \mathbb{R}^{n_M}.$$

It is worth noticing that such a task function is zero when all the joints are in their acceptable interval, while grows up when joints are out of the acceptable interval and move towards their mechanical limits. A careful choice of the limits $\overline{q}_i$ and $\underline{q}_i$ is needed since if they are too close to the mechanical limits the task function might be insufficient since, due to the dynamics of the system, joints cannot stop immediately. In the experiments the software limits have been set heuristically, a more accurate choice would require to consider the joint velocity and acceleration as well as limits on joint velocity and acceleration (Guarino Lo Bianco and Zanasi, 2003).

3.2 Redundancy resolution via NSB-based approach

The UAVM system is characterized by $4 + n_M$ DOFs, namely 4 DOFs for the quadrotor vehicle (position and yaw angle) and n_M DOFs for the arm. Thus, if $4 + n_M$ is larger than the number of DOFs required by the main task function, the system is *kinematically redundant* and the redundant DOFs can be exploited to fulfill multiple behaviors via a task-priority algorithm based on the NSB control (Antonelli et al, 2009, 2010). The overall reference velocity for the controlled variables can be obtained by merging the velocity due to each elementary behavior (computed via (12)), in such a way that the lower-priority behavior contributions are projected onto the null space of the higher-priority ones. In this way, the velocity components that conflict with the higher priority behaviors are removed, and, thus, the secondary behaviors can be achieved only if they are compatible with those at higher priority. The compatibility issues of elementary behaviors can be analysed on the basis of the concepts of task Jacobian orthogonality and independence (Antonelli, 2009).

By assigning n_b behaviors with a given priority, the overall system velocity is given by the following recursive scheme:

$$\dot{\boldsymbol{\zeta}}_{c,r} = \dot{\boldsymbol{\zeta}}_{c,1} + \sum_{k=2}^{n_b} \boldsymbol{N}_{1,k-1}\dot{\boldsymbol{\zeta}}_{c,k}, \tag{14}$$

$$\boldsymbol{N}_{1,k} = \left(\boldsymbol{I} - \boldsymbol{J}_{1,k}^{\dagger}\boldsymbol{J}_{1,k}\right), \tag{15}$$

where the subscript $k = 1, \dots, n_b$ represents the behavior priority, $\boldsymbol{N}_{1,k}$ is a projector onto the null space of the augmented Jacobian $\boldsymbol{J}_{1,k}$, given by

$$\boldsymbol{J}_{1,k} = \left[\boldsymbol{J}_1^{\mathrm{T}}\ \boldsymbol{J}_2^{\mathrm{T}} \dots \boldsymbol{J}_k^{\mathrm{T}}\right]^{\mathrm{T}}. \tag{16}$$

In order to meet the requirements of complex mission scenarios, elementary behaviors can be combined into complex tasks, named *compound behaviors*. A compound behavior is a hierarchical combination of a set of elementary behaviors arranged in a given priority order. Such compound behaviors, and thus the elementary behaviors' priority, are determined *a priori* by the user, on the basis of the mission's need and of practical considerations (e.g., safety behaviors, as obstacles avoidance, have often higher priority).

3.3 Supervisor

The adoption of the NSB paradigm implies that real-time switching between compound behaviors must be ensured by a *supervisor*, explicitly designed for a given mission. The supervisor (implemented, e.g., via a Finite State Automata) is in charge of dynamically selecting the compound behavior to be activated, according to the state of the robot and of the external environment. For example, a compound behavior including the obstacle avoidance must be activated when the distance between the obstacle and the vehicle is below a certain safety value. Many approaches might be pursued for the supervisor design; in this paper an Automata has been considered (Alur and Yannakakis, 2001). The adoption of a Finite State Automata implies that only a limited number of compound behaviors, mutually exclusive, can be assigned to the system. Moreover, to build the Automata, the finite set of all allowed compound behaviors (set of states of the Automata), as well as the set of causes forcing the UAVM to change its state (namely, the switching rules between the different behaviors), must be defined. The supervisor and the corresponding Automata, must be designed on the basis of the mission to be accomplished and strictly depend on the implemented compound behaviors. Examples of supervisors are described in detail in Section 5 and their main properties, such as completeness and consistency, are analysed (Ouimet and Lundqvist, 2007).

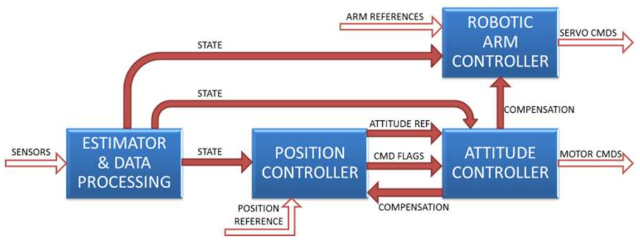


Fig. 4 Block scheme of the motion controller.

4 Motion Control

The behavioral control provides reference values for the vehicle position, for the vehicle yaw angle and for the manipulator joints (the controlled variables ζ_c). Therefore, it is necessary to develop a motion controller in charge of tracking such reference trajectories. It is worth noticing that the behavioral control is, in principle, compatible with any motion controller, since it only provides motion references. In this section, the controller implemented on the adopted UAVM system, developed within the ARCAS project and available at the CATEC, is presented.

A special control architecture has been developed to effectively control an aerial platform equipped with a 6-DOFs arm that weights significantly, compared to the total system's mass (see Section 5.1 for details). The ARCAS control layer architecture is composed by 4 main modules (Fig. 4): one is specifically designed for the robotic arm, while the others are standard modules for control of multirotor vehicles, that have been heavily modified to adapt them to the ARCAS system. It can be viewed as a multilayer control system: the basic layer is a classic PID controller, then the second layer is a mechanism aimed at moving a counterweight on a linear slider to compensate the arm movements; finally, in order to overcome mechanical limits of the previous mechanism, the residual static momentum due to the effect of the manipulator's gravity on the multirotor is compensated (Ruggiero et al, 2015).

The Estimator and Data Processing Module is in charge of estimating and processing the state of the complete system (position, attitude, angular velocity, servos data, sensors and safety operator radio references). The Position Controller Module takes care of the platform stabilization; its output is the reference for the attitude controller. The controller needs the state of the platform, the position reference given by the higher level controller and a compensation value coming from the attitude controller. The compensation signal (a (3×1) acceleration vector) is added to the output of the PIDs. This signal is generated in the attitude controller and it helps to stabilize the platform, rejecting the perturbations coming from the arm movement. The Robotic Arm Controller module is the one in charge of the final checks of the references given to the arm, its deployment, retraction and parsing the reference values to servo control signals. In addition, there is an emergency state in which the arm is retracted at a very high velocity. Finally, the Attitude Controller module runs the lowest level controller of the platform and it is the most complex one. It receives references from the position controller and stabilizes the platform, sending a control signals to the 8 motors. The problem of controlling the attitude of a multirotor with an equipped moving arm is addressed next by employing a two layer control system.

4.1 Attitude Controller

The first layer of the attitude control system is a mechanical compensation. The base of the manipulator is an auxiliary component fixed to the landing gear that supports the arm and also hosts the Center of Gravity (CoG) Displacement Compensation System (DCS). The DCS consists of a counterweight that is moved on a linear slider during the manipulator operation to keep the CoG of the whole system (UAV + manipulator + load + counterweight) as close as possible to the multirotor geometric center. The center is coincident with the CoG of the whole system when the robotic arm is retracted in its compact configuration. The components used as counterweight are all the batteries carried onboard. These batteries are heavy enough to compensate the manipulator CoG displacements with short movements.

The instantaneous CoG position of each link i, referred to the robotic arm fixed axis system $\mathcal{F}_0 = \{O_0, x_0, y_0, z_0\}$, is given by $(i = 1, \dots, 6)$

$$\begin{bmatrix} x_{A_i} \\ y_{A_i} \\ z_{A_i} \\ 1 \end{bmatrix}_O = \boldsymbol{T}_0^i \begin{bmatrix} x_{A_i} \\ y_{A_i} \\ z_{A_i} \\ 1 \end{bmatrix}_{O_i}, \tag{17}$$

where $\boldsymbol{T}_0^i \in \mathbb{R}^{4\times 4}$ is the homogeneous transformation matrix of link i obtained from the Denavit-Hartenberg (D-H) parameters table and updated by the servos feedback. Notice that $i = 7$ corresponds to the payload grasped by the robotic arm.

The robotic arm CoG position vector referred to the multirotor frame $\mathcal{F}_V$ is given by

$$\boldsymbol{p}_A^V = \boldsymbol{E}_3 \boldsymbol{T}_0^V \sum_{i=1}^{7} \left(\frac{m_i}{m_A} \begin{bmatrix} x_{A_i} \\ y_{A_i} \\ z_{A_i} \\ 1 \end{bmatrix}_O \right), \tag{18}$$

where m_i is the mass of the ith link, $m_A = \sum_{i=1}^{7} m_i$ is the total mass of the arm, $\boldsymbol{T}_0^V \in \mathbb{R}^{4\times 4}$ is the homogeneous transformation matrix from arm to vehicle frame and the (3×4) matrix $\boldsymbol{E}_3$ selects the first three components of a vector. It is supposed that the CoG of the platform is coincident with its geometric center, thus the position reference for the battery is computed via

$$m_A \boldsymbol{p}_A^V + m_b \boldsymbol{p}_B^V = \boldsymbol{0}_3 \Rightarrow \boldsymbol{p}_B^{V*} = \frac{m_A}{m_B} \boldsymbol{p}_A^V, \tag{19}$$

where $\boldsymbol{0}_3$ is the (3×1) null vector, $\boldsymbol{p}_B^V$ is the actual CoG position of the battery with respect to the vehicle frame and m_B is the battery mass. The position reference given to the servo is the projection of $\boldsymbol{p}_B^{V*}$ above the battery axis. As the battery movement is linear, it can only achieve part of the gravity compensation.

This system is really effective for slow motions of the robotic arm as it maintains the CoG of the full system very close to the geometric center. However, it has 2 limitations: the first one is the mechanical limits of the DCS that restrict the movement; the second one is that the mechanical compensation is often not fast enough to avoid dynamic unbalance, because of the servo limitations and software saturations. For that reason, a software compensation modifying the commanded propellers velocities, i.e., the velocities of the brushless motors aimed at rotating the propellers, is needed as they have a much quicker response. Namely, an additive term is added to the input in order to perform static momentum equilibrium around the geometric center of the platform by compensating the effect of the manipulator's gravity on the multirotor.

By assuming that the aerial platform is in hovering (or near hovering) conditions, any static torque around the yaw axis can be neglected; thus, the additive term, $\boldsymbol{\tau}_{xy}$, can be written as

$$\boldsymbol{\tau}_{xy} = \begin{bmatrix} \boldsymbol{f}_{sc} \\ \boldsymbol{\tau}_{sc} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0}_3 \\ \boldsymbol{E}_2 \left(m_A \boldsymbol{p}_A^V + m_B \boldsymbol{p}_B^V \right) \end{bmatrix}, \tag{20}$$

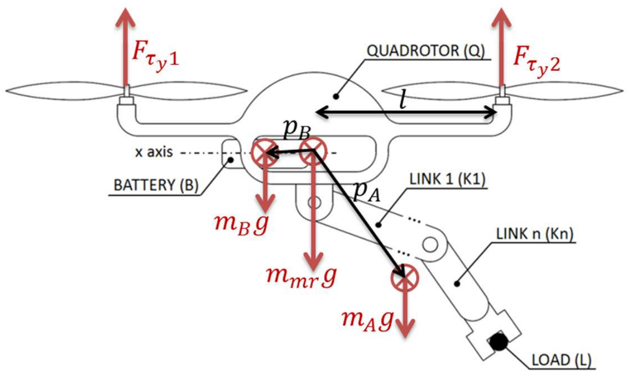


Fig. 5 Momentum equilibrium above the geometric center.

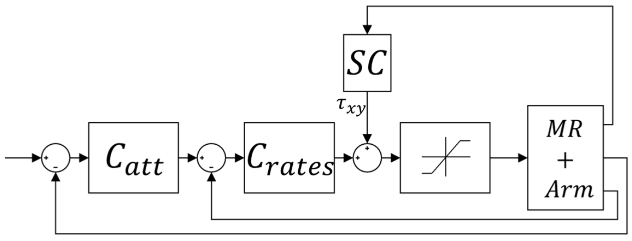


Fig. 6 Static compensation control scheme.

where $\boldsymbol{E}_2 \in \mathbb{R}^{3\times 3}$ selects the first two components and puts the third to zero.

The last part of the controller architecture consists in a roll-pitch-yaw controller, an angular rates controller and a torques/forces saturation. The SC module computes the torques using (20) and they are injected directly after the angular rates controller which outputs are forces and torques. As shown in (Ruggiero et al, 2015), the use of compensation (20) together with the DCS has been proven to improve the controller performance.

5 Experiments

The proposed behavioral control has been applied to the indoor platform of the ARCAS project (ARCAS, 2011), whose goal is the development of a cooperative free-flying robot system for assembly and structure construction. In the following, two experimental case studies, involving a certain number of different compound behaviors, are reported. In detail, in the first case study, the vehicle obstacle avoidance is considered, i.e., the vehicle modifies its trajectory in order to avoid an unexpected obstacle not taken into account during the planning, while the arm keeps a given configuration; in the second one the vehicle avoids the obstacle while the manipulator end-effector tracks the planned trajectory thanks to a suitable reconfiguration of the arm.

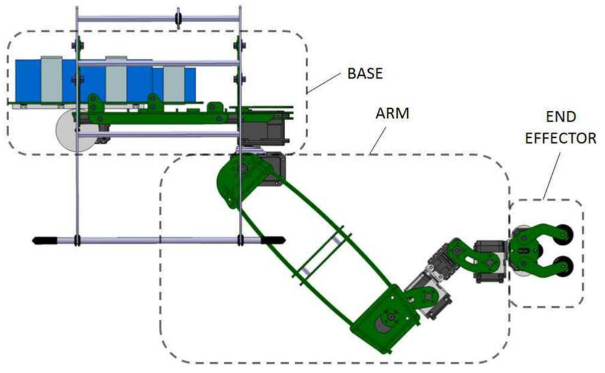


Fig. 7 Manipulator components (base, arm, and end-effector) mounted on multirotor landing gear

It is worth noticing that, during the experiments the take-off, the landing and the arm extension phases are managed manually for safety reasons; therefore, they are not taken into account in the supervisor design.

5.1 The ARCAS indoor platform

The ARCAS indoor platform is an eight rotor aircraft in coaxial configuration with a tip-to-tip wingspan of 105 cm, 33 cm propellers, height of 50 cm and mass of 8.2 kilograms, including the Lithium Polymer batteries and the robotic arm (see Fig. 1). The ARCAS 6-DOF manipulator (Cano et al, 2013) consists of three components (see Fig. 7): a fixed base, a multi-joint arm and an end effector. The multi-joint arm is an articulated component that contains all the manipulator's DOFs. It includes a first section with two motorized joints, followed by an elongated structure, and a second section composed by a chain of four motors driving the remaining joints. The fixed base of the manipulator has a mass of about 0.65 Kg, the moving parts of the manipulator have a total mass of about 0.767 Kg, while the vehicle frame with propellers, motors, onboard pc and sensors have a total mass of about 5 Kg. The moving mass of the batteries tray plus the batteries themselves is about 1.723 Kg. The current version of the arm is based on low-cost off-the-shelf motors; the inputs for the servos are the desired position of the joints, while the native local motion control loops are used. Future versions of the arm will be based on more costly and better performing actuators.

From the software architecture point of view, the ARCAS indoor platform counts two processing units: an autopilot and on-board computer. Both units are integrated into a common framework that has the following levels (Fig. 8):

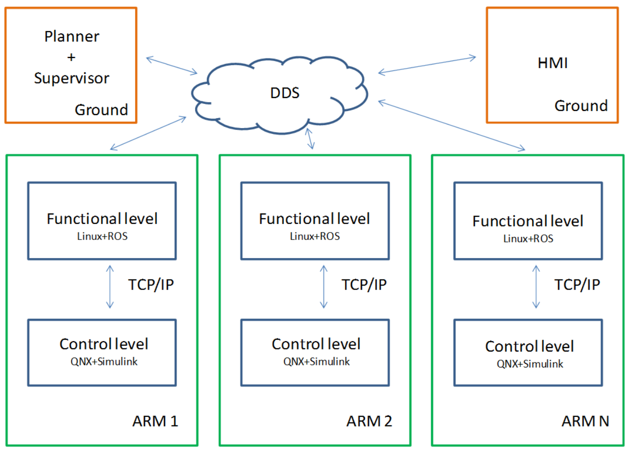


Fig. 8 ARCAS integration architecture.

- Control level: this level includes the integration of the control algorithms for the aerial platform and the robotic arm. Also, the standard estimation algorithms for the aerial robotic platform navigation are implemented at this level. The level uses a model-based development framework based on Matlab/ Simulink and implemented over the autopilot, using the real-time operating system QNX.
- Functional level: this level includes the integration of the perception and cooperation algorithms that will run on-board of the aerial robot. A Linux processing unit with ROS (Quigley et al, 2009) is used as the framework for the integration of the different functionalities.
- Multi-vehicle level: this level includes the integration of the software modules that require the information from multiple vehicles. The standard DDS middleware is used to interconnect the different software modules that will be integrated at this level.

With regards to the control level, the autopilot in use allows full control of all the hardware and software in order to integrate the robotic arm and the control algorithms developed in this paper. To test the control algorithms, the ARCAS project is using a Model-Based Design (MBD) methodology (Santamaria et al, 2012), based on Simulink©, relying on code generation tools. This methodology allows not only to easily integrate the code generated from Simulink into the target autopilot, but also, to access to all the internal variables and change parameters while the platform is flying. Also, the same software (Simulink) can be used in the design and simulation phase, as well as in software-in-the-loop testing and in real flying experiments.

For costly computing code, such as image processing or coordinated control, the system is equipped with an i7 Asctec Mastermind on board that runs ROS. This

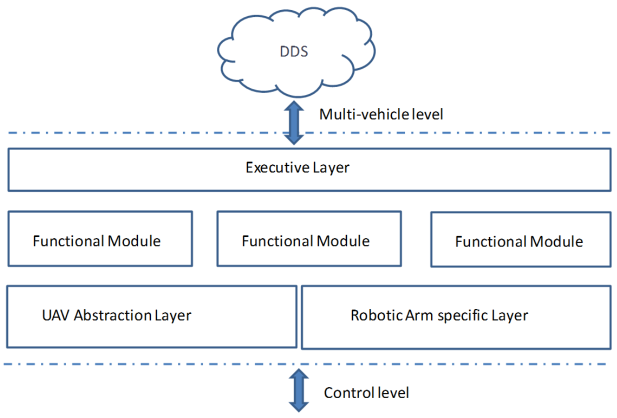


Fig. 9 Functional level integration architecture.

Fig. 10 Two views of CATEC indoor testbed.

unit implements what it has been called the functional level (Fig. 9). This functional level has a well-defined API with the control level, formed by two software modules: the UAV Abstraction layer and the Robotic Arm specific layer. These two layers allowed to develop a simulation system based on Gazebo simulator with exactly the same interfaces what the aerial robot actually has.

Finally, the experiments have been performed in the CATEC indoor testbed (Fig. 10) that has a useful volume, i.e., the area in which the UAVM is allowed to move, of $15 \times 15 \times 5$ meters, where this entire software infrastructure is deployed. The Vicon system (VICON, Ltd.) is used as the positioning system, which is extensively used for indoor environments. Vicon runs at 100Hz and it only provides the position of the multirotor, while the attitude is obtained with an estimator using the IMU and magnetometer data.

5.2 First case study

In this experiment, the arm and the vehicle are controlled separately. At the beginning, the vehicle takes off and, then, extends the arm: these tasks are manually managed for safety reason, thus they are not taken into account in the following description. The goal is to set the arm joints to a suitable configuration and move the vehicle according to a planned trajectory. In normal operating conditions, the compound behavior composed by the tasks VP and RNC is assigned to the system. The matrix $\boldsymbol{\Pi}$ in $\boldsymbol{\sigma}_{RNC}$ (see (13)) has been set as $\boldsymbol{\Pi} = \boldsymbol{I}_6$, namely all the joints are controlled. The reference velocities for the controlled variables, $\dot{\boldsymbol{\zeta}}_{c,r}$, are computed according to

$$\begin{aligned}\dot{\boldsymbol{\zeta}}_{c,r} = &\boldsymbol{J}_{VP}^{\dagger}\left[\dot{\boldsymbol{p}}_{V,d} + \boldsymbol{K}_{VP}(\boldsymbol{p}_{V,d} - \boldsymbol{\sigma}_{VP})\right] \\ &+\boldsymbol{J}_{RNC}^{\dagger}\boldsymbol{K}_{RNC}(\boldsymbol{\Pi}\boldsymbol{q}_d - \boldsymbol{\sigma}_{RNC}),\end{aligned} \quad (21)$$

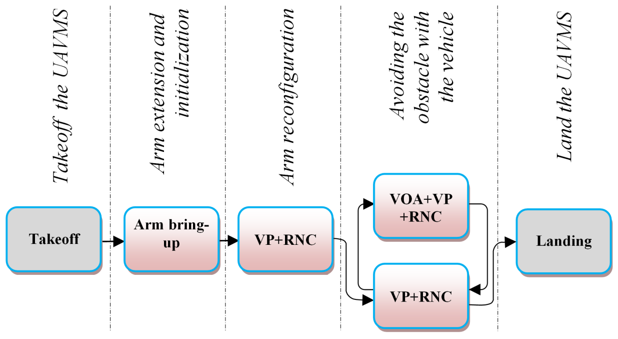
where $\boldsymbol{p}_{V,d}$ ($\boldsymbol{q}_d$) is the desired vehicle (joint) position, $\boldsymbol{K}_{VP}$ and $\boldsymbol{K}_{RNC}$ are positive definite matrix gains, whose values are reported in Table 1, and the null-space projection matrix has been dropped since the two tasks are fully independent (Antonelli, 2009). An obstacle, not taken into account during the path planning phase, intercepts the motion of the vehicle and the supervisor has to manage the switching from the compound behavior VP+RNC to a new compound behavior (VOA+VP +RNC), as shown in Fig. 11(a). The switching is commanded when the distance between the obstacle and the vehicle is below a safety value d_s (that has been set to 1 m), while, once the vehicle has overcome the obstacle, the supervisor switches back to VP+RNC and the vehicle moves back to the planned trajectory. For the new compound behavior the reference velocities are computed as

$$\begin{aligned}\dot{\boldsymbol{\zeta}}_{c,r} = &\boldsymbol{J}_{VOA}^{\dagger}k_{VOA}(d_s^2 - \sigma_{VOA}) + \boldsymbol{N}(\boldsymbol{J}_{VOA})\boldsymbol{J}_{VP}^{\dagger} \\ &\left[\dot{\boldsymbol{p}}_{V,d} + \boldsymbol{K}_{VP}(\boldsymbol{p}_{V,d} - \boldsymbol{p}_V)\right] + \boldsymbol{J}_{RNC}^{\dagger}\boldsymbol{K}_{RNC} \\ &(\boldsymbol{\Pi}\boldsymbol{q}_d - \boldsymbol{\sigma}_{RNC}),\end{aligned} \quad (22)$$

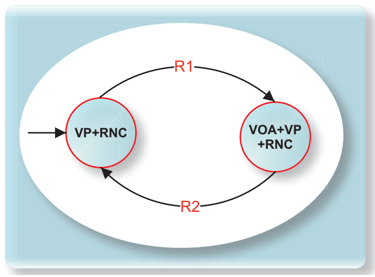
where k_{VOA} is a scalar gain and $\boldsymbol{N}(\boldsymbol{J}_{VOA})$ is a projector onto the null space of the Jacobian $\boldsymbol{J}_{VOA}$.

Remark 1 In the previous paragraph and in the following, the notation $B_1 + B_2 + \cdots + B_l$ represents a compound behavior including the elementary behaviors $B_1, B_2, \ldots, B_l$, in which B_1 has the highest priority and B_l the lowest priority. In the presence of fully independent behaviors, such as, e.g., VP and RNC, the order does not represent the behavior priority since both the behaviors can be fulfilled with the available degrees of freedom without any priority order.

Figure 11(b) shows the Finite State Automata and the switching rules that model the supervisor. In this case the supervisor includes only two states representing the above mentioned compound behaviors. It is worth verifying that the supervisor is consistent (namely only a single transition rule can be enabled at each time instant) and complete (namely it operates correctly for all possible input/state sequences). With



(a) Sequence of compound behaviors



R1: $\|\boldsymbol{p}_V - \boldsymbol{p}_{obs}\| < d_s$
AND $\dot{\boldsymbol{p}}_V^{\mathrm{T}}(\boldsymbol{p}_{obs} - \boldsymbol{p}_V) \geq 0$

R2: $\|\boldsymbol{p}_V - \boldsymbol{p}_{obs}\| \geq d_s$
AND $\dot{\boldsymbol{p}}_V^{\mathrm{T}}(\boldsymbol{p}_{obs} - \boldsymbol{p}_V) < 0$

(b) Supervisor automata

Fig. 11 First case study: sequence of compound behaviors and supervisor.

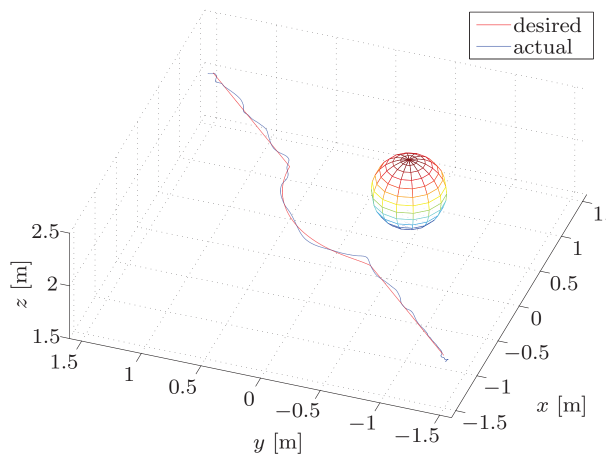


Fig. 12 First case study: 3-D vehicle trajectory.

regards to the consistency, at each time instant it is required that only one state can be active, and it is trivially verified since the two rules are mutually exclusive. Also the completeness can be easily verified since in the considered scenario the only admissible situations are:

– no obstacles are in the safety area (namely in a sphere of radius d_s): the system keeps the state VP+RNC;
– an obstacle is in the safety area (namely $\|\boldsymbol{p}_V - \boldsymbol{p}_{obs}\| < d_s$), the UAVM is in the state VP+RNC and the vehicle trajectory moves towards the obstacle (namely $\dot{\boldsymbol{p}}_V^{\mathrm{T}}(\boldsymbol{p}_{obs} - \boldsymbol{p}_V) \geq 0$): the supervisor switches to VOA+VP+RNC;
– the obstacle is in the safety area and the UAVM is in the state VOA+VP+RNC;
– the obstacle is on the border of the safety area (namely $\|\boldsymbol{p}_V - \boldsymbol{p}_{obs}\| \geq d_s$), the UAVM is in the state VOA+VP+RNC, and the vehicle trajectory moves away from the obstacle (namely $\dot{\boldsymbol{p}}_V^{\mathrm{T}}(\boldsymbol{p}_{obs} - \boldsymbol{p}_V) < 0$): the system switches to VP+RNC.

Figures 12-14 show the experimental results, obtained by setting the gain matrices as in Table 1. In Fig. 12 the 3-D trajectory of the vehicle with the presence of the obstacle is reported, while Fig. 13 shows the vehicle position error and the comparison between the reference vehicle trajectory, computed by the inverse kinematics, and the actual one. It is worth noticing that the position error presents its maximum (about 6 cm) when the vehicle modifies its trajectory in order to avoid the obstacle. Finally, Fig. 14 shows the distance between the vehicle center of mass and the obstacle in the presence of the obstacle avoidance behavior (green line). The red line represents the distance from the obstacle which would have been obtained if the vehicle had tracked the reference trajectory generated by the kinematic inversion (21), in the absence of the VOA elementary behavior. For safety reasons, such data have been obtained via numerical simulation. It can be noticed that, when the compound behavior VOA+VP+RNC is active, the distance is always close to the safety value, while the reference trajectory, in the absence of behavior VOA, is very close to the obstacle.

5.3 Second Case Study

In the second case study, the main objective is to track a given trajectory for the end-effector of the UAVM in terms of position and orientation, and, at same time, avoid collisions with an obstacle, not considered during the off-line planning, that is present along the vehicle path but not along the end-effector path. Thus, in order to avoid the obstacle and, at same time, track the end-effector trajectory, the arm must reconfigure itself during the vehicle obstacle avoidance phase (Fig. 15). To this aim, the elementary behaviors EEC, VOA, VP,

Table 1 Controller gains

Parameter	Value	Parameter	Value
$\boldsymbol{K}_{VP}$	$10\ \boldsymbol{I}_3$	$\boldsymbol{K}_{EEC}$	$10\ \boldsymbol{I}_6$
$\boldsymbol{K}_{RNC}$	$30\ \boldsymbol{I}_6$	k_{MJL}	10
k_{VOA}	50		

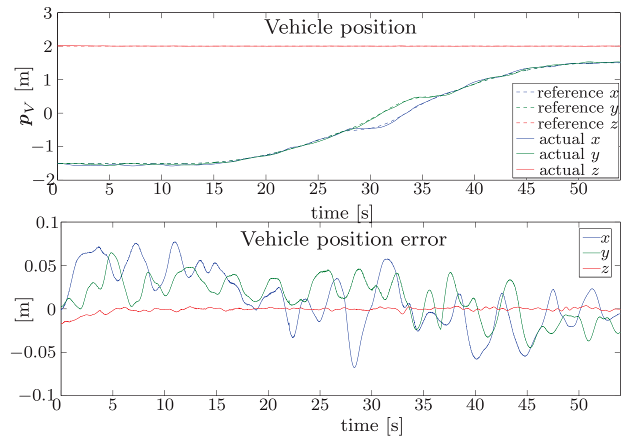


Fig. 13 First case study: vehicle position (top) and vehicle position error (bottom).

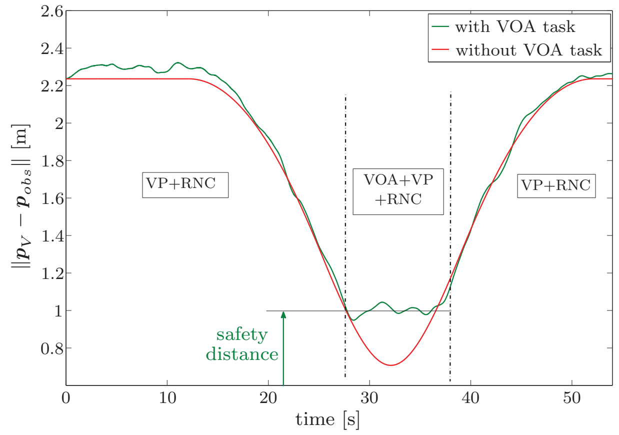


Fig. 14 First case study: distance between the vehicle and the obstacle during the experiment, when the VOA behavior is activated (green line); distance between the obstacle and the vehicle reference trajectory obtained, via numerical simulation, in the absence of the VOA behavior (red line).

RNC and MJL, already defined in Section 3.1, are activated during the experiment and combined into the compound behaviors shown in Fig. 16(a). At the beginning, after the take off and the arm extension (manually managed), the UAVM reconfigures the arm from the initial configuration in Fig. 15(a) to the configuration in Fig. 15(b), while keeping the vehicle position constant. Namely, the initial compound behavior includes the tasks VP and RNC. The matrix $\boldsymbol{\Pi}$ in $\boldsymbol{\sigma}_{RNC}$ has been set as

$$\boldsymbol{\Pi} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix},$$

in order to control the position of only the joints 2, 3 and 5, that are characterized by parallel axis. The reference velocity $\dot{\boldsymbol{\zeta}}_{c,r}$ is computed as in (21).

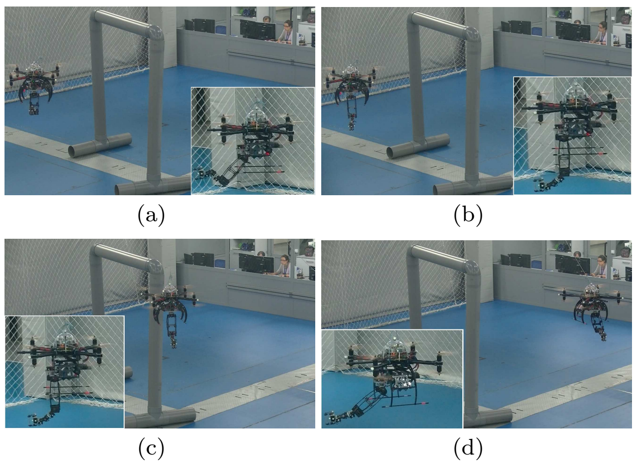


Fig. 15 Some snapshots of the second case study with zooms on the arm configuration.

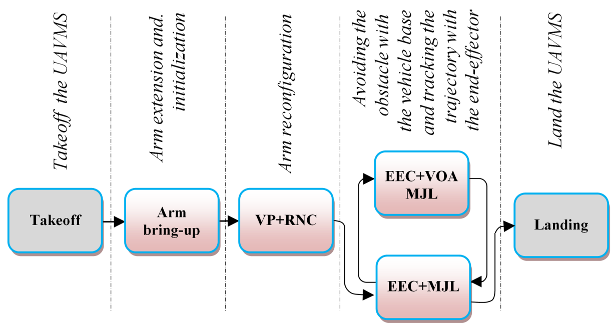
Once the arm reached its final configuration, $\boldsymbol{q}_{fin}$, with a certain accuracy, ϵ, the supervisor switches to a new compound behavior that is in charge of tracking the end-effector desired trajectory. Such compound behavior is composed by the elementary behaviors EEC and MJL. Namely, the main task is the tracking of the end-effector position and orientation, while the secondary task is aimed at avoiding the joint limits violation during the motion. As concerns the MJL behaviors, the limits $\overline{\boldsymbol{q}}$ and $\underline{\boldsymbol{q}}$ has been respectively set 5 deg less the upper mechanical limit and 5 deg over the lower mechanical limit. The reference velocity for the controlled variables is

$$\begin{aligned} \dot{\boldsymbol{\zeta}}_{c,r} = \boldsymbol{J}_c^{\dagger}&\left[\dot{\boldsymbol{x}}_{E,d} + \boldsymbol{K}_{EEC}(\boldsymbol{x}_{E,d} - \boldsymbol{\sigma}_{EEC}) - \boldsymbol{J}_{\sigma}\dot{\boldsymbol{\zeta}}_u\right] \\ &+\boldsymbol{N}(\boldsymbol{J}_c)\boldsymbol{J}_{MJL}^{\dagger}k_{MJL}(-\sigma_{MJL}), \end{aligned} \tag{23}$$

where $\boldsymbol{K}_{EEC}$ and k_{MJL} are positive definite matrix and scalar gain, respectively, $\boldsymbol{x}_{E,d}$ is the desired end-effector pose, and the desired value for the task function σ_{MJL} has been set to 0. Again, the values of the gains used in the experiment are reported in Table 1.

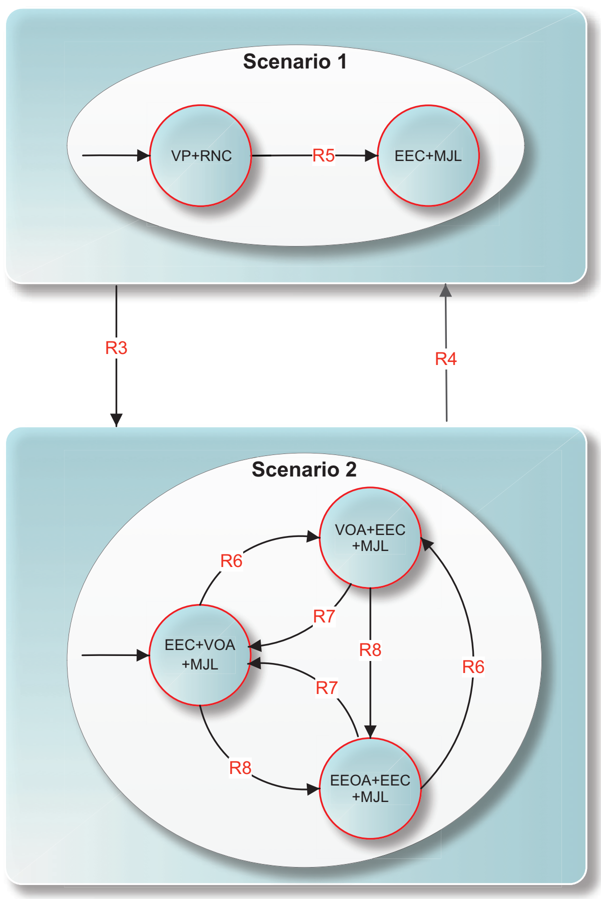
During this phase, the motion of the vehicle interferes with an obstacle, while the planned trajectory of the end-effector is safe. When the distance between the vehicle and the obstacle is below a safety value, d_s (set to 1.4 m), the compound behavior EEC+VOA+MJL is activated. Namely the vehicle modifies its trajectory in order to avoid the obstacle, while the end-effector keeps on the planned trajectory. The joint limit task, at lowest priority, is included to avoid dangerous situation during the arm reconfiguration. The reference velocity of this compound behavior is

$$\begin{aligned} \dot{\boldsymbol{\zeta}}_{c,r} = \boldsymbol{J}_c^{\dagger}&\left[\dot{\boldsymbol{x}}_{E,d} + \boldsymbol{K}_{EEC}(\boldsymbol{x}_{E,d} - \boldsymbol{\sigma}_{EEC}) - \boldsymbol{J}_{\sigma}\dot{\boldsymbol{\zeta}}_u\right] \\ &+\boldsymbol{N}(\boldsymbol{J}_c)\boldsymbol{J}_{VOA}k_{VOA}(d_s^2 - \sigma_{VOA})+ \\ &+\boldsymbol{N}(\boldsymbol{J}_{aug})\boldsymbol{J}_{MJL}^{\dagger}\boldsymbol{K}_{MJL}(-\sigma_{MJL}), \end{aligned} \tag{24}$$



Takeoff the UAVMS
Arm extension and initialization
Arm reconfiguration
Avoiding the obstacle with the vehicle base and tracking the trajectory with the end-effector
Land the UAVMS
Takeoff
Arm bring-up
VP+RNC
EEC+VOA MJL
EEC+MJL
Landing

(a) Sequence of compound behaviors



(b) Supervisor automata

Fig. 16 Second case study: sequence of compound behaviors and supervisor.

where $\boldsymbol{J}_{aug} = \left[\boldsymbol{J}_c^{\mathrm{T}}\ \boldsymbol{J}_{VOA}^{\mathrm{T}}\right]^{\mathrm{T}}$ is the augmented Jacobian.

Figure 15 shows some snapshots of the experiments. In detail, Fig. 15(a) shows the initial configuration of the arm and the vehicle, in Fig. 15(b) the configuration that the arm reached at the end of the first phase is reported, Fig. 15(c) shows the vehicle at the beginning of the obstacle avoidance and, finally, in Fig. 15(d) the final configuration, after overcoming the obstacle, is reported. It can be noticed that the arm configurations in Figs. 15(b) and 15(c) are very similar, while in Fig. 15(d) the arm has strongly modified its configuration in order to compensate the motion of the vehicle.

Figure 16(a) shows the sequence of compound behaviors activated during the whole mission, while Fig. 16(b) reports the Finite State Automata and the switching rules that model the supervisor. Based on the above described desired behavior of the UAVM (i.e., the mission), the set of compound behaviors to be implemented on board are VP+RNC, EEC+MJL and EEC+VOA +MJL. Even if not activated during the mission, two another compound behaviors must be defined for safety reason in case an obstacle is present in the safety area of the sole end-effector or of both the vehicle and the end-effector and the UAVM is moving toward the obstacle. In these cases, the obstacle avoidance must have higher priority than the end effector trajectory, since it is not possible to keep the desired trajectory of the end-effector without collisions with the obstacle; thus the compound behaviors EEOA+EEC+MJL and VOA +EEC+MJL must be implemented.

The supervisor can be arranged in a two-level hierarchical way (Alur and Yannakakis, 2001). At the top layer there are two scenarios: the first one corresponding to the situations in which no obstacles are in the safety area of the vehicle base, while in the second scenario an obstacle is close to the vehicle. At the lowest level there are the states corresponding to the compound behaviors. Let us verify that the supervisor is consistent and complete. With regards to the consistency, at each time instant it is required that only one scenario can be active, then for each scenario only one state can be active.

5.3.1 Consistency of the Scenario layer

The transition between the first and second scenario is commanded when the vehicle base and/or the end-effector are in the neighborhood of an obstacle and the vehicle trajectory moves towards the obstacle. It can be summarized by the following rule:

R3: $\{\|\boldsymbol{p}_V - \boldsymbol{p}_{obs}\| < d_s$ OR $\|\boldsymbol{p}_E - \boldsymbol{p}_{obs}\| < d_s\}$ AND $\{\dot{\boldsymbol{p}}_V^{\mathrm{T}}(\boldsymbol{p}_{obs} - \boldsymbol{p}_V) > 0$ OR $\dot{\boldsymbol{p}}_E^{\mathrm{T}}(\boldsymbol{p}_{obs} - \boldsymbol{p}_E) > 0\}$,

The transition rule for switching between the second and the first scenario is the following

R4: $\{\|\boldsymbol{p}_V - \boldsymbol{p}_{obs}\| \geq d_s\}$ AND $\{\|\boldsymbol{p}_E - \boldsymbol{p}_{obs}\| \geq d_s\}$ AND $\{\dot{\boldsymbol{p}}_V^{\mathrm{T}}(\boldsymbol{p}_{obs} - \boldsymbol{p}_V) \leq 0\}$,

namely, the obstacle is on the border of the safety area and the vehicle trajectory moves away from the obstacle. Since the two conditions are mutually exclusive it is trivially proven that only one scenario can be active at each time instant.

5.3.2 Consistency of the State layer

In the first scenario, two different states are present; the supervisor switches between the initial state (VP+RNC) and the state EEC+MJL when the desired final vehicle position and joint positions are reached with a certain accuracy, i.e.,

$$\text{R5: } \{\|\boldsymbol{q} - \boldsymbol{q}_{fin}\| \le \epsilon_q\} \text{ AND } \{\|\boldsymbol{p}_V - \boldsymbol{p}_{V,fin}\| \le \epsilon_V\},$$

where ϵ_q and ϵ_V denote, respectively, the desired accuracy for the arm joint position and for the vehicle position. Once the state EEC+MJL is activated, it remains active until the vehicle is commanded to land. The landing phase is managed manually and thus is not considered in the supervisor. The experiment is designed in such a way that the system will never try to return to its initial state, hence initial state conflicts will never arise.

In the second scenario, three states are present and the following transition rules are defined

$$\begin{aligned}&\text{R6: } \{\|\boldsymbol{p}_V - \boldsymbol{p}_{obs}\| \le d_s\} \text{ AND } \{\|\boldsymbol{p}_E - \boldsymbol{p}_{obs}\| \le d_s\},\\ &\text{R7: } \{\|\boldsymbol{p}_V - \boldsymbol{p}_{obs}\| \le d_s\} \text{ AND } \{\|\boldsymbol{p}_E - \boldsymbol{p}_{obs}\| > d_s\},\\ &\text{R8: } \{\|\boldsymbol{p}_V - \boldsymbol{p}_{obs}\| > d_s\} \text{ AND } \{\|\boldsymbol{p}_E - \boldsymbol{p}_{obs}\| \le d_s\}.\end{aligned}$$

Again the three rules are mutually exclusive, thus only one state can be active at each time instant.

Regarding the completeness, the admissible situations are

- the vehicle and/or the arm are not in the desired configuration for starting the end-effector trajectory: the system keeps the initial state VP+RNC;
- the vehicle and the arm reach the desired configuration with the desired accuracy: the system switches to the state EEC+MJL;
- no obstacles are in the safety area (namely in a sphere of radius d_s) both for the vehicle and the end-effector: the system keeps the state EEC+MJL;
- an obstacle is present in the safety area of the vehicle but not in the safety area of the end-effector and the vehicle is moving toward the obstacle: the supervisor switches to EEC+VOA+MJL;
- an obstacle is in the safety area both of the vehicle and the end-effector, and the vehicle is moving toward the obstacle: the supervisor switches to the state VOA+EEC+MJL;
- an obstacle is present in the safety area of the sole end-effector and the UAVM is moving toward the obstacle: the supervisor switches to the state EEOA +EEC+MJL;
- the obstacle is on the border of the safety area and the vehicle trajectory moves away from the obstacle: the obstacle avoidance is deactivated and the system switches to EEC+MJL.

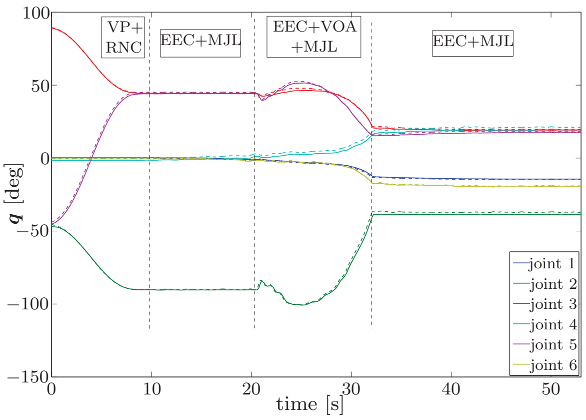


Fig. 17 Second case study: desired (dashed) and actual (solid) joint position.

Since any sequence of the previous situations is admissible, the completeness of the automata is proven.

In Fig. 17 the joint positions during the whole experiment are reported. It can be noticed that the joints 2, 3 and 5, modify their positions when the RNC task is active, while, during the second phase, all the joints keep a configuration almost constant. Thus, most of the motion is performed by the vehicle. Finally, during the obstacle avoidance, all the joints modify their positions in order to reconfigure the arm. Moreover, it can be noticed that a maximum error of about 4 deg is obtained during the motion.

Fig. 18 shows the end-effector position and orientation errors. It can be noticed that the maximum error occurs during the obstacle avoidance, when the vehicle modifies its trajectory and the arm tries to compensate this motion. Fig. 19 shows the distance between the vehicle center of mass and the obstacle (green line): despite the VOA behavior has not the highest priority, such distance is always close to the safety value. The red line in Fig. 19 represents the distance from the obstacle which would have been obtained in the presence of the compound behavior EEC+MJL, i.e., in the absence of the VOA elementary behavior. Such data have been obtained, for safety reasons, via numerical simulation. Finally, Fig. 20, shows the paths of the vehicle and the end-effector; it can be noticed that, despite the vehicle motion in the neighborhood of the obstacle, the end-effector tracks the desired linear path.

Some videos of the experiments are available at http://www2.unibas.it/automatica/multimedia.html

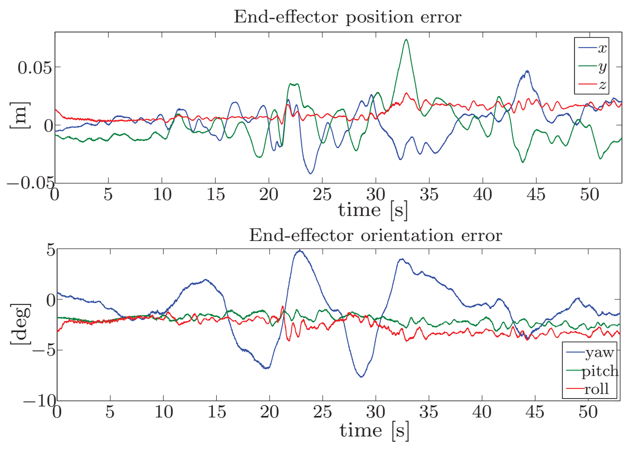


Fig. 18 Second case study: end effector position (top) and orientation (bottom) error.

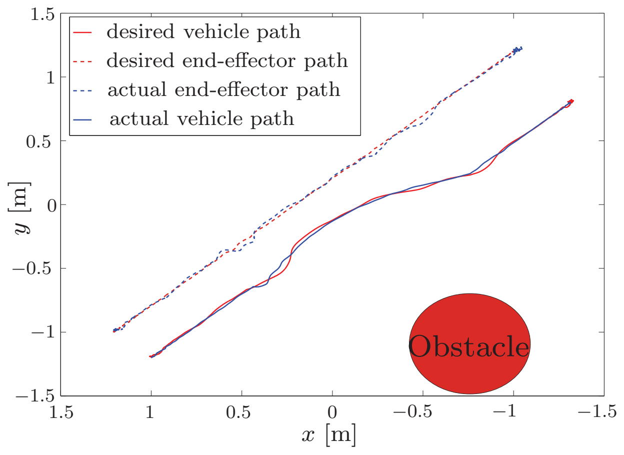


Fig. 20 Second case study: vehicle and end-effector paths on the xy plane

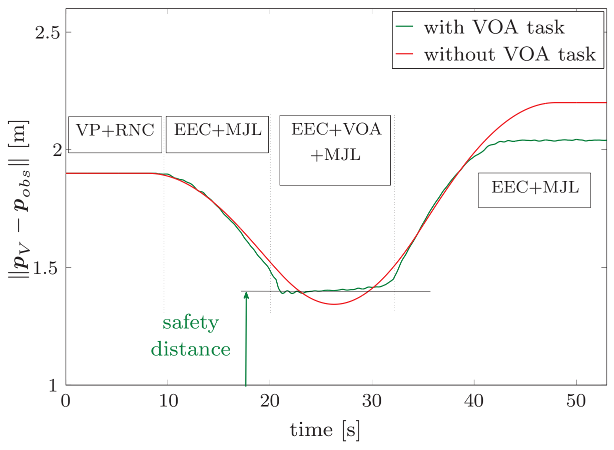


Fig. 19 Second case study: distance between the vehicle and the obstacle during the experiment, when the VOA behavior is activated (green line); distance between the obstacle and the vehicle reference trajectory obtained, via numerical simulation, in the absence of the VOA behavior (red line).

6 Discussion and future work

In this paper the NSB framework has been adopted to control a multirotor aerial vehicle equipped with a robotic arm, in order to ensure the vehicle-arm coordination and manage complex multi-task missions, where different behaviors must be encompassed in a clear and meaningful way. The method has been validated in a real testbed. The results show that, by properly designing a set of compound behaviors and a supervisor, vehicle-arm coordination in complex missions can be effectively managed. At the the best of authors' knowledge, this is the first time that an UAVM system is experimentally tested in the execution of complex multi-task missions. The overall performance of the system are satisfactory and present an end-effector tracking error of few centimeters, coherent with the available sensing and actuating technology. Such performance is strongly dependent upon the underlying motion control layer; in the current setup, in fact, the manipulator's joint servos are based on off-the-shelf components, characterized by a good performance/cost ratio, but also by a few limitations in terms of accuracy and bandwidth. Future versions of the experimental setup, based on more costly and accurate components, will allow to achieve more accurate tracking. The present research is part of a wider project whose final goal is the demonstration of a cooperative multi-UAVM system, aimed at executing complex coordinated tasks, such as, e.g., cooperative transportation of a payload and coordinated assembly operations. Thus, the research presented in this paper can be viewed as a necessary step towards the design and the deployment of multi-UAVM systems. To this purpose, a twin UAVM is under development, while cooperative coordinated control approaches for multi-UAVM systems are being designed.

Acknowledgements This research has been supported by the European Commission's 7th Framework Program under grant agreement No.287617 (IP project ARCAS - Aerial Robotics Cooperative Assembly system) and grant agreement No. 608849 (IP project EUROC - European Robotics Challenges), and by the European Commission's Horizon 2020 Program under grant agreement No. 644271 (IP project AEROARMS - AErial RObotic system integrating multiple ARMS and advanced manipulation capabilities for inspection and maintenance).

References

Alur R, Yannakakis M (2001) Model checking of hierarchical state machines. ACM Tranactions on Programming Languages and Systems 23(3):273–303

Antonelli G (2009) Stability analysis for prioritized closed-loop inverse kinematic algorithms for redundant robotic system. IEEE Transactions on Robotics 25:985–994

Antonelli G, Cataldi E (2014) Adaptive control of arm-equipped quadrotors. theory and simulations. In: Proceedings of 22th Mediterranean Conference on Control and Automation

Antonelli G, Arrichiello F, Chiaverini S (2008) The Null-Space-based Behavioral control for autonomous robotic systems. Journal of Intelligent Service Robotics 1(1):27–39

Antonelli G, Arrichiello F, Chiaverini S (2009) Experiments of formation control with multirobot systems using the null-space-based behavioral control. IEEE Transactions on Control Systems Technology 17(5):1173–1182

Antonelli G, Arrichiello F, Chiaverini S (2010) The NSB control: a behavior-based approach for multi-robot systems. Paladyn Journal of Behavioral Robotics 1(1):48–56

Antonelli G, Baizid K, Caccavale F, Giglio G, Muscio G, Pierri F (2014a) Control software architecture for cooperative multiple unmanned aerial vehicle-manipulator systems. Journal of Software Engineering for Robotics pp 123–134

Antonelli G, Baizid K, Caccavale F, Giglio G, Pierri F (2014b) Cooperative unmanned aerial vehicles manipulator systems: a control software architecture. In: 19th World Congress of the International Federation of Automatic Control, pp 1108–1113

ARCAS (2011) ARCAS - Aerial Robotics Cooperative Assembly System. URL http://www.arcas-project.eu

Arkin R (1998) Behavior-Based Robotics. The MIT Press, Cambridge, MA

Arleo G, Caccavale F, Muscio G, Pierri F (2013) Control of quadrotor aerial vehicles equipped with a robotic arm. In: Proc. of 21th Mediterranean Conference on Control and Automation, pp 1174–1180

Baizid K, Caccavale F, Chiaverini S, Giglio G, Pierri F (2014) Safety in coordinated control of multiple unmanned aerial vehicle manipulator systems: Case of obstacle avoidance. In: Proceedings of 22th Mediterranean Conference on Control and Automation

Baizid K, Giglio G, Pierri F, Trujillo M, Antonelli G, Caccavale F, Viguria A, Chiaverini S, Ollero A (2015) Experiments on behavioral coordinated control of an unmanned aerial vehicle manipulator system. In: Robotics and Automation (ICRA), 2015 IEEE International Conference on, IEEE, pp 4680–4685

Caccavale F, Giglio G, Muscio G, Pierri F (2014) Adaptive control for UAVs equipped with a robotic arm. In: Proceedings of the 19th World Congress The International Federation of Automatic Control (IFAC), pp 11,049–11,054

Cano R, Pérez C, Pruaño F, Ollero A, Heredia G (2013) Mechanical design of a 6-DOF aerial manipulator for assembling bar structures using UAVs. In: 2nd RED-UAS 2013 Workshop on Research, Education and Development of Unmanned Aerial Systems

Chiaverini S, Oriolo G, Walker ID (2008) Springer Handbook of Robotics, B. Siciliano, O. Khatib, (Eds.), Springer-Verlag, Heidelberg, D, chap Kinematically Redundant Manipulators, pp 245–268

Doitsidis L, Weiss S, Renzaglia A, Kosmatopoulos E, Siegwart R, Scaramuzza D, Achtelik M (2012) Optimal surveillance coverage for teams of micro aerial vehicles in gps-denied environments using onboard vision. Autonomous Robots

Escande A, Mansard N, Wieber PB (2014) Hierarchical quadratic programming: Fast online humanoid-robot motion generation. International Journal of Robotics Research 33(7):1006–1028

Fumagalli M, Naldi R, Macchelli A, Forte F, Keemink A, Stramigioli S, Carloni R, Marconi L (2014) Developing an aerial manipulator prototype: Physical interaction with the environment. IEEE Robotics Automation Magazine 21(3):41–50

Guarino Lo Bianco C, Zanasi R (2003) Smooth profile generation for a tile printing machine. Industrial Electronics, IEEE Transactions on 50(3):471–477

Huber F, Kondak K, Krieger K, Sommer D, Schwarzbach M, Laiacker M, Kossyk I, Parusel S, Haddadin S, Albu-Schaffer A (2013) First analysis and experiments in aerial manipulation using fully actuated redundant robot arm. In: Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on, pp 3452–3457

Kendoul F, Fantoni I, Lozano R (2008) Asymptotic stability of hierarchical inner-outer loop-based flight controllers. In: Proc. of the 17th IFAC World Congress, pp 1741–1746

Kim S, Choi S, Kim H (2013) Aerial manipulation using a quadrotor with a two dof robotic arm. In: Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on, pp 4990–4995

Kondak K, Krieger K, Albu Schaeffer A, Ollero A (2013) Closed-loop behavior of an autonomous helicopter equipped with a robotic arm for aerial manipulation tasks. International Journal of Advanced Robotic Systems 10:1–9

Kondak K, Huber F, Schwarzbach M, Laiacker M, Sommer D, Bejar M, Ollero A (2014) Aerial manipulation robot composed of an autonomous helicopter and a 7 degrees of freedom industrial manipulator. In: Robotics and Automation (ICRA), 2014 IEEE International Conference on, pp 2107–2112

Lippiello V, Ruggiero F (2012a) Cartesian impedance control of uav with a robotic arm. In: Proc. of 10th Int. IFAC Symp. on Robot Control, pp 704–709

Lippiello V, Ruggiero F (2012b) Exploiting redundancy in cartesian impedance control of uavs equipped with a robotic arm. In: Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp 3768–3773

Mansard N, Chaumette F (2009) Directional redundancy for robot control. Automatic Control, IEEE Transactions on 54(6):1179–1192

Maza I, Ollero A (2011) Autonomous transportation and deployment with aerial robots for search and rescue missions. Journal of Field Robotics 28(6):914931

Maza I, Kondak K, Bernard M, Ollero A (2010) Multi-UAV cooperation and control for load transportation and deployment. Journal of Intelligent and Robotic Systems 57:417–449

Mellinger D, Lindsey Q, Shomin M, Kumar V (2011) Design, modelling, estimation and control for aerial grasping and manipulation. In: Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp 2668–2673

Merino L, Caballero F, Martinez-de-Dios J, Maza I, Ollero A (2012) An unmanned aircraft system for automatic forest fire monitoring and measurement. Journal of Intelligent and Robotic Systems 65(1):533–548

Muscio G, Pierri F, Trujillo MA, Cataldi E, Giglio G, Antonelli G, Caccavale F, Viguria A, Chiaverini S, Ollero A (2016) Experiments on coordinated motion of aerial robotic manipulators. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), IEEE, pp 1224–1229

Orsag M, Korpela C, Bogdan S, Oh P (2013a) Lyapunov based model reference adaptive control for aerial manipulation. In: 2013 Int. Conf. on Unmanned Aircraft Systems (ICUAS), pp 966–973

Orsag M, Korpela C, Oh P (2013b) Modeling and control of MM-UAV: Mobile manipulating unmanned aerial vehicle. Journal of Intelligent and Robotic Systems 69:227–240

Ott C, Dietrich A, Albu-Schäffer A (2015) Prioritized multi-task compliance control of redundant manipulators. Automatica 53(1):416–423

Ouimet M, Lundqvist K (2007) Automated verification of completeness and consistency of abstract state machine specifications using a {SAT} solver. Electronic Notes in Theoretical Computer Science 190(2):85 – 97

Pounds P, Bersak D, Dollar A (2011) Vision based mav navigation in unknown and unstructured environments. In: Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA), pp 2491–2498

Quigley M, Conley K, Gerkey BP, Faust J, Foote T, Leibs J, Wheeler R, Ng AY (2009) ROS: an open-source robot operating system. In: ICRA Workshop on Open Source Software

Ruggiero F, Trujillo M, Cano R, Ascorbe H, Viguria A, Perez C, Lippiello V, Ollero A, Siciliano B (2015) A multilayer control for multirotor uavs equipped with a servo robot arm. In: Robotics and Automation (ICRA), 2015 IEEE International Conference on, pp 4014–4020

Santamaria D, Alarcon F, Jimenez A, Viguria A, Bejar M, Ollero A (2012) Model-based design, development and validation for uas critical software. Journal of Intelligent & Robotic Systems 65:103–114

Siciliano B (1990) Kinematic control of redundant robot manipulators: A tutorial. Journal of Intelligent and Robotic Systems 3(3):201–212

Siciliano B, Sciavicco L, Villani L, Oriolo G (2009) Robotics – Modelling, Planning and Control. Springer, London, UK

Simetti E, Casalino G, Torelli S, Sperindé A, Turetta A (2013) Floating underwater manipulation: Developed control methodology and experimental validation within the TRIDENT project. Journal of Field Robotics 31(3):364–385

VICON (Ltd.) Vicon Motion Systems. URL http://www.vicon.com